

Hybrid genetic search for the traveling salesman problem with hybrid electric vehicle and time windows

Yure Talis Rocha Silva



CENTRO DE INFORMÁTICA
UNIVERSIDADE FEDERAL DA PARAÍBA

João Pessoa
2023

Yure Talis Rocha Silva

Hybrid genetic search for the traveling salesman problem with hybrid electric vehicle and time windows

Monograph presented in fulfillment of the requirements for the Bachelor's degree in Computer Engineering, Centro de Informática, Universidade Federal da Paraíba.

Advisor: Anand Subramanian

João Pessoa

2023

Catálogo na publicação
Seção de Catalogação e Classificação

S586b Silva, Yure Talis Rocha.

Busca genética híbrida para o problema do caixeiro viajante com veículo elétrico híbrido e janelas de tempo / Yure Talis Rocha Silva. - João Pessoa, 2023.
38 f. : il.

Orientação: Anand Subramanian.
TCC (Graduação) - UFPB/CI.

1. Caixeiro viajante. 2. Veículo elétrico híbrido.
3. Algoritmo genético híbrido. 4. Busca local. I.
Subramanian, Anand. II. Título.

UFPB/CI

CDU 004.421



CENTRO DE INFORMÁTICA
UNIVERSIDADE FEDERAL DA PARAÍBA

Work entitled *Hybrid genetic search for the traveling salesman problem with hybrid electric vehicle and time windows* presented in fulfillment of the requirements for the Bachelor's degree in Computer Engineering, approved by:

Anand Subramanian

Prof. Dr. Anand Subramanian
Universidade Federal da Paraíba

Bruno Petrato Bruck

Prof. Dr. Bruno Petrato Bruck
Universidade Federal da Paraíba

Teobaldo Leite Bulhões Júnior

Prof. Dr. Teobaldo Leite Bulhões Júnior
Universidade Federal da Paraíba

Yuri Laio Teixeira Veras Silva

Prof. Dr. Yuri Laio Teixeira Veras Silva
Universidade Federal de Campina Grande

João Pessoa, June 23, 2023

To my brother.

Acknowledgments

I would like to thank Dr. Teobaldo Bulhões for many insightful discussions concerning the efficient move evaluation.

Resumo

O problema do caixeiro viajante com janelas de tempo, do inglês *traveling salesman problem with time windows* (TSPTW), é uma variante do problema clássico do caixeiro viajante, no qual clientes devem ser atendidos dentro de janelas de tempo. Este trabalho propõe uma busca genética híbrida para o TSPTW com veículo elétrico híbrido, o qual, comparado a veículos tradicionais, é mais amigável ao meio ambiente e ajuda a reduzir a emissão de gases de efeito estufa. A abordagem desenvolvida inclui um operador baseado no *order crossover* (OX) para melhorar soluções, juntamente com uma estratégia de limitação de busca e um sistema eficiente de avaliação de movimentos para acelerar a etapa de busca local. Experimentos computacionais foram realizados em mais de 200 instâncias de benchmark. O algoritmo proposto se mostrou efetivo em sistematicamente encontrar soluções de alta qualidade quando comparadas àquelas encontradas pela melhor heurística para o problema. Soluções melhores foram encontradas, especialmente para instâncias maiores e mais desafiadoras, nas quais o algoritmo se mostrou, em média, 9 vezes mais rápido do que o melhor método existente. Além disso, foi examinada a distribuição de frequência do uso dos modos de operação do veículo associados com as melhores soluções encontradas.

Palavras-chave: Caixeiro viajante, veículo elétrico híbrido, algoritmo genético híbrido, busca local.

Abstract

The *traveling salesman problem with time windows* (TSPTW) is a variant of the classical *traveling salesman problem* (TSP), in which customers must be served within specific time windows. This work proposes a hybrid genetic search for the TSPTW considering a *hybrid electric vehicle* (HEV), which is more environmental friendly than conventional vehicles and helps to decrease the emission of greenhouse gases. The developed approach includes a specific operator based on the order crossover (OX) to obtain improved solutions, as well as a search limitation strategy and an efficient move evaluation scheme to speed up the local search phase. Extensive computational experiments were conducted on more than 200 benchmark instances. The proposed algorithm was revealed to be effective in systematically finding high-quality solutions when compared to those achieved by the best heuristic for the problem. Improved solutions were found, especially for the larger and more challenging cases, for which our algorithm performed, on average, 9 times faster than the quickest method available. Moreover, we examine the frequency distribution of the operation mode usage of the vehicle associated with the best solutions found.

Key-words: Traveling salesman, hybrid electric vehicle, hybrid genetic algorithm, local search.

Contents

Glossary	v
List of Figures	vii
List of Tables	viii
1 Introduction	1
1.1 Preliminaries	1
1.2 Motivation	2
1.3 Objectives	3
1.4 Monograph outline	3
2 Related work	4
3 Problem description	7
4 Proposed algorithm	9
4.1 Search space	10
4.2 Initial population	11
4.3 Evaluation of individuals	11
4.4 Computation of the <i>average distance</i>	12
4.5 Computation of the <i>biased fitness</i>	12
4.6 Subpopulation management	13
4.7 Parent selection and offspring generation	13
4.8 Local search	14
4.9 Move evaluation	17
4.10 Concatenation of subsequences	19
5 Computational experiments	26
5.1 Determining the parameter values, neighborhood structures and search strategy	26
5.2 Comparison with the literature	29

5.3	Impact of the newly proposed offspring generation procedure	34
5.4	Impact of the efficient move evaluation scheme	36
5.5	Impact of restricting the number of arc combinations during local search .	36
5.6	Impact of the instance size on the operation modes	36
6	Concluding remarks and future work	38
	Bibliography	38

Glossary

ALNS	:	Adaptive large neighborhood search
BF	:	Biased fitness
EFSMFTW	:	Electric fleet size and mix vehicle routing problem with time windows and recharging stations
EV	:	Electric vehicle
EVRP	:	Electric vehicle routing problem
EVRPNL	:	Electric vehicle routing problem with nonlinear charging function
EVRPTW	:	Electric vehicle routing problem with time windows
EVRPTWMF	:	Electric vehicle routing problem with time windows and multiple, full recharges
EVRPTWMP	:	Electric vehicle routing problem with time windows and multiple, partial recharges
EVRPTWPR	:	Electric vehicle routing problem with time windows and partial recharges
EVRPTWSF	:	Electric vehicle routing problem with time windows and single, full recharge
EVRPTWSP	:	Electric vehicle routing problem with time windows and single, partial recharge
ERX	:	Edge recombination crossover
ERXMC	:	Edge recombination crossover with mode changes
GVRP	:	Green vehicle routing problem
GRASP	:	Greedy randomized adaptive search procedure
HEV	:	Hybrid electric vehicle
HEVRP	:	Hybrid electric vehicle routing problem
HEVTSP	:	Hybrid electric vehicle traveling salesman problem
HEVTSPTW	:	Hybrid electric vehicle traveling salesman problem with time windows
HGS	:	Hybrid genetic search
HRVND	:	Hierarchical randomized variable neighborhood descent
HVRP	:	Hybrid vehicle routing problem
ICE	:	Internal combustion engine
ILS	:	Iterated local search
LNS	:	Large neighborhood search
MILP	:	Mixed integer linear programming
MIP	:	Mixed integer programming
MOX	:	Modified order crossover
MOXMC	:	Modified order crossover with mode changes

OX	: Order crossover
OXMC	: Order crossover with mode changes
PHEV	: Plugin hybrid electric vehicle
PVNS	: Parallel variable neighborhood search
PSO	: Particle swarm optimization
QO	: Quality optimized
RO	: Runtime optimized
RVND	: Randomized variable neighborhood descent
SA	: Simulated annealing
SCX	: Sequential constructive crossover
SCXMC	: Sequential constructive crossover with mode changes
SVND	: Sequential variable neighborhood descent
TS	: Tabu search
TSP	: Traveling salesman problem
VNS	: Variable neighborhood search
VNSB	: Variable neighborhood search with branching
VRP	: Vehicle routing problem
3MC	: 3 mode change

List of Figures

1	Example of a solution regarding time windows and operation modes, with a maximum battery level of 10.	8
2	Arcs and operation modes of S	12
3	MOXMC operator.	14
4	Reinsertion of customer 6 with evaluation of the operation modes of the new arcs.	16
5	Solution S	21
6	All subsequences associated with a 6-customer solution.	22
7	Illustration of a reinsertion move regarding subsequences.	24
8	Impacts of the number of iterations and the search strategy.	29
9	Impact of different crossover operators.	35
10	Relation between instance length and the operation mode use.	37

List of Tables

1	Data associated with each operation mode for an arc (i, j) in an arbitrary solution.	7
2	Chromosomes extracted from S	12
3	Combinations of operation modes evaluated with respect to the neighborhood structures.	17
4	Time and battery specifications of the 6-customer instance.	20
5	Values of Equations (9)–(12) for every subsequence of the 6-customer solution.	23
6	Computation of Equations (9)–(12) for concatenations of subsequences associated with a reinsertion move.	25
7	Results obtained for five different settings of neighborhood structures. . . .	27
8	Search strategy settings.	28
9	Results obtained on the 8-customer instances.	30
10	Results obtained on the 10-customer instances.	31
11	Results obtained on the 20-customer instances.	32
12	Results obtained on the 50-customer instances.	33
13	Summary of the results found by HGS _{HEV} compared with those of versions RO and QO of PVNS.	34
14	Number of cases in which the average solution of one operator was better than the other.	35
15	Speed-up achieved by using the efficient move evaluation scheme.	36
16	Impact of restricting the number of arc combinations during local search. .	36
17	Average length of the arcs, in meters, per operation mode and instance size.	37

1 Introduction

1.1 Preliminaries

The *traveling salesman problem* (TSP) is a combinatorial optimization problem and one of the most prominent in the field of operations research. In the TSP, each of a set of customers must be visited exactly once, and the vehicle must start from and return to a depot. Moreover, the sequence of visited customers is considered a solution to the problem, the overall distance can be regarded as the cost, and the optimal solution is the one with the least cost.

Different variants of the TSP have been proposed over the years. For instance, the TSP *with time windows* (TSPTW), in which every customer has a service time and must be served within specific start and end times of a time window. A solution to the TSPTW is said to be *feasible* if it satisfies the time-window constraints.

The *hybrid electric vehicle* TSPTW (HEVTSPW) is a green variant of the TSP, which adopts a more environmental friendly vehicle instead of the conventional *internal combustion engine* (ICE) one. In addition to the ICE, HEVs can also benefit from an electric motor. Green logistics considers not only economic factors of logistics activities, but also the sustainability of the process [Sbihi and Eglese, 2007].

In combinatorial problems such as the TSPTW, the *search space* is composed of a finite set of solutions. *Exact* and *heuristic* algorithms can be applied to find the optimal solution in the search space. The former approach, searches for all possible feasible solutions and returns the optimal one, but usually takes a lot of computational time. The latter, uses different techniques to rapidly converge to good solutions, on average, even though there is no guarantee that the optimal one will be found.

The *hybrid genetic search* (HGS) is a heuristic approach that combines *population-based search* and *local search*. In population-based heuristics, characteristics of a set of solutions, or individuals, are used to iteratively guide the search for improved solutions. This is also the rationale behind *genetic algorithms*. *Local search* methods look for improved solutions by moving, by means of incremental changes, from solution to solution. Moreover, the HGS relies on an offspring generation operator applied to individuals selected from the population. The idea of the crossover operator is to select characteristics from promising parents to be passed on to offsprings.

This work presents a *hybrid genetic search* approach to solve the *hybrid electric vehicle traveling salesman problem with time windows*.

1.2 Motivation

Over the past decades, the application of operations research in the fields of logistics and transportation led to substantial savings. In that sense, ICE vehicles have been of paramount importance. Nevertheless, they also have been contributing negatively to climate changes due to their significant share of greenhouse gas emissions. The European Union, for instance, aims at reducing, by 2030, 55% of the level of emission of such gases in comparison to the amount produced in 1990 [Zhongming et al., 2020]. To this end, two threads of operation might be considered: efficient exploitation of currently available resources or the use of more environmental friendly technologies. The former alternative has the potential of reducing, in some cases, up to 6.9% of pollutant emissions, as can be observed in Suzuki [2011]. In such work, the objective is to minimize the time a vehicle spends carrying a heavy payload and the waiting times to serve the customers. The latter alternative is a hot topic that has been the subject of many scientific papers (see, e.g., Lin et al. [2014]) and has a greater potential for minimizing the emission of greenhouse gases.

A popular environmental friendly technology for last mile deliveries is the *electric vehicle* (EV) [Mancini, 2017]. This type of vehicle, however, has limitations regarding its maximum driving range, due to battery capacity, as well as its charging time. The HEVs present a reasonable solution to the driving range constraint and the charging time problem by combining an electric motor and an ICE. Unlike *plugin* HEVs (PHEVs), which can also be connected to power grids to recharge their batteries, HEVs can only be charged while the vehicle operates with the ICE [Ghorbani et al., 2020].

Four distinct operation modes are assumed in the HEV considered in this article: combustion, charging (while driving with the combustion engine), boost (which is a combination of the combustion engine and the electric motor), and electric. Moreover, the operation mode can only be changed at the vertices. The time and cost required to traverse an arc depend on the selected mode, and each of them has a different impact on the battery charging level. For more details about the vehicle, the reader should refer to AG [2021].

The HEVTSPTW, proposed in Doppstadt et al. [2020], generalizes the well-known TSPTW (so it is also $\mathcal{NP}$ -hard) by employing an HEV instead of a regular combustion vehicle. In this case, the traveling costs depend on the operation mode (i.e., combustion, charging, boost, or electric) of the vehicle when going from one point to the other, and one must ensure that the battery level is never violated during the tour.

1.3 Objectives

The main objective of this work is to propose a *hybrid genetic search* (HGS) [Vidal et al., 2014] algorithm to solve the HEVTSPTW, which incorporates a *hierarchical randomized variable neighborhood descent* (HRVND) procedure in the local search phase. The remaining objectives are listed as follows.

- Propose a novel crossover operator that exploits the particular characteristics of the HEVTSPTW, such as the battery operation modes. The efficiency of such operator is attested by comparing its performance with several others, including well-known operators from the literature.
- Devise a simple limitation strategy that avoids all possible arc combinations associated with battery operation modes to be evaluated during local search, and empirically demonstrate that such strategy can yield a significant speed-up without compromising the solution quality.
- Develop an efficient procedure for computing violations on the battery level in amortized $\mathcal{O}(1)$ operations, which substantially improves the runtime performance.
- Examine the frequency distribution of the operation mode usage of the vehicle associated with the best solutions found. We could observe that one of the modes is likely to be used more as the instance size increases.

1.4 Monograph outline

The remainder of this work is organized as follows.

- Section 2 reviews the related literature.
- Section 3 formally defines the problem.
- Section 4 describes the proposed algorithm, as well as the efficient move evaluation and limitation strategy schemes.
- Section 5 contains the results of extensive computational experiments.
- Finally, Section 6 concludes.

2 Related work

Many works have studied the use of EVs as an alternative solution for decreasing the amount of greenhouse gas emissions in routing problems. Some of them have EVs as part of homogeneous or heterogeneous fleets of *vehicle routing problems* (VRPs), or as the single vehicle used in TSP variants. This section reviews the problems that are closely related to the variant addressed in this paper. The reader is referred to Lin et al. [2014], Erdelić and Carić [2019], and Qin et al. [2021] for a comprehensive review on electric VRP variants.

The *green* VRP (GVRP), proposed by Erdoğan and Miller-Hooks [2012], is one of the most prominent examples of the use of EVs in VRP-like variants. In such work, the vehicle can only be charged to full battery capacity at specific stations, and the charging time is constant. A general *electric* VRP (EVRP) is presented in Lin et al. [2016], in which an exact approach is used to find a minimal cost solution based on travel time, energy consumption and the amount of EVs dispatched. Furthermore, such work is claimed to be the first EVRP model to study the relationship between vehicle load and battery consumption.

Introduced by Schneider et al. [2014] and also studied (among others) by Bruglieri et al. [2015], the EVRPTW is a time-window extension of the EVRP, in which the battery level and the recharging time (after being recharged, the battery is assumed to be full) depend on the energy consumption. Both works presented a *mixed integer linear programming* (MILP) formulation, but the former devised a hybrid solution method based on *variable neighborhood search* (VNS) [Hansen et al., 2019, Sifaleras and Konstantaras, 2020, Lan et al., 2021] and *tabu search* (TS) [Laguna, 2018], while the latter implemented a so-called *VNS with branching* (VNSB) procedure [Hansen et al., 2006], which combines VNS with *local branching* [Fischetti and Lodi, 2003]. Moreover, four variants of the EVRPTW were addressed by Desaulniers et al. [2016], namely: EVRPTWSF, EVRPTWMF, EVRPTWSP, and EVRPTWMP. The main difference among these variants is in the maximum number of recharges per route (at most one – S, multiple recharges – M), the fact that batteries have to be fully recharged at each visit to a single station (F), and the possibility to partially recharge the battery (P). In such work, solutions were obtained by exact branch-price-and-cut algorithms.

The EVRPTWPR is another EVRPTW variant, proposed by Keskin and Çatay [2016], where the full recharging constraint is relaxed, allowing *partial recharge* of the battery. The authors formulated the problem as a 0-1 MILP and developed an *adaptive large neighborhood search* (ALNS) [Pisinger and Ropke, 2019] heuristic to solve it. Furthermore, the EVRPNL introduced by Montoya et al. [2017] adds a nonlinear charging function, and the resulting problem is solved by a hybrid metaheuristic consisting

in the combination of *iterated local search* (ILS) [Lourenço et al., 2019] and the so-called heuristic concentration [Rosing and ReVelle, 1997]. Finally, the *electric fleet size and mix VRP with time windows and recharging stations* (EFSMFTW) was put forward by Hiermann et al. [2016], which essentially integrates the characteristics of the classical *fleet size and mix* VRP with those of the EVRPTW. The problem is solved exactly by a branch-and-price algorithm and heuristically via a hybrid ALNS approach.

The *hybrid* VRP (HVRP) is another extension of the GVRP. Two early works on this problem are those by Vincent et al. [2017] and Mancini [2017]. The former proposed a *simulated annealing* (SA) [Delahaye et al., 2019] heuristic for an HFVRP that employs PHEVs, while the latter put forward a *large neighborhood search* (LNS) [Pisinger and Ropke, 2019] based matheuristic for an HFVRP with a fleet of EVs. A more complicated version of the HVRP with PHEVs was later studied by Li et al. [2020], who devised a heuristic algorithm that combines *memetic algorithm* with *sequential variable neighborhood descent* (SVND). Moreover, Murakami [2018] studied a routing and scheduling problem using a single PHEV and implemented two exact approaches, more precisely, a *mixed integer programming* (MIP) model and a labeling-based algorithm, as well as two heuristics. Bahrami et al. [2020] proposed an HVRP that also considers power management optimization, and solved it by means of branch-and-price and heuristic algorithms. Hiermann et al. [2019], on the other hand, studied a heterogeneous fleet VRP with combustion, plug-in hybrid, and electric vehicles. Solutions were obtained via a hybrid approach, i.e., combining metaheuristic and exact algorithms. More details about hybrid VRPs can be found in the surveys by Dascioglu and Tuzkaya [2019] and Ammar et al. [2022].

Zhen et al. [2020] and Seyfi et al. [2022] are two seminal works on *hybrid electric* VRPs (HEVRPs) *with mode selection*. The former put forward an improved *particle swarm optimization* (PSO) [Poli et al., 2007] algorithm to solve an HEVRP with a fleet of PHEVs, with the delivery area composed of a depot, and sets of (i) customers, (ii) charging, and (iii) gas stations. The latter, on the other hand, solved the HEVRP by means of a matheuristic that combines VNS with *mathematical programming*.

Concerning TSP-like problems, Chau et al. [2016] developed an approach for selecting the drive modes of a PHEV based on prediction using historical data. In this sense, a driving profile is obtained, which takes into account parameters such as traffic conditions. Such profile is used in a formula together with driving speed, vehicle weight, and other metrics to determine the appropriate drive mode at each part of the road.

Another single vehicle variant is the *hybrid electric vehicle traveling salesman problem* (HEVTSP), introduced by Doppstadt et al. [2016]. To solve it, the authors developed a mathematical formulation and a TS heuristic. Doppstadt et al. [2020] later extended the HEVTSP, as well as the formulation, by adding time-window constraints (HEVTSPTW), and they solved the problem heuristically via a *parallel* VNS (PVNS) algorithm. More

specifically, they divided their solution approach into two phases: initialization and improvement. The former is further divided into three steps: firstly, customers are ordered according to their time window end time in non-decreasing order (operation modes are set to combustion at this step); secondly, insertion-moves are employed using a so-called *hill-climbing* procedure with the aim of modifying the position of single customers in the route; lastly, the *3-mode change* (3MC) approach [Doppstadt et al., 2016] (see Section 4.8) is executed as a hill-climbing procedure. The latter phase consists in applying the PVNS algorithm and infeasible solutions are allowed during the search.

Given the above, we can observe that the amount of works addressing hybrid electric routing problems without recharging stations is rather limited. Moreover, we have also identified opportunities for devising more efficient heuristic procedures to solve this very challenging class of problems when compared to existing works. Even when only a single hybrid vehicle is considered, the best available strategies require from several minutes to a few hours to achieve very good solutions for 50-customer instances when time windows are taken into account. Therefore, our work attempts to make relevant methodological progress when it comes to obtaining high-quality solutions for this latter case in a matter of seconds.

3 Problem description

Let $G = (V, A)$ be a directed graph, $V = \{0, 1, 2, \dots, n\}$ be a set of vertices, and $A = \{(i, j) \in V^2, i \neq j\}$ be a set of arcs. For each vertex $i \in V$, there is a time window $[b_i, e_i]$ in which such vertex can be visited, and a service time s_i . Vertex 0 is the depot and the remaining ones are the customers. Moreover: (i) $e_i - b_i \geq s_i, \forall i \in V$; and (ii) $e_0 - b_0$ provides the maximum route duration (or working hours).

Each arc $(i, j) \in A$ has a cost and a travel time for each operation mode of the vehicle, i.e, combustion (c), charging (ch), electric (e), and boost (b), as indicated in Table 1. Regarding the battery, there is a minimum (0) and a maximum l_{max} charging level, as well as charging r_c and discharging r_d rates. In addition, l_i is the battery charging level at vertex $i \in V$. Each mode of operation has a different impact on the battery when traversing an arc, as also presented in Table 1.

Tabela 1: Data associated with each operation mode for an arc (i, j) in an arbitrary solution.

Operation mode	Arc cost	Arc travel time	l_j
Combustion (c)	c_{ij}^c	t_{ij}^c	l_i
Charging (ch)	c_{ij}^{ch}	t_{ij}^{ch}	$l_i + \min(l_{max} - l_i, t_{ij}^{ch} r_c)$
Electric (e)	c_{ij}^e	t_{ij}^e	$l_i - \min(l_i, t_{ij}^e r_d)$
Boost (b)	c_{ij}^b	t_{ij}^b	$l_i - \min(l_i, t_{ij}^b r_d)$

For each mode, the last column in Table 1 ensures that the battery level stays between 0 and l_{max} . Therefore, we can conclude that:

- in combustion, the charging level is not affected;
- in charging, the increasing level is proportional to the time required to traverse the arc in this mode multiplied by the charging rate;
- in electric or boost, the power consumption is proportional to the time required to traverse the arc in this mode multiplied by the discharging rate.

The objective of the HEVTSPTW is to find the least-cost Hamiltonian cycle, starting and ending at the depot, serving each customer within its specific time window and with a non-negative battery level throughout the tour. A mathematical formulation for the problem, as well as a more comprehensive description regarding the HEVTSPTW constraints, can be found in Doppstadt et al. [2020].

Figure 1 depicts an example of a solution containing a depot and three customers, with the associated time windows indicated in brackets and the arrival times showed just below them. The battery charging level is depicted to the right of the nodes. Assuming

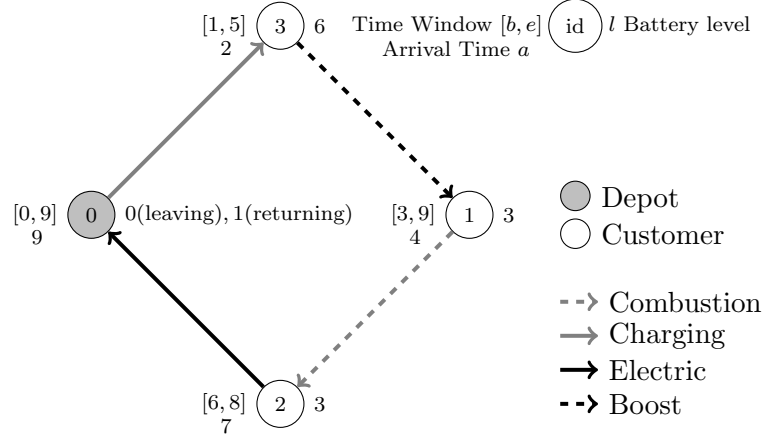


Figure 1: Example of a solution regarding time windows and operation modes, with a maximum battery level of 10.

that the battery charging level at the depot is 0, the vehicle can only use the charging or combustion mode in order to keep the non-negative battery level requirement. In the example, the charging mode is selected, and thus the battery is partially recharged. From customer 3 to 1, the vehicle can use the combustion or charging mode, and, depending on the battery level at 3, the electric or boost mode. Assuming that there is enough charge available on the battery, the boost mode is selected. Moreover, one can only use the combustion or charging mode from customer 1 to 2, if we assume that the use of the electric or boost mode would incur in a negative battery level. In such arc $(1, 2)$, we also have to take into account the time window requirement, i.e. the use of a mode with a long travel time might incur in a time window violation. Hence, with the aim of reaching customer 2 within its time window, the combustion mode is selected, as it offers the least travel time between the two available options. Finally, the vehicle returns to the depot in electric mode, partially consuming the remaining charge on the battery.

4 Proposed algorithm

The proposed algorithm is based on the HGS framework that includes an HRVND procedure in the local search phase. At each iteration, two parents, selected from a population of individuals, undergo a crossover operator to create a new individual to be potentially improved by the local search. New individuals are stored in subpopulations according to their feasibility. The pseudocode of our method, here denoted as HGS_{HEV}, is presented in Algorithm 1. The meaning of each input parameter is explained throughout this section.

Algorithm 1 HGS_{HEV}

```

1: procedure HGS-RVND( $N_{it}, \theta, \alpha, \mu, \mu_{elite}, \mu_{close}, \lambda$ )
2:    $\Omega, \Omega' \leftarrow \mathbf{InitializePopulation}(\theta, \alpha, \mu, \mu_{elite}, \mu_{close}, \lambda)$ 
3:    $s \leftarrow$  Select the individual with the best cost out of the subpopulations  $\Omega \cup \Omega'$ .
4:    $it \leftarrow 0$ 
5:   while  $it < N_{it}$  do
6:      $it \leftarrow it + 1$ 
7:      $P_1 \leftarrow \mathbf{BinaryTournament}(\Omega, \Omega')$ 
8:      $P_2 \leftarrow \mathbf{BinaryTournament}(\Omega, \Omega')$ 
9:      $S \leftarrow \mathbf{MOXMC}(P_1, P_2)$ 
10:     $S \leftarrow \mathbf{HRVND}(S)$ 
11:    if  $\mathbf{F}(S)$  then
12:      Insert  $S$  into  $\Omega$ 
13:       $\Omega \leftarrow \mathbf{SubpopulationManagement}(\Omega, \mu, \mu_{elite}, \mu_{close}, \lambda)$ 
14:      if  $f(S) < f(s)$  then
15:         $s \leftarrow S$ 
16:         $it \leftarrow 0$ 
17:    else
18:      Insert  $S$  into  $\Omega'$ 
19:       $\Omega' \leftarrow \mathbf{SubpopulationManagement}(\Omega', \mu, \mu_{elite}, \mu_{close}, \lambda)$ 
20:  return  $s$ 

```

HGS_{HEV} starts by creating two subpopulations of solutions: the first containing only feasible individuals (Ω) and the second with only the infeasible ones (Ω') (line 2). The solution with the best (i.e. least) cost (see Equation (1)), considered as the global solution, is then selected from the population (line 3) and a counter is set to 0 (line 4). The main loop of HGS_{HEV} (lines 5 to 19) is executed at most N_{it} times without improvement on the global solution. Two binary tournaments are performed (lines 7 and 8) at each iteration to choose two individuals from the population. For every tournament, two individuals P_1 and P_2 are randomly selected from each subpopulation and the one with the best biased fitness wins. The individuals returned from the binary tournament are used to generate an offspring by means of a *modified order crossover with mode changes* (MOXMC). This offspring undergoes an HRVND (line 10) and according to its feasibility it is included in

the corresponding subpopulation. The **F** procedure (line 11) returns *true* if the solution S is feasible and *false*, otherwise, and function f (line 14) returns the cost of an individual. If the solution S is feasible (line 11), it is placed in the feasible subpopulation (line 12) and in case it has a better cost than the best individual s found so far (line 14), then it becomes the global solution (line 15) and the counter is set to 0 (line 16). Otherwise, if S is infeasible (line 17), it is inserted into the infeasible subpopulation (line 18). Whenever an individual is added to a subpopulation, a population management procedure is executed (lines 13 and 19). At the end of the algorithm, the global (best) solution is returned (line 20).

The next subsections thoroughly describe the main aspects of the proposed algorithm, namely: search space; constructive procedure and criterion used to insert an individual in a population; fitness function; subpopulations management; binary tournament and the crossover operator; local search operators; efficient move evaluation; and limitation strategy.

4.1 Search space

The HGS_{HEV} explores both feasible and infeasible solutions of the search space. The infeasibility may be caused by late arrivals at customers (or at the depot) or due to negative battery levels across vertices. The simplest violation is excess duration, which occurs whenever the vehicle arrives back at the depot after its time-window ending time. Moreover, when the vehicle arrives late at a given customer, a “time-warp” is necessary in order to serve it within its time window. The time-warp can be thought of as a kind of time travel, and it is also a violation. Finally, a negative battery level occurs whenever the vehicle departs from an origin vertex to a destination one, in electric or boost mode, without enough charge to traverse the arc connecting them. In this case, we consider that the vehicle is able to reach the destination and the negative amount of charge consumed is considered a violation. The sum of the negative battery levels on all arcs of a solution constitutes the battery violation.

In order to properly evaluate the solution quality, taking into account possible violations, we decided to adopt an objective function similar to the one discussed in Vidal et al. [2012]. This approach also allows one to compare feasible and infeasible individuals, and, in our case, the former is always preferred over the latter, even if it has a worse cost. Let x_{ij}^m be a binary variable that takes value 1 if arc $(i, j) \in A$ in the operation mode $m \in \{c, ch, e, b\}$ is selected; and 0, otherwise. Therefore, the objective function can be expressed as follows:

$$\min \sum_{(i,j) \in A} (c_c x_{ij}^c + c_{ch} x_{ij}^{ch} + c_e x_{ij}^e + c_b x_{ij}^b) + \omega^d d + \omega^{tw} tw + \omega^c v, \quad (1)$$

where ω^d , ω^{tw} and ω^c represent the penalties for violating route duration, time windows and battery level constraints, respectively; and d , tw and v represent excess duration, time-warp use and negative battery level, respectively.

On the one hand, based on empirical analyses, duration and battery charging do not seem to be the most challenging constraints. Hence, the values for ω_d and for ω_v were set to 1. On the other hand, due to the tightness of some time windows, we implemented a basic preprocessing routine to compute the value of ω^{tw} for each instance as follows:

$$\omega^{tw} = m^{tw} - (m^{tw} - 1) \times (avg^{tw} / max^{wh}),$$

where avg^{tw} is the average length of the time windows of the instance and, $max^{wh} = 480$ corresponds to the maximum working hours of the benchmark instances. The parametric expression is intended to keep the values of ω^{tw} within the range of 1 to m^{tw} . Such range was empirically selected after conducting experiments on challenging instances (see Section 5) with different values for m^{tw} , namely 1, 10, 50, and 100. While value 1 and 10 yielded many infeasible solutions, 100 led to more feasible solutions, but with inferior quality than those achieved by $m^{tw} = 50$. We thus chose the latter value as it offered an interesting compromise between the exploration of feasible and infeasible solutions.

4.2 Initial population

At each iteration, a solution is created by applying the constructive approach of the metaheuristic *greedy randomized adaptive search procedure* (GRASP) [Feo and Resende, 1995]. The parameter that controls the greediness/randomness of the construction step is α . The newly generated individual is then potentially improved by means of local search. At this stage, the algorithm checks whether there is a solution in the population with the same cost of such individual, which can be seen as a *clone* according to Vidal et al. [2012]. If so, it is discarded and the constructive procedure restarts. Otherwise, the new individual is placed in the appropriate subpopulation with respect to its feasibility status.

4.3 Evaluation of individuals

The evaluation function of an individual considers both its contribution to the diversity of the population and its quality. Hence, two indicators are of most importance, the *average distance* ($\overline{D}$) of an individual to its μ_{close} closest neighbors and its cost according to Equation (1), which in turn are combined to form the *biased fitness* (BF) of a solution, as in Vidal et al. [2013].

4.4 Computation of the *average distance*

Figure 2 depicts a solution S with 6 customers and the depot. Two types of information are extracted from this individual for the computation of $\overline{D}$, as shown in Table 2, namely: (i) the arcs chromosome $\phi_i(S)$, which is the sequence of arcs used in the solution; and (ii) the modes chromosome $\pi_i(S)$, which corresponds to the selected operation modes of such arcs.

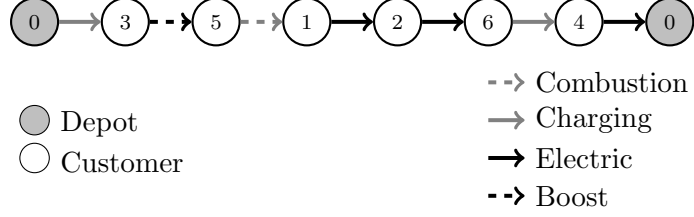


Figure 2: Arcs and operation modes of S .

Tabela 2: Chromosomes extracted from S .

Solution arcs i	(0, 3)	(3, 5)	(5, 1)	(1, 2)	(2, 6)	(6, 4)	(4, 0)
Arcs chromosome $\phi_i(S)$	(0, 3)	(3, 5)	(5, 1)	(1, 2)	(2, 6)	(6, 4)	(4, 0)
Modes chromosome $\pi_i(S)$	r	b	c	c	r	e	e

The *distance* $\delta(S_1, S_2)$ between two individuals S_1 and S_2 is calculated as follows:

$$\delta(S_1, S_2) = \frac{1}{2n} \sum_{i=1,2,\dots,n} (\mathbf{V}(\phi_i(S_1), \phi_i(S_2)) + \mathbf{V}(\pi_i(S_1), \pi_i(S_2))), \quad (2)$$

where the binary operator $\mathbf{V}$ returns 1 if both arguments are different, and 0 otherwise; n is the number of arcs in the solution.

Whenever an individual is added to a subpopulation, it is necessary to compute its own list of closest neighbors (chosen among all individuals from the corresponding subpopulation), as well as to update such list for the remaining solutions of the subset. It is also mandatory to update the same lists when removing a solution from a subset. The average distance of S to its closest neighbors is computed as follows:

$$\overline{D}(S) = \frac{1}{\mu_{close}} \sum_{i=1,2,\dots,\mu_{close}} \delta(S, S_i). \quad (3)$$

4.5 Computation of the *biased fitness*

Two ranks are created and continuously updated for each subpopulation during the HGS_{HEV} execution: the cost rank (f_p), in which the subpopulation is sorted in non-decreasing order of cost computed using Equation (1); and the average distance rank

(f_d), in which the subpopulation is sorted in non-ascending order of average distance. We used the same sorting scheme as in Mecler et al. [2021]. Moreover, every individual S is associated with a position in one of such ranks. The BF is then calculated as follows.

$$BF(S) = f_p(S) + \left(1 - \frac{\mu_{elite}}{n}\right) f_d(S), \quad (4)$$

where μ_{elite} is the number of elite individuals preserved for the next generation.

4.6 Subpopulation management

Each subpopulation has a maximum capacity of $\mu + \lambda$ individuals. When one tries to insert one more solution in a subpopulation of size $\mu + \lambda$, the worst λ solutions, regarding their biased fitness, are discarded. Thus, only the μ best individuals survive for the next generation. Furthermore, the number of solutions in each subpopulation depends on their feasibility and is not necessarily proportional.

4.7 Parent selection and offspring generation

In the binary tournament, one individual from each subpopulation is randomly selected and the one with the best BF wins. As there is no guarantee that a subpopulation is not empty, the two individuals may be chosen from the same subset.

We devised a specific *order crossover* (OX) [Oliver et al., 1987] with a view of trying to conserve the beginning and the ending subsequences of one parent chromosome in the offspring, thus giving preference for changes in the middle subsequence. The rationale behind this idea is that the combustion and the charging modes are likely to be used in the beginning of a subsequence, as the charging level is initially set to zero, whereas the electric and the boost modes are likely to be used in the end of a subsequence, as possibly enough charging level is available on the battery. The middle of a subsequence, on the other hand, has more room for changes because the battery level is probably not zero anymore, thus more combinations can be evaluated. We call such new operator *modified order crossover* (MOX). In this case, the operator preserves the modes chromosome of a random parent in the child. Nevertheless, we also implemented an alternative version of the operator, here denoted as *MOX with mode changes* (MOXMC), by selecting each operation mode of the offspring based on parent and customer, as further explained. This latter version of the operator yielded better solutions, on average, as shown in Section 5.3, and it was the one adopted in our algorithm.

Figure 3 illustrates the execution of the MOXMC on two individuals P_1 and P_2 . The execution of the MOX is similar regarding the subsequences, with the difference relying on the selection of the operation modes, as previously mentioned.



4.8 Local search

- *Reinsertion* – a customer is removed and inserted in another position of the tour.
- *Or-opt-k* – k consecutive customers are removed and inserted in another position of the tour.
- *2-opt* – two non-adjacent arcs are removed and two others are inserted to form a new route.
- *Exchange* – permutation move between two customers.
- *Swap-k* – k consecutive customers are swapped with another k consecutive customers.

- *3-mode change (3MC)* – every subset of three distinct arcs is evaluated. At each evaluation, we test all $3^3 = 27$ combinations of unused operation modes in the selected arcs. For example, consider three arcs $x, y, z \in A$, with c , ch , and e operation modes, respectively. In such case, the tests consist in evaluating every combination of the modes: ch, e , and b for arc x ; c, e , and b for arc y ; and c, ch , and b for arc z .

In contrast to *3MC*, the remaining neighborhoods do not consider all possible combinations of operation modes in our implementation. This search limitation strategy of restricting the number of possible combinations was based on the characteristics of hybrid electric vehicles. The challenge of the HEVTSPTW is to properly explore the use of the electric motor. Therefore, the modes involving the battery, i.e., ch , e , and b , have to be evaluated often. However, two of them need special attention: the ch mode, as the charging rate is much slower than the discharging one, and the e mode, as it offers the largest travel time. We chose to consider only the ch and the e modes in order to improve the runtime performance of the method. Any drawbacks originated by such approach are likely to be mitigated by the *3MC*, as it focuses the search exclusively on the operation modes.

Figure 4 depicts an example of a reinsertion move in which customer 6 is removed from index 5 and reinserted in between customers 3 and 5. For such move, Figure 4a shows the initial solution configuration (before reinsertion) and Figures 4b–4i show the restricted combinations of operation modes on the newly created arcs. In particular, as only two operation modes are considered for each arc, $2^3 = 8$ distinct arc combinations are checked instead of $4^3 = 64$. Table 3 contains the total and restricted number of combinations verified for a single move of each neighborhood structure, respectively.

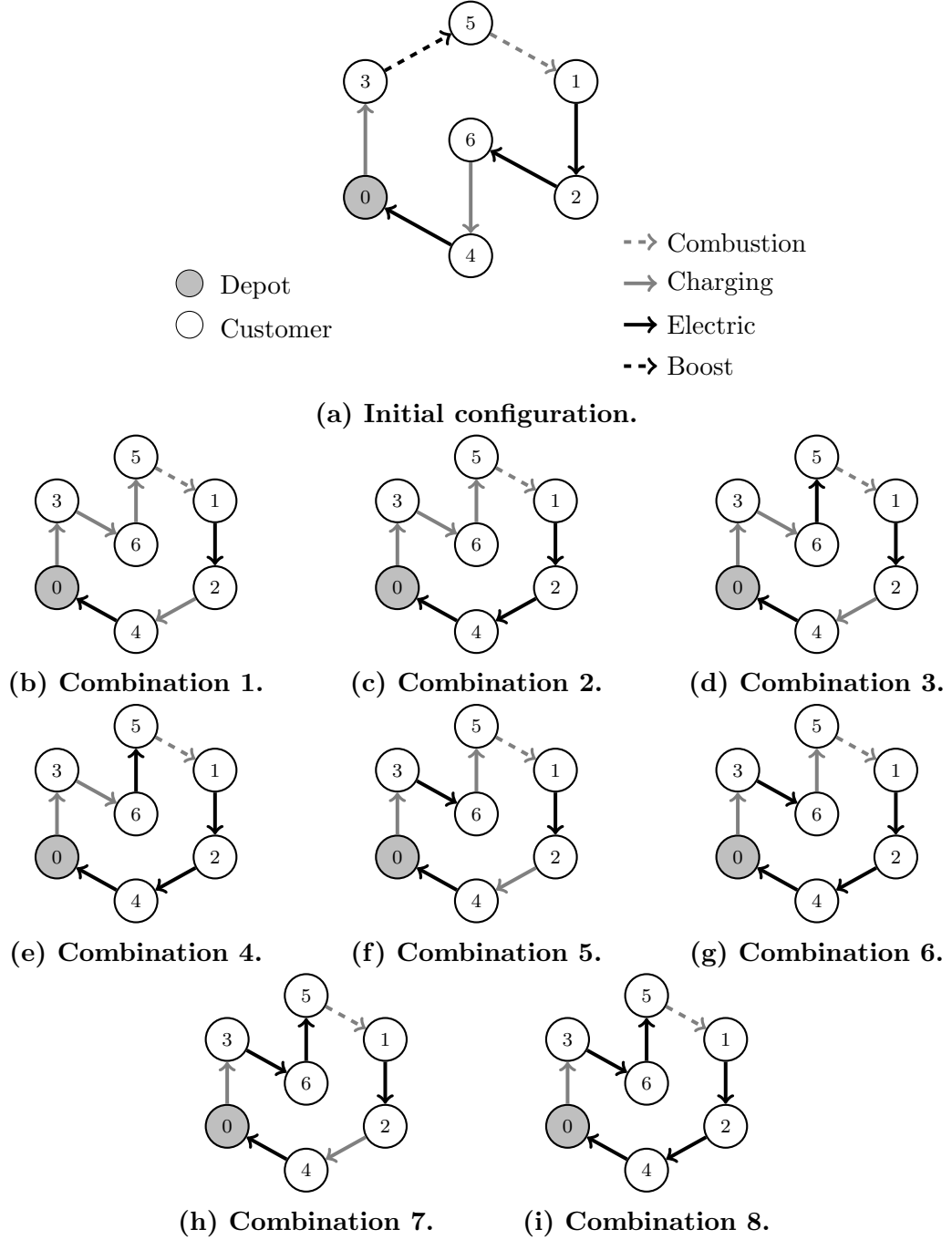


Figura 4: Reinsertion of customer 6 with evaluation of the operation modes of the new arcs.

The pseudocode of the proposed HRVND procedure is presented in Algorithm 2. The first step consists of applying the $3MC$ neighborhood to the individual S , storing the resulting solution on S' (line 2). Next, the loop extending from lines 3 to 7 tries to achieve improvements by means of the traditional *randomized variable neighborhood descent* (RVND) scheme (line 4), introduced in Subramanian et al. [2010], using the remaining neighborhoods, which in turn are less computationally costly. When no improvement is reached (line 5), solution S is returned (line 6), otherwise, the $3MC$ neighborhood is ap-

Tabela 3: Combinations of operation modes evaluated with respect to the neighborhood structures.

Neighborhood	#Arcs involved in the move	All #combinations	Restricted #combinations
<i>Reinsertion</i>	3	$4^3 = 64$	$2^3 = 8$
<i>Or-opt-k</i>	3	$4^3 = 64$	$2^3 = 8$
<i>2-opt</i>	2	$4^2 = 16$	$2^2 = 4$
<i>Exchange</i>	4	$4^4 = 256$	$2^4 = 16$
<i>Swap-k</i>	4	$4^4 = 256$	$2^4 = 16$

plied (line 4). The reason for adopting this hierarchical approach is to limit the number of calls to the $3MC$ neighborhood, clearly the most costly one, with a view of reducing the total CPU time of the method.

Algorithm 2 HRVND

```

1: procedure HRVND( $S$ )
2:    $S' \leftarrow 3MC(S)$ 
3:   while true do
4:      $S \leftarrow RVND(S')$ 
5:     if  $f(S') = f(S)$  then
6:       return  $S$ 
7:    $S' \leftarrow 3MC(S)$ 

```

4.9 Move evaluation

As infeasible solutions are allowed in the HGS_{HEV} algorithm, a mechanism is required to efficiently quantify the route duration, time-warp, and battery violations. All of them rely on preprocessed data for every consecutive subsequence associated with the current solution. Regarding the first two types, one could directly apply the procedures described in Vidal et al. [2013] to evaluate these violations in amortized $\mathcal{O}(1)$ time. Concerning the latter type, there are other works that suggested efficient move evaluation schemes, such as those by Schneider et al. [2014] and Hiermann et al. [2016]. While the former authors devised their own approach to perform move evaluations in $\mathcal{O}(1)$ operations, the latter somewhat merged the procedures implemented in Schneider et al. [2014], Vidal et al. [2013]. However, the data structures proposed in those papers were specifically designed for EVRPs with recharging stations in which a vehicle is completely recharged at one of such stations. Also, they addressed problems whose vehicles have only one operation mode that, in turn, always discharges the battery, as opposed to our problem. Goeke and Schneider [2015] considered a heterogeneous fleet EVRP where the recharging rate is not a linear function of the distance traveled and it actually depends, among other things, on the vehicle mass. They have extended the structures by Schneider et al. [2014]

to cope with such dependency on the cargo load by proposing a so-called surrogate battery violation to avoid keeping track of the real battery violation, which is computationally expensive, thus allowing them to perform move evaluations in constant time. Finally, Hiermann et al. [2019] dealt with a heterogeneous fleet variant considering partial recharging in which the vehicles are either (i) pure electric, (ii) ICE (pure combustion), or (iii) PHEVs with two engines (electric motor and ICE) that can be switched at any point of the route. Due to the complexity of the problem, the authors applied move evaluations on sequences of customer visits combining (i) labeling techniques to compute the cost of inserting recharging stations, and (ii) greedy policies adapting the ideas presented in Hiermann et al. [2016] to decide on charging levels of the battery and the selection of propulsion modes. To avoid systematic calls to computationally expensive procedures, they make use of lower bounding approaches [Vidal, 2017] computed in $\mathcal{O}(1)$ operations by considering pure electric and PHEVs as ICE vehicles.

In contrast to the aforementioned problems, the HEVTSPTW assumes that the vehicle has 4 operation modes, with one recharging option and different discharging possibilities. Also, it does not consider recharging stations nor it imposes the battery to always be completely recharged. In addition, the operations modes can only be switched in the vertices. Although it is seemingly possible to adapt the structures proposed in previous works (e.g., Schneider et al. [2014], Hiermann et al. [2016]) to our specific case, we decided, for the sake of simplicity, to investigate the possibility of minimally extending the scheme by Vidal et al. [2013] for the VRPTW to address battery violations in the HEVTSPTW. Interestingly enough, we found a correspondence between the rationale used to compute the time-window infeasibilities and the charging violations, in a way that the same structures can be adapted to evaluate the battery infeasibility, as described in the following.

Let $\sigma = (\sigma_0, \dots, \sigma_{|\sigma|-1})$ be a subsequence of vertices, and let $\sigma_{ij} = (\sigma_i, \dots, \sigma_j)$ be a subsequence starting at the i -th position and ending at the j -th position. We compute and store the following information regarding the battery for every possible subsequence σ of consecutive vertices of a given solution (original sequence): $l(\sigma)$ minimum battery level, $v(\sigma)$ minimum battery violation, and minimum $s(\sigma)$ and maximum $g(\sigma)$ battery levels allowing a subsequence of vertices with the minimum battery violation.

Such data is straightforward to compute for the trivial case, i.e. a subsequence containing only a single customer σ_0 , that is, $l(\sigma_0) = 0$, $v(\sigma_0) = 0$, $s(\sigma_0) = -l_{max}$, $g(\sigma_0) = 0$. If such customer is the starting depot, then $s(\sigma_0) = 0$, which is based on the assumption that the initial battery level is 0, as imposed by the benchmark instances. Equations (5)–(8) are used to iteratively compute the same data for the remaining subsequences. Let $\sigma = (\sigma_i, \dots, \sigma_j)$ and $\sigma' = (\sigma'_i, \dots, \sigma'_j)$ be two subsequences of visits. We use the following

data to compute the concatenated subsequence $\sigma \oplus \sigma'$:

$$l(\sigma \oplus \sigma') = l(\sigma) + l(\sigma') + \Delta_l + \Delta_e \quad (5)$$

$$v(\sigma \oplus \sigma') = v(\sigma) + v(\sigma') + \Delta_v \quad (6)$$

$$s(\sigma \oplus \sigma') = \max\{s(\sigma') - \Delta, s(\sigma)\} - \Delta_e \quad (7)$$

$$g(\sigma \oplus \sigma') = \min\{g(\sigma') - \Delta, g(\sigma)\} + \Delta_v \quad (8)$$

where: $\Delta_l = -t_{\sigma_j \sigma'_i} r_{\sigma_j}$, $\Delta = l(\sigma) - v(\sigma) + \Delta_l$, $\Delta_e = \max\{s(\sigma') - \Delta - g(\sigma), 0\}$, and $\Delta_v = \max\{s(\sigma) + \Delta - g(\sigma'), 0\}$. A more detailed explanation on the computation of the above data through an example is provided in Subsection 4.10.

The structures above have a direct correspondence with those by Vidal et al. [2013], meaning that the same proof used to demonstrate the validity of their equations can also be applied to show that Equations (5)–(8) are valid. In particular, battery level corresponds to duration, minimum violation to time-warp use, and minimum and maximum battery levels allowing a sequence of visits with the minimum violation to earliest and latest visits, respectively. However, we remark that the computation of Δ_l is rather distinct from the original expression because in this case we are dealing with variations in battery charging levels ($-t_{\sigma_j \sigma'_i} r_{\sigma_j}$) instead of time ($t_{\sigma_j \sigma'_i}$). Moreover, the data structures can also be used with instances with different characteristics (e.g. a different initial battery level), as long as the necessary changes to the parameters in the trivial case are provided. Finally, because of the 2-opt neighborhood structure, we also compute the aforementioned data for reverse subsequences.

4.10 Concatenation of subsequences

Table 4 contains a reduced version of the 8-customer benchmark instance HEVTSP_1_08_1_TW0, obtained by removing its last two customers. Information regarding distances, costs, and time windows are not shown since they are irrelevant in this context. The time, which is given in minutes, is presented according to the selected operation mode.

The individual S of Figure 5 illustrates an example of a possible solution for such 6-customer instance. All subsequences associated with S are then depicted in Figure 6. Furthermore, let $\sigma = (\sigma_i, \dots, \sigma_j)$ and $\sigma' = (\sigma'_i, \dots, \sigma'_j)$ be two subsequences of visits. For the sake of convenience, we reintroduce the following data used to compute the concatenated

Tabela 4: Time and battery specifications of the 6-customer instance.

Initial Charging: 0								
Max Charging (in Watt/h): 16800								
Charging Rate (in Watt/h): 12000								
Discharging Rate (in Watt/h): 48000								
Time for combustion mode								
(Depot-Depot, Depot-1, Depot-..., Depot-6, 1-Depot, 1-1, ..., 6-6):								
0.0	7.21	9.47	9.8	11.92	4.27	11.96	10.17	14.63
7.21	0.0	7.17	5.12	8.48	4.23	9.36	7.85	10.67
9.47	7.17	0.0	12.27	6.82	5.53	2.54	1.51	11.77
9.8	5.12	12.27	0.0	12.04	8.67	11.88	10.7	12.14
11.92	8.48	6.82	12.04	0.0	10.18	6.42	6.72	5.25
4.27	4.23	5.53	8.67	10.18	0.0	8.07	6.27	11.41
11.96	9.36	2.54	11.88	6.42	8.07	0.0	2.54	11.67
10.17	7.85	1.51	10.7	6.72	6.27	2.54	0.0	11.79
14.63	10.67	11.77	12.14	5.25	11.41	11.67	11.79	0.0
Time for charging mode								
(same as time for combustion mode):								
0.0	7.87	10.33	10.69	13.3	4.65	13.04	11.09	16.32
7.87	0.0	7.82	5.58	9.25	4.61	10.21	8.56	11.64
10.33	7.82	0.0	13.38	7.44	6.03	2.77	1.64	12.83
10.69	5.58	13.38	0.0	13.14	9.46	13.26	11.94	13.24
13.3	9.25	7.44	13.14	0.0	11.11	7.01	7.33	5.73
4.65	4.61	6.03	9.46	11.11	0.0	8.81	6.84	12.72
13.04	10.21	2.77	13.26	7.01	8.81	0.0	2.7	12.73
11.09	8.56	1.64	11.94	7.33	6.84	2.7	0.0	12.86
16.32	11.64	12.83	13.24	5.73	12.72	12.73	12.86	0.0
Time for electric mode								
(same as time for combustion mode):								
0.0	8.66	11.36	11.76	15.03	5.12	14.35	12.2	18.45
8.66	0.0	8.6	6.14	10.17	5.07	11.23	9.42	12.81
11.36	8.6	0.0	14.72	8.18	6.64	3.05	1.81	14.12
11.76	6.14	14.72	0.0	14.45	10.41	14.98	13.49	14.56
15.03	10.17	8.18	14.45	0.0	12.22	7.71	8.06	6.3
5.12	5.07	6.64	10.41	12.22	0.0	9.69	7.53	14.38
14.35	11.23	3.05	14.98	7.71	9.69	0.0	2.88	14.0
12.2	9.42	1.81	13.49	8.06	7.53	2.88	0.0	14.15
18.45	12.81	14.12	14.56	6.3	14.38	14.0	14.15	0.0
Time for boost mode								
(same as time for combustion mode):								
0.0	6.66	8.74	9.04	10.17	3.94	11.04	9.39	12.48
6.66	0.0	6.62	4.72	7.82	3.9	8.64	7.24	9.85
8.74	6.62	0.0	11.32	6.3	5.1	2.35	1.39	10.86
9.04	4.72	11.32	0.0	11.12	8.01	10.14	9.13	11.2
10.17	7.82	6.3	11.12	0.0	9.4	5.93	6.2	4.85
3.94	3.9	5.1	8.01	9.4	0.0	7.45	5.79	9.73
11.04	8.64	2.35	10.14	5.93	7.45	0.0	2.4	10.77
9.39	7.24	1.39	9.13	6.2	5.79	2.4	0.0	10.89
12.48	9.85	10.86	11.2	4.85	9.73	10.77	10.89	0.0

subsequence $\sigma \oplus \sigma'$:

$$l(\sigma \oplus \sigma') = l(\sigma) + l(\sigma') + \Delta_l + \Delta_e \quad (9)$$

$$v(\sigma \oplus \sigma') = v(\sigma) + v(\sigma') + \Delta_v \quad (10)$$

$$s(\sigma \oplus \sigma') = \max\{s(\sigma') - \Delta, s(\sigma)\} - \Delta_e \quad (11)$$

$$g(\sigma \oplus \sigma') = \min\{g(\sigma') - \Delta, g(\sigma)\} + \Delta_v \quad (12)$$

where: $\Delta_l = -t_{\sigma_j \sigma'_i} r_{\sigma_j}$, $\Delta = l(\sigma) - v(\sigma) + \Delta_l$, $\Delta_e = \max\{s(\sigma') - \Delta - g(\sigma), 0\}$, and $\Delta_v = \max\{s(\sigma) + \Delta - g(\sigma'), 0\}$.

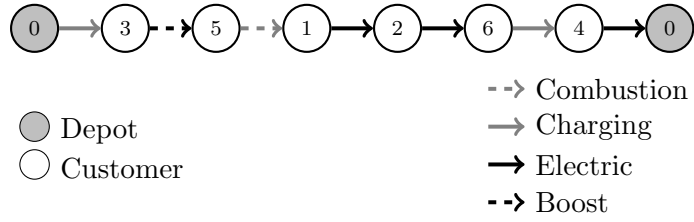


Figure 5: Solution S .

Table 5 shows the results of Equations (9)–(12) for the subsequences of Figure 6, taking into account the time matrix of Table 4 and its battery information. It is important to note that we need to divide the battery charging and discharging rates by 60 in order to properly multiply them by the time for traversing an arc.

We use a reinsertion move to illustrate the concatenation of subsequences, which consists in reinserting customer 6 in between customers 3 and 5, as depicted in Figure 7. For simplicity, the charging operation mode is used on the three newly connected arcs. The concatenation is performed using the preprocessed data of Table 5.

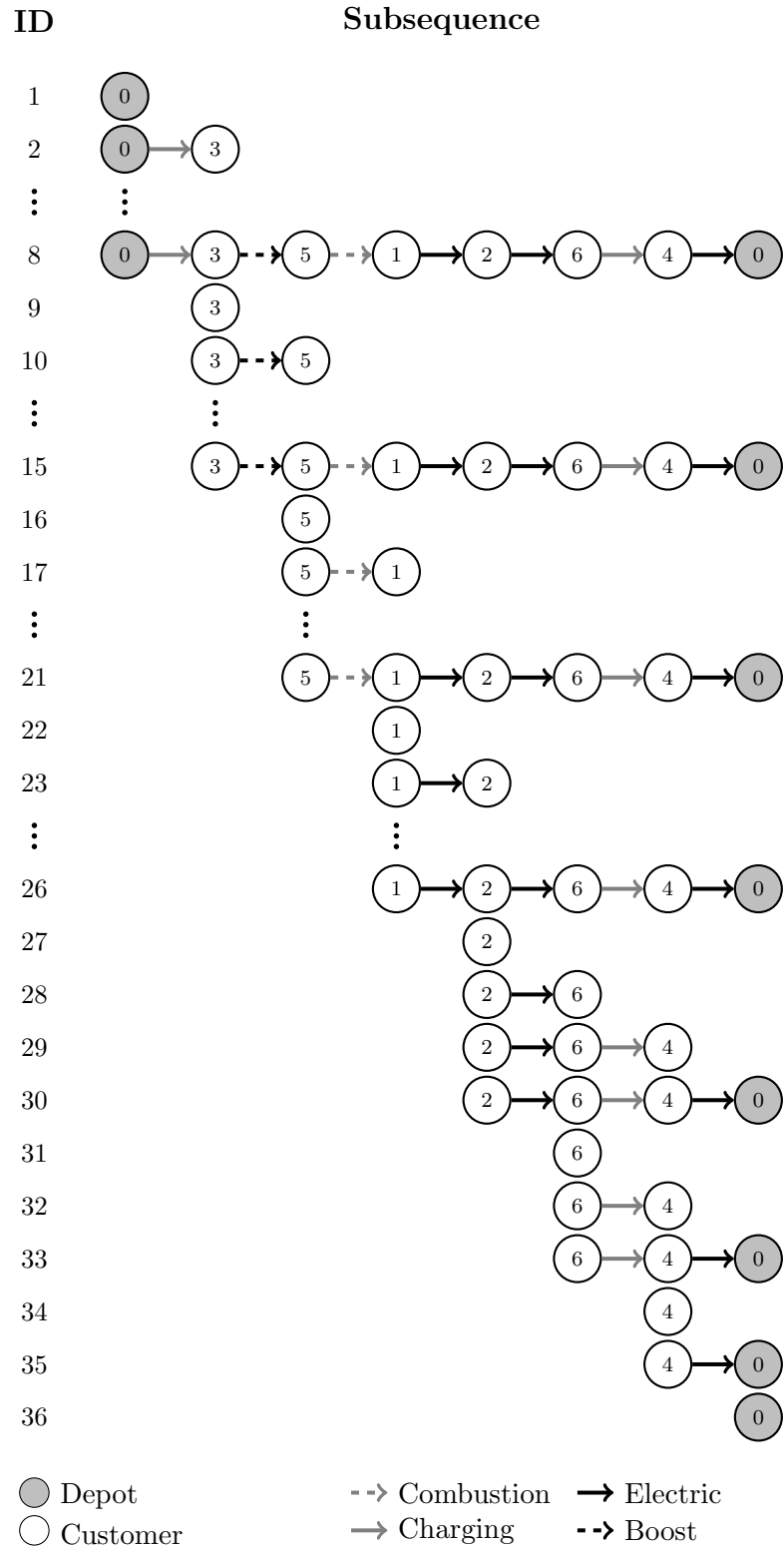


Figura 6: All subsequences associated with a 6-customer solution.

Tabela 5: Values of Equations (9)–(12) for every subsequence of the 6-customer solution.

ID	σ_{ij}	$l(\sigma_{ij})$		$v(\sigma_{ij})$		$s(\sigma_{ij})$		$g(\sigma_{ij})$	
1	σ_{00}	0.0	(0.0%)	0.0	(0.0%)	0.0	(0.0%)	0.0	(0.0%)
2	σ_{01}	-2138.0	(-12.7%)	0.0	(0.0%)	0.0	(0.0%)	0.0	(0.0%)
3	σ_{02}	4270.0	(25.4%)	4270.0	(25.4%)	0.0	(0.0%)	0.0	(0.0%)
4	σ_{03}	4270.0	(25.4%)	4270.0	(25.4%)	0.0	(0.0%)	0.0	(0.0%)
5	σ_{04}	11150.0	(66.4%)	11150.0	(66.4%)	0.0	(0.0%)	0.0	(0.0%)
6	σ_{05}	13590.0	(80.9%)	13590.0	(80.9%)	0.0	(0.0%)	0.0	(0.0%)
7	σ_{06}	12188.0	(72.5%)	13590.0	(80.9%)	0.0	(0.0%)	0.0	(0.0%)
8	σ_{07}	24212.0	(144.1%)	24212.0	(144.1%)	0.0	(0.0%)	0.0	(0.0%)
9	σ_{11}	0.0	(0.0%)	0.0	(0.0%)	-16800.0	(-100.0%)	0.0	(0.0%)
10	σ_{12}	6408.0	(38.1%)	0.0	(0.0%)	-16800.0	(-100.0%)	-6408.0	(-38.1%)
11	σ_{13}	6408.0	(38.1%)	0.0	(0.0%)	-16800.0	(-100.0%)	-6408.0	(-38.1%)
12	σ_{14}	13288.0	(79.1%)	0.0	(0.0%)	-16800.0	(-100.0%)	-13288.0	(-79.1%)
13	σ_{15}	15728.0	(93.6%)	0.0	(0.0%)	-16800.0	(-100.0%)	-15728.0	(-93.6%)
14	σ_{16}	14326.0	(85.3%)	0.0	(0.0%)	-16800.0	(-100.0%)	-15728.0	(-93.6%)
15	σ_{17}	26350.0	(156.8%)	9550.0	(56.8%)	-16800.0	(-100.0%)	-16800.0	(-100.0%)
16	σ_{22}	0.0	(0.0%)	0.0	(0.0%)	-16800.0	(-100.0%)	0.0	(0.0%)
17	σ_{23}	0.0	(0.0%)	0.0	(0.0%)	-16800.0	(-100.0%)	0.0	(0.0%)
18	σ_{24}	6880.0	(41.0%)	0.0	(0.0%)	-16800.0	(-100.0%)	-6880.0	(-41.0%)
19	σ_{25}	9320.0	(55.5%)	0.0	(0.0%)	-16800.0	(-100.0%)	-9320.0	(-55.5%)
20	σ_{26}	7918.0	(47.1%)	0.0	(0.0%)	-16800.0	(-100.0%)	-9320.0	(-55.5%)
21	σ_{27}	19942.0	(118.7%)	3142.0	(18.7%)	-16800.0	(-100.0%)	-16800.0	(-100.0%)
22	σ_{33}	0.0	(0.0%)	0.0	(0.0%)	-16800.0	(-100.0%)	0.0	(0.0%)
23	σ_{34}	6880.0	(41.0%)	0.0	(0.0%)	-16800.0	(-100.0%)	-6880.0	(-41.0%)
24	σ_{35}	9320.0	(55.5%)	0.0	(0.0%)	-16800.0	(-100.0%)	-9320.0	(-55.5%)
25	σ_{36}	7918.0	(47.1%)	0.0	(0.0%)	-16800.0	(-100.0%)	-9320.0	(-55.5%)
26	σ_{37}	19942.0	(118.7%)	3142.0	(18.7%)	-16800.0	(-100.0%)	-16800.0	(-100.0%)
27	σ_{44}	0.0	(0.0%)	0.0	(0.0%)	-16800.0	(-100.0%)	0.0	(0.0%)
28	σ_{45}	2440.0	(14.5%)	0.0	(0.0%)	-16800.0	(-100.0%)	-2440.0	(-14.5%)
29	σ_{46}	1038.0	(6.2%)	0.0	(0.0%)	-16800.0	(-100.0%)	-2440.0	(-14.5%)
30	σ_{47}	13062.0	(77.8%)	0.0	(0.0%)	-16800.0	(-100.0%)	-13062.0	(-77.8%)
31	σ_{55}	0.0	(0.0%)	0.0	(0.0%)	-16800.0	(-100.0%)	0.0	(0.0%)
32	σ_{56}	-1402.0	(-8.3%)	0.0	(0.0%)	-15398.0	(-91.7%)	0.0	(0.0%)
33	σ_{57}	10622.0	(63.2%)	0.0	(0.0%)	-15398.0	(-91.7%)	-10622.0	(-63.2%)
34	σ_{66}	0.0	(0.0%)	0.0	(0.0%)	-16800.0	(-100.0%)	0.0	(0.0%)
35	σ_{67}	12024.0	(71.6%)	0.0	(0.0%)	-16800.0	(-100.0%)	-12024.0	(-71.6%)
36	σ_{77}	0.0	(0.0%)	0.0	(0.0%)	-16800.0	(-100.0%)	0.0	(0.0%)

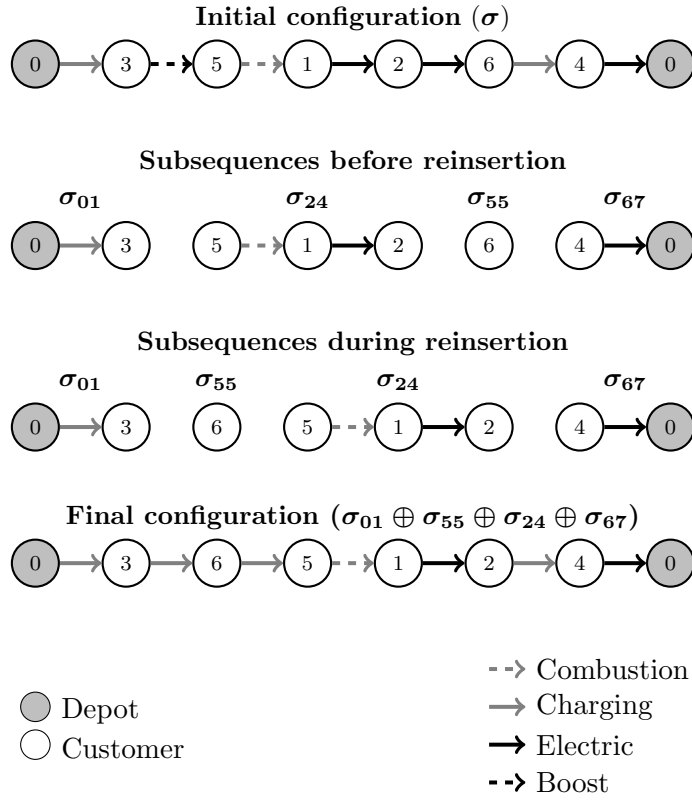


Figura 7: Illustration of a reinsertion move regarding subsequences.

Tabela 6: Computation of Equations (9)–(12) for concatenations of subsequences associated with a reinsertion move.

Concatenated subsequences*	Parameter	Expression	Result
$\sigma_{01} \oplus \sigma_{55}$	Δ_l	$-\delta_{\sigma_1 \sigma_5} r_{\sigma_1}$	$-200.0 \times 13.26 = -2652.0$
	Δ	$l(\sigma_{01}) - v(\sigma_{01}) + \Delta_l$	$-2138.0 - 0.0 + (-2652.0) = -4790.0$
	Δ_e	$\max\{s(\sigma_{55}) - \Delta - g(\sigma_{01}), 0\}$	$\max\{-16800.0 - (-4790.0) - 0.0, 0\} = 0.0$
	Δ_v	$\max\{s(\sigma_{01}) + \Delta - g(\sigma_{55}), 0\}$	$\max\{0.0 + (-4790.0) - 0.0, 0\} = 0.0$
	$l(\sigma_{01} \oplus \sigma_{55})$	$l(\sigma_{01}) + l(\sigma_{55}) + \Delta_l + \Delta_e$	$-2138.0 + 0.0 + (-2652.0) + 0.0 = -4790.0$
	$v(\sigma_{01} \oplus \sigma_{55})$	$v(\sigma_{01}) + v(\sigma_{55}) + \Delta_v$	$0.0 + 0.0 + 0.0 = 0.0$
	$s(\sigma_{01} \oplus \sigma_{55})$	$\max\{s(\sigma_{55}) - \Delta, s(\sigma_{01})\}$	$\max\{-16800.0 - (-4790.0), 0.0\} - 0.0 = 0.0$
	$g(\sigma_{01} \oplus \sigma_{55})$	$\min\{g(\sigma_{55}) - \Delta, g(\sigma_{01})\} + \Delta_v$	$\min\{0.0 - (-4790.0), 0.0\} + 0.0 = 0.0$
$\sigma' \oplus \sigma_{24}$	Δ_l	$-\delta_{\sigma_5 \sigma_2} r_{\sigma_5}$	$-200.0 \times 8.81 = -1762.0$
	Δ	$l(\sigma') - v(\sigma') + \Delta_l$	$-4790.0 - 0.0 + (-1762.0) = -6552.0$
	Δ_e	$\max\{s(\sigma_{24}) - \Delta - g(\sigma'), 0\}$	$\max\{-16800.0 - (-6552.0) - 0.0, 0\} = 0.0$
	Δ_v	$\max\{s(\sigma') + \Delta - g(\sigma_{24}), 0\}$	$\max\{0.0 + (-6552.0) - (-6880.0), 0\} = 328.0$
	$l(\sigma' \oplus \sigma_{24})$	$l(\sigma') + l(\sigma_{24}) + \Delta_l + \Delta_e$	$-4790.0 + 6880.0 + (-1762.0) + 0.0 = 328.0$
	$v(\sigma' \oplus \sigma_{24})$	$v(\sigma') + v(\sigma_{24}) + \Delta_v$	$0.0 + 0.0 + 328.0 = 328.0$
	$s(\sigma' \oplus \sigma_{24})$	$\max\{s(\sigma_{24}) - \Delta, s(\sigma')\}$	$\max\{-16800.0 - (-6552.0), 0.0\} - 0.0 = 0.0$
	$g(\sigma' \oplus \sigma_{24})$	$\min\{g(\sigma_{24}) - \Delta, g(\sigma')\} + \Delta_v$	$\min\{-6880.0 - (-6552.0), 0.0\} + 328.0 = 0.0$
$\sigma'' \oplus \sigma_{67}$	Δ_l	$-\delta_{\sigma_4 \sigma_6} r_{\sigma_4}$	$-200.0 \times 7.44 = -1488.0$
	Δ	$l(\sigma'') - v(\sigma'') + \Delta_l$	$328.0 - 328.0 + (-1488.0) = -1488.0$
	Δ_e	$\max\{s(\sigma_{67}) - \Delta - g(\sigma''), 0\}$	$\max\{-16800.0 - (-1488.0) - 0.0, 0\} = 0.0$
	Δ_v	$\max\{s(\sigma'') + \Delta - g(\sigma_{67}), 0\}$	$\max\{0.0 + (-1488.0) - (-12024.0), 0\} = 10536.0$
	$l(\sigma'' \oplus \sigma_{67})$	$l(\sigma'') + l(\sigma_{67}) + \Delta_l + \Delta_e$	$328.0 + 12024.0 + (-1488.0) + 0.0 = 10864.0$
	$v(\sigma'' \oplus \sigma_{67})$	$v(\sigma'') + v(\sigma_{67}) + \Delta_v$	$328.0 + 0.0 + 10536.0 = \mathbf{10864.0}$
	$s(\sigma'' \oplus \sigma_{67})$	$\max\{s(\sigma_{67}) - \Delta, s(\sigma'')\}$	$\max\{-16800.0 - (-1488.0), 0.0\} - 0.0 = 0.0$
	$g(\sigma'' \oplus \sigma_{67})$	$\min\{g(\sigma_{67}) - \Delta, g(\sigma'')\} + \Delta_v$	$\min\{-12024.0 - (-1488.0), 0.0\} + 10536.0 = 0.0$

* $\sigma' = \sigma_{01} \oplus \sigma_{55}$, $\sigma'' = \sigma' \oplus \sigma_{24}$

Table 6 contains the computation performed in the reinsertion. Firstly, subsequences σ_{01} and σ_{55} are concatenated. Secondly, the resulting data is used to concatenate the newly created subsequence with σ_{24} . Finally, the remaining of the individual is concatenated, concluding the reinsertion procedure. By performing such move, the violation is reduced from 24212.0 to 10864.0 (see σ_{07} in Table 5).

5 Computational experiments

The proposed algorithm was coded in C++, and all experiments were executed on a single thread of an Intel(R) Core(TM) i7-3770 CPU 3.40 GHz, with 8 GB of RAM running Linux Ubuntu 16.04. We decided to run HGS 10 times for each instance following a common practice adopted by many renowned authors (see, e.g., Cordeau and Maischberger [2012], Vidal et al. [2014], Erdoğan [2017], Hellsten et al. [2020], and Keskin et al. [2021]).

A set of benchmark instances was proposed in Doppstadt et al. [2020] and solved by means of a so-called PVNS algorithm considering two versions, *runtime optimized* (RO) and *quality optimized* (QO), with a focus on the runtime and quality performances, respectively. The benchmark dataset is publicly available at <https://data.mendeley.com/datasets/9j3tt84hyx/1>. The instances are divided into 4 groups according to the number of customers, namely 8, 10, 20 and 50. Each group contains 54 instances.

On the other hand, the PVNS approach was executed only 5 times for each instance (possibly because of the large computational times reported for each run) on multiple threads (4 threads for the set RO and 8 threads for the set QO) of an Intel(R) Core(TM) i7-4790 CPU 3.6 GHz, with 24 GB of RAM, running Windows 10. Moreover, according to the site <https://www.cpubenchmark.net/compare/Intel-i7-3770-vs-Intel-i7-4790/896vs2226>, the CPU used to run the PVNS is approximately 1.07 times faster than the one used to execute the HGS algorithm.

Finally, in order to check whether there are significant statistical differences between the average gaps found during parameter tuning and also when comparing HGS with RO, and HGS with QO, we have conducted a non-parametric significance test, more specifically, the Wilcoxon signed-rank test, after statistically confirming that the differences between the average gap values do not seem to be normally distributed. The tests were always conducted separately for each different group of instances.

5.1 Determining the parameter values, neighborhood structures and search strategy

We selected the 108 most challenging instances, among the 216 of the current benchmark, to perform the parameter tuning. The selected instances appear to be more difficult due to the density of their time windows.

The routine for building the initial population is executed $\theta = 4\mu$ times (the same value adopted in Vidal et al. [2012]). Regarding parameter α , at first, we tried to randomly select it from the set $\{0.1, 0.2, 0.3, 0.4, 0.5\}$, as in Bulhões et al. [2018]. However, the results obtained were sometimes not favorable, especially when using 0.1 and 0.2 on

smaller instances, because the method tends to become completely greedy in practice. Therefore, we chose to adopt the set $\{0.3, 0.4, 0.5\}$.

With respect to the parameters used in the subpopulations management, we adopted in all experiments the same values as Mecler et al. [2021]: $\mu = 20$, $\lambda = 40$, $\mu_{elite} = 10$, and $\mu_{close} = 3$. The remaining parameters were calibrated as discussed in the following.

With the purpose of tuning the remaining parameters of the HGS_{HEV} algorithm, we started with a baseline configuration and incrementally adjusted each parameter, as can be observed in the next subsections. For this configuration, $N_{it} = n$, the violation coefficients ω^d , ω^{tw} , and ω^c were set to 100, and the search strategy was best improvement. The values of the violation coefficients focus the search on the space of feasible solutions. This is in principle a desired behavior as only a few iterations are performed.

With respect to the impact of the neighborhoods, we conducted experiments with block-based settings, and with the 2-opt structure. Such settings are specified as follows.

- $\Pi^{(1)}$ – *reinsertion* and *exchange* neighborhoods.
- $\Pi^{(k)}$ – *or-opt-k* and *swap-k*, $k = \{2, 3, 4, 5\}$, with each k associated with a neighborhood.
- Π_{2-opt} – *2-opt* neighborhood.

Table 7 reports the average gap and CPU time for five combinations of settings. The gaps (%) are computed as follows: $gap = 100 \times \{[avg(HGS_{HEV}) - best(PVNS)]/best(PVNS)\}$. It is important to note that for small instances, e.g. the ones with 8 and 10 customers, the larger the value of k , the larger the number of infeasible solutions, considering only the employment of individual settings instead of combinations of them. Moreover, we decided not to report the results of applying individual settings due to the number of infeasible solutions found.

Tabela 7: Results obtained for five different settings of neighborhood structures.

Setting	Combination	Avg. Gap (%)	Avg. CPU (s)
1	$\Pi^{(1)} + \Pi_{2-opt}$	0.224	0.92
2	$\Pi^{(1)} + \Pi^{(2)} + \Pi_{2-opt}$	0.135	0.86
3	$\Pi^{(1)} + \Pi^{(2)} + \Pi^{(3)} + \Pi_{2-opt}$	0.134	0.81
4	$\Pi^{(1)} + \Pi^{(2)} + \Pi^{(3)} + \Pi^{(4)} + \Pi_{2-opt}$	0.118	0.77
5	$\Pi^{(1)} + \Pi^{(2)} + \Pi^{(3)} + \Pi^{(4)} + \Pi^{(5)} + \Pi_{2-opt}$	0.125	0.76

We chose the fourth setting of Table 7 (a combination of settings $\Pi^{(1)}$, $\Pi^{(2)}$, $\Pi^{(3)}$, $\Pi^{(4)}$, and Π_{2-opt}), which yielded a good compromise between average gap and CPU time.

In fact, the Wilcoxon signed-rank test pointed out a statistical difference between the average gaps of settings 3 and 4 for the 10- and the 50- customer instances (p -values of 0.0463 and 0.0436, respectively), but no statistical differences between settings 4 and 5 (p -value > 0.2 for all 4 groups of instances).

Furthermore, we verified the impact of different number of iterations of the main loop of the algorithm, together with two different search strategies: (i) best improvement, which consists of evaluating all possible moves of a neighborhood in search for the best one; and (ii) first improvement, which aims at searching for the first improving move. Hereafter, the parameters discussed in Subsection 4.1 are applied for the violation coefficients as enough iterations are performed to also allow for better exploration of infeasible solutions. Three distinct values were then tested for N_{it} , namely: $\max(12n, 600)$, $\max(20n, 600)$, and $\max(30n, 600)$. For each of those values, four different settings were employed regarding the search strategy. Furthermore, the reason for the increase in the number of iterations is due to the use of the first improvement strategy on some settings, requiring more iterations to converge. Figure 8 depicts the results found using the different settings described in Table 8. Note that the reason for testing a different search strategy on the $3MC$ was due to the fact that this neighborhood structure deals exclusively with the operation modes of the arcs.

Tabela 8: Search strategy settings.

Setting	Search strategy			
	3-mc		remaining moves	
	first impr.	best impr.	first impr.	best impr.
A	✓		✓	
B	✓			✓
C		✓	✓	
D		✓		✓

As Figure 8 suggests, settings A and B converge faster, even though the gaps are worse in comparison to those of settings C and D. Such fact can be explained by the employment of the best improvement strategy on the $3MC$ neighborhood. Moreover, setting C is the only which was dominated in all situations, regardless of the values of N_{it} . Setting D, with $N_{it} = N_{it}^{(1)} = \max(12n, 600)$, was selected as it offers an interesting compromise between solution quality and CPU time. To ratify our choice, we performed a statistical analysis on the differences between the average gaps of the non-dominated settings of Figure 8, namely settings D and B ($N_{it}^{(1)}$, for both), and settings D ($N_{it}^{(1)}$) and B ($N_{it}^{(2)}$). The Wilcoxon signed-rank test indicated a difference on the average gaps, more specifically, for the 20- and the 50-customer instances (p -values < 0.001), in both comparisons.

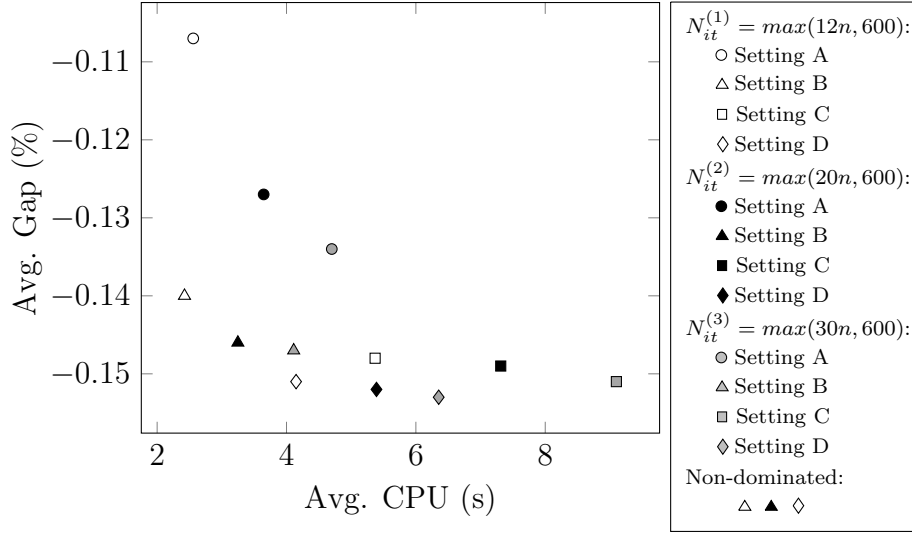


Figure 8: Impacts of the number of iterations and the search strategy.

5.2 Comparison with the literature

Tables 9, 10, 11, and 12 present the results obtained by the HGS_{HEV} on the HEVTSPTW benchmark instances, which are compared with those found by two versions of PVNS: RO and QO. Hereafter, all experiments were conducted considering all instances of the current benchmark. For each instance and method, we report the cost of the best solution, the average cost and the average CPU time. We also present the gap between the average solution and the best known solution (BKS). Gaps of improved solutions are highlighted in bold.

Each instance is composed of four attributes: a distance, in kilometers, from the depot to the delivery area (**D**); the number of customers (**C**); a number of time windows (**TWs**); and for a combination of **D**, **C**, and **TWs**, variations in service times and customer locations (**V**). Some **TWs** have a “+” symbol to indicate a pair of customers with very late time windows. In addition, the delivery area covers approximately 25 km².

Tables 9 and 10 show the results achieved on the instances containing 8 and 10 customers, respectively. All algorithms managed to find the optimal solutions, according to the values found by CPLEX and reported in Doppstadt et al. [2020]. However, HGS_{HEV} attained better CPU times especially in comparison with the QO version of PVNS.

The results obtained on the 20-customer instances are presented in Table 11. It can be observed that the HGS_{HEV} was capable of finding highly competitive results in terms of solution cost and with an average runtime 4.69 and 144.79 times faster than those of versions RO and QO of PVNS, respectively. Furthermore, two new improved solutions were found.

Table 12 illustrates the results found on the 50-customer instances. Overall,

Tabela 9: Results obtained on the 8-customer instances.

Instance				P-VNS _{RO}				P-VNS _{QO}				HGS _{HEV}			
				Best	Avg.	Gap (%)	CPU (s)	Best	Avg.	Gap (%)	CPU (s)	Best	Avg.	Gap (%)	CPU (s)
0	8	0	0	1830.69	1830.69	0.00	0.10	1830.69	1830.69	0.00	1.80	1830.69	1830.69	0.00	0.08
0	8	2	0	1849.37	1849.37	0.00	0.10	1849.37	1849.37	0.00	1.80	1849.37	1849.37	0.00	0.07
0	8	6	0	1849.37	1849.37	0.00	0.10	1849.37	1849.37	0.00	2.00	1849.37	1849.37	0.00	0.06
0	8	2+	0	1849.37	1849.37	0.00	0.10	1849.37	1849.37	0.00	2.10	1849.37	1849.37	0.00	0.07
0	8	4+	0	1849.37	1849.37	0.00	0.10	1849.37	1849.37	0.00	2.10	1849.37	1849.37	0.00	0.07
0	8	8+	0	1849.37	1849.37	0.00	0.10	1849.37	1849.37	0.00	2.10	1849.37	1849.37	0.00	0.07
0	8	0	1	1553.15	1553.15	0.00	0.10	1553.15	1553.15	0.00	1.70	1553.15	1553.15	0.00	0.07
0	8	2	1	1553.15	1553.15	0.00	0.10	1553.15	1553.15	0.00	1.80	1553.15	1553.15	0.00	0.08
0	8	6	1	1553.15	1553.15	0.00	0.10	1553.15	1553.15	0.00	2.00	1553.15	1553.15	0.00	0.07
0	8	2+	1	1553.15	1553.15	0.00	0.10	1553.15	1553.15	0.00	1.90	1553.15	1553.15	0.00	0.07
0	8	4+	1	1553.15	1553.15	0.00	0.10	1553.15	1553.15	0.00	2.00	1553.15	1553.15	0.00	0.09
0	8	8+	1	1553.15	1553.15	0.00	0.10	1553.15	1553.15	0.00	1.70	1553.15	1553.15	0.00	0.07
0	8	0	2	1435.74	1435.74	0.00	0.10	1435.74	1435.74	0.00	1.80	1435.74	1435.74	0.00	0.06
0	8	2	2	1445.89	1445.89	0.00	0.10	1436.09	1438.05	0.14	1.80	1436.09	1436.09	0.00	0.08
0	8	6	2	1460.31	1460.31	0.00	0.10	1460.31	1460.31	0.00	1.80	1460.31	1460.31	0.00	0.09
0	8	2+	2	1436.09	1445.78	0.67	0.10	1436.09	1436.09	0.00	2.10	1436.09	1436.09	0.00	0.07
0	8	4+	2	1436.09	1440.93	0.34	0.10	1436.09	1436.09	0.00	2.10	1436.09	1436.09	0.00	0.07
0	8	8+	2	1460.31	1460.31	0.00	0.10	1460.31	1460.31	0.00	1.70	1460.31	1460.31	0.00	0.08
28	8	0	0	7189.85	7189.85	0.00	0.10	7189.85	7189.85	0.00	1.70	7189.85	7189.85	0.00	0.09
28	8	2	0	7189.85	7189.85	0.00	0.10	7189.85	7189.85	0.00	1.70	7189.85	7189.85	0.00	0.09
28	8	6	0	7189.85	7189.85	0.00	0.10	7189.85	7189.85	0.00	2.00	7189.85	7189.85	0.00	0.08
28	8	2+	0	7202.07	7202.07	0.00	0.10	7202.07	7202.07	0.00	2.00	7202.07	7202.07	0.00	0.07
28	8	4+	0	7202.07	7202.07	0.00	0.10	7202.07	7202.07	0.00	2.00	7202.07	7202.07	0.00	0.05
28	8	8+	0	7202.07	7202.07	0.00	0.10	7202.07	7202.07	0.00	2.10	7202.07	7202.07	0.00	0.07
28	8	0	1	7140.95	7140.95	0.00	0.10	7140.95	7140.95	0.00	1.80	7140.95	7140.95	0.00	0.09
28	8	2	1	7174.85	7174.85	0.00	0.10	7174.85	7174.85	0.00	1.80	7174.85	7174.85	0.00	0.10
28	8	6	1	7174.85	7174.85	0.00	0.10	7174.85	7174.85	0.00	2.10	7174.85	7174.85	0.00	0.07
28	8	2+	1	7174.85	7174.85	0.00	0.10	7174.85	7174.85	0.00	1.80	7174.85	7174.85	0.00	0.07
28	8	4+	1	7174.85	7174.85	0.00	0.10	7174.85	7174.85	0.00	1.90	7174.85	7174.85	0.00	0.07
28	8	8+	1	7174.85	7174.85	0.00	0.10	7174.85	7174.85	0.00	2.10	7174.85	7174.85	0.00	0.06
28	8	0	2	7292.25	7292.25	0.00	0.10	7292.25	7292.25	0.00	1.60	7292.25	7292.25	0.00	0.09
28	8	2	2	7292.25	7292.25	0.00	0.10	7292.25	7292.25	0.00	1.80	7292.25	7292.25	0.00	0.07
28	8	6	2	7292.25	7292.25	0.00	0.10	7292.25	7292.25	0.00	1.80	7292.25	7292.25	0.00	0.08
28	8	2+	2	7292.25	7292.25	0.00	0.10	7292.25	7292.25	0.00	1.90	7292.25	7292.25	0.00	0.07
28	8	4+	2	7292.25	7292.25	0.00	0.10	7292.25	7292.25	0.00	1.90	7292.25	7292.25	0.00	0.08
28	8	8+	2	7292.25	7292.25	0.00	0.10	7292.25	7292.25	0.00	1.90	7292.25	7292.25	0.00	0.07
57	8	0	0	12706.60	12708.00	0.01	0.10	12706.60	12706.60	0.00	2.20	12706.60	12706.60	0.00	0.10
57	8	2	0	12706.60	12706.60	0.00	0.10	12706.60	12706.60	0.00	2.30	12706.60	12706.60	0.00	0.10
57	8	6	0	12724.75	12724.75	0.00	0.10	12724.75	12724.75	0.00	2.30	12724.75	12724.75	0.00	0.07
57	8	2+	0	12706.60	12706.60	0.00	0.10	12706.60	12706.60	0.00	2.50	12706.60	12706.60	0.00	0.10
57	8	4+	0	12706.60	12706.60	0.00	0.10	12706.60	12706.60	0.00	2.60	12706.60	12706.60	0.00	0.08
57	8	8+	0	12724.75	12724.75	0.00	0.10	12724.75	12724.75	0.00	2.40	12724.75	12724.75	0.00	0.07
57	8	0	1	12687.90	12687.90	0.00	0.10	12687.90	12687.90	0.00	2.10	12687.90	12687.90	0.00	0.08
57	8	2	1	12712.96	12712.96	0.00	0.10	12712.96	12712.96	0.00	2.10	12712.96	12712.96	0.00	0.07
57	8	6	1	12712.96	12712.96	0.00	0.10	12712.96	12712.96	0.00	2.30	12712.96	12712.96	0.00	0.07
57	8	2+	1	12712.96	12712.96	0.00	0.10	12712.96	12712.96	0.00	2.30	12712.96	12712.96	0.00	0.09
57	8	4+	1	12712.96	12712.96	0.00	0.10	12712.96	12712.96	0.00	2.30	12712.96	12712.96	0.00	0.09
57	8	8+	1	12712.96	12712.96	0.00	0.10	12712.96	12712.96	0.00	2.50	12712.96	12712.96	0.00	0.07
57	8	0	2	12708.57	12708.57	0.00	0.10	12708.57	12708.57	0.00	1.90	12708.57	12708.57	0.00	0.08
57	8	2	2	12708.57	12708.57	0.00	0.10	12708.57	12708.57	0.00	2.00	12708.57	12708.57	0.00	0.07
57	8	6	2	12738.05	12747.31	0.07	0.10	12738.05	12738.05	0.00	2.20	12738.05	12738.05	0.00	0.07
57	8	2+	2	12761.21	12761.21	0.00	0.10	12738.05	12738.05	0.00	2.20	12738.05	12738.05	0.00	0.09
57	8	4+	2	12761.21	12761.21	0.00	0.10	12738.05	12738.05	0.00	2.30	12738.05	12738.05	0.00	0.08
57	8	8+	2	12738.05	12746.12	0.06	0.10	12738.05	12738.05	0.00	2.50	12738.05	12738.05	0.00	0.09
Avg.						0.02	0.10			< 0.01	2.02			0.00	0.08

Tabela 10: Results obtained on the 10-customer instances.

Instance				P-VNS _{RO}				P-VNS _{QO}				HGS _{HEV}			
D	C	TWs	V	Best	Avg.	Gap (%)	CPU (s)	Best	Avg.	Gap (%)	CPU (s)	Best	Avg.	Gap (%)	CPU (s)
0	10	0	0	1798.94	1799.44	0.03	0.20	1798.94	1799.44	0.03	4.50	1798.94	1798.94	0.00	0.11
0	10	3	0	1798.94	1798.94	0.00	0.30	1798.94	1799.95	0.06	4.80	1798.94	1798.94	0.00	0.12
0	10	8	0	1801.46	1801.46	0.00	0.20	1801.46	1801.46	0.00	5.10	1801.46	1801.46	0.00	0.09
0	10	2+	0	1798.94	1798.94	0.00	0.20	1801.46	1801.46	0.00	4.80	1798.94	1798.94	0.00	0.12
0	10	5+	0	1798.94	1798.94	0.00	0.20	1801.46	1801.46	0.00	5.00	1798.94	1798.94	0.00	0.12
0	10	10+	0	1801.46	1801.46	0.00	0.20	1801.46	1801.46	0.00	4.60	1801.46	1801.46	0.00	0.11
0	10	0	1	1598.18	1598.18	0.00	0.30	1598.18	1598.18	0.00	4.60	1598.18	1598.18	0.00	0.13
0	10	3	1	1601.73	1601.73	0.00	0.30	1602.08	1602.08	0.00	4.70	1601.73	1601.73	0.00	0.17
0	10	8	1	1601.73	1601.73	0.00	0.20	1601.73	1601.73	0.00	5.90	1601.73	1601.73	0.00	0.10
0	10	2+	1	1601.73	1601.73	0.00	0.20	1602.08	1602.08	0.00	4.90	1601.73	1601.73	0.00	0.13
0	10	5+	1	1601.73	1601.73	0.00	0.20	1602.08	1602.08	0.00	4.70	1601.73	1601.73	0.00	0.13
0	10	10+	1	1601.73	1601.73	0.00	0.20	1601.73	1601.73	0.00	5.40	1601.73	1601.73	0.00	0.12
0	10	0	2	1478.90	1478.90	0.00	0.20	1478.90	1478.90	0.00	4.30	1478.90	1478.90	0.00	0.10
0	10	3	2	1531.45	1533.67	0.14	0.20	1531.45	1531.45	0.00	4.30	1531.45	1531.45	0.00	0.14
0	10	8	2	1531.45	1534.89	0.22	0.20	1531.45	1531.45	0.00	4.40	1531.45	1531.45	0.00	0.11
0	10	2+	2	1531.45	1531.45	0.00	0.20	1531.45	1531.45	0.00	4.70	1531.45	1531.45	0.00	0.13
0	10	5+	2	1531.45	1533.17	0.11	0.20	1531.45	1531.45	0.00	4.50	1531.45	1531.45	0.00	0.11
0	10	10+	2	1531.45	1532.03	0.04	0.20	1531.45	1531.45	0.00	4.40	1531.45	1531.45	0.00	0.12
28	10	0	0	7308.15	7308.15	0.00	0.20	7308.15	7308.15	0.00	4.10	7308.15	7310.25	0.03	0.12
28	10	3	0	7314.32	7314.32	0.00	0.20	7314.32	7314.32	0.00	4.40	7314.32	7314.32	0.00	0.12
28	10	8	0	7329.18	7329.18	0.00	0.20	7329.18	7329.18	0.00	4.70	7329.18	7329.18	0.00	0.11
28	10	2+	0	7314.32	7314.32	0.00	0.20	7314.32	7314.32	0.00	4.80	7314.32	7314.32	0.00	0.10
28	10	5+	0	7314.32	7314.32	0.00	0.20	7314.32	7314.32	0.00	5.10	7314.32	7314.32	0.00	0.09
28	10	10+	0	7329.18	7329.18	0.00	0.20	7329.18	7329.18	0.00	5.20	7329.18	7329.18	0.00	0.10
28	10	0	1	7362.93	7362.93	0.00	0.20	7362.93	7362.93	0.00	4.10	7362.93	7362.93	0.00	0.14
28	10	3	1	7362.93	7362.93	0.00	0.20	7362.93	7362.93	0.00	4.40	7362.93	7362.93	0.00	0.12
28	10	8	1	7362.93	7362.93	0.00	0.20	7362.93	7362.93	0.00	4.90	7362.93	7362.93	0.00	0.11
28	10	2+	1	7362.93	7362.93	0.00	0.20	7362.93	7362.93	0.00	4.80	7362.93	7362.93	0.00	0.10
28	10	5+	1	7362.93	7362.93	0.00	0.20	7362.93	7362.93	0.00	4.80	7362.93	7362.93	0.00	0.10
28	10	10+	1	7362.93	7362.93	0.00	0.20	7362.93	7362.93	0.00	5.00	7362.93	7362.93	0.00	0.11
28	10	0	2	7290.82	7290.82	0.00	0.20	7290.82	7290.82	0.00	4.10	7290.82	7290.82	0.00	0.16
28	10	3	2	7290.82	7290.82	0.00	0.20	7290.82	7290.82	0.00	4.10	7290.82	7290.82	0.00	0.16
28	10	8	2	7290.82	7290.82	0.00	0.20	7290.82	7290.82	0.00	4.50	7290.82	7290.82	0.00	0.12
28	10	2+	2	7290.82	7290.82	0.00	0.20	7290.82	7290.82	0.00	4.60	7290.82	7290.82	0.00	0.12
28	10	5+	2	7290.82	7290.82	0.00	0.20	7290.82	7290.82	0.00	4.60	7290.82	7290.82	0.00	0.12
28	10	10+	2	7290.82	7290.82	0.00	0.20	7290.82	7290.82	0.00	4.40	7290.82	7290.82	0.00	0.10
57	10	0	0	12747.22	12747.22	0.00	0.20	12747.22	12747.22	0.00	5.20	12747.22	12747.22	0.00	0.14
57	10	3	0	12759.24	12760.75	0.01	0.20	12759.24	12759.24	0.00	5.70	12759.24	12759.24	0.00	0.16
57	10	8	0	12759.24	12759.24	0.00	0.20	12759.24	12759.24	0.00	6.20	12759.24	12759.24	0.00	0.13
57	10	2+	0	12759.24	12759.24	0.00	0.20	12759.24	12759.24	0.00	5.70	12759.24	12759.24	0.00	0.13
57	10	5+	0	12759.24	12759.24	0.00	0.20	12759.24	12759.24	0.00	6.00	12759.24	12759.24	0.00	0.12
57	10	10+	0	12759.24	12759.24	0.00	0.20	12759.24	12759.24	0.00	5.80	12759.24	12759.24	0.00	0.14
57	10	0	1	12725.88	12725.88	0.00	0.20	12725.88	12725.88	0.00	5.10	12725.88	12725.88	0.00	0.13
57	10	3	1	12738.38	12738.38	0.00	0.20	12738.38	12738.38	0.00	5.50	12738.38	12738.38	0.00	0.15
57	10	8	1	12738.38	12738.38	0.00	0.20	12738.38	12738.38	0.00	6.30	12738.38	12738.38	0.00	0.14
57	10	2+	1	12738.38	12738.38	0.00	0.20	12738.38	12738.38	0.00	5.90	12738.38	12738.38	0.00	0.12
57	10	5+	1	12738.38	12738.38	0.00	0.20	12738.38	12738.38	0.00	6.10	12738.38	12738.38	0.00	0.14
57	10	10+	1	12738.38	12738.38	0.00	0.20	12738.38	12738.38	0.00	6.90	12738.38	12738.38	0.00	0.13
57	10	0	2	12935.48	12935.58	0.00	0.20	12935.48	12935.48	0.00	4.90	12935.48	12935.48	0.00	0.15
57	10	3	2	12935.48	12935.48	0.00	0.20	12935.48	12935.48	0.00	5.40	12935.48	12935.55	0.00	0.16
57	10	8	2	12935.48	12935.48	0.00	0.20	12935.48	12935.48	0.00	6.10	12935.48	12935.48	0.00	0.11
57	10	2+	2	12935.48	12935.48	0.00	0.20	12935.48	12935.48	0.00	5.50	12935.48	12935.48	0.00	0.12
57	10	5+	2	12935.48	12935.48	0.00	0.20	12935.48	12935.48	0.00	5.70	12935.48	12935.48	0.00	0.13
57	10	10+	2	12935.48	12935.48	0.00	0.20	12935.48	12935.48	0.00	6.50	12935.48	12935.48	0.00	0.10
Avg.						0.01	0.21			< 0.01	5.05			< 0.01	0.12

Tabela 11: Results obtained on the 20-customer instances.

Instance				P-VNS _{RO}				P-VNS _{QO}				HGS _{HEV}			
D	C	TW _s	V	Best	Avg.	Gap (%)	CPU (s)	Best	Avg.	Gap (%)	CPU (s)	Best	Avg.	Gap (%)	CPU (s)
0	20	0	0	2005.89	2009.63	0.19	4.10	2005.89	2005.89	0.00	96.10	2005.89	2005.89	0.00	0.70
0	20	8	0	2005.89	2005.89	0.00	4.10	2005.89	2005.89	0.00	108.40	2005.89	2005.89	0.00	0.67
0	20	18	0	2005.89	2005.89	0.00	3.80	2005.89	2005.89	0.00	91.00	2005.89	2005.89	0.00	0.58
0	20	2+	0	2005.89	2005.89	0.00	4.10	2005.89	2005.89	0.00	103.30	2005.89	2005.89	0.00	0.71
0	20	10+	0	2005.89	2005.89	0.00	4.00	2005.89	2005.89	0.00	116.90	2005.89	2005.89	0.00	0.64
0	20	20+	0	2005.89	2005.89	0.00	3.70	2005.89	2005.89	0.00	99.50	2005.89	2005.89	0.00	0.62
0	20	0	1	1969.78	1970.77	0.05	4.10	1969.78	1969.80	0.00	100.10	1969.78	1969.78	0.00	0.87
0	20	8	1	1969.78	1969.78	0.00	4.30	1969.78	1969.78	0.00	116.30	1969.78	1969.78	0.00	0.67
0	20	18	1	1969.78	1969.80	0.00	4.20	1969.78	1969.78	0.00	133.10	1969.78	1969.78	0.00	0.81
0	20	2+	1	1969.78	1969.80	0.00	4.10	1969.78	1969.78	0.00	110.00	1969.78	1969.78	0.00	0.67
0	20	10+	1	1969.78	1969.78	0.00	4.10	1969.78	1969.78	0.00	143.70	1969.78	1969.78	0.00	0.63
0	20	20+	1	1969.78	1969.78	0.00	3.00	1969.78	1969.78	0.00	111.80	1969.78	1969.78	0.00	0.65
0	20	0	2	1606.42	1606.42	0.00	4.60	1606.57	1606.57	0.00	93.00	1606.42	1606.42	0.00	0.73
0	20	8	2	1606.42	1606.42	0.00	4.40	1606.42	1606.51	0.01	104.70	1606.42	1606.42	0.00	0.66
0	20	18	2	1608.37	1608.37	0.00	3.80	1608.37	1608.37	0.00	103.30	1608.37	1608.37	0.00	0.61
0	20	2+	2	1606.42	1606.81	0.02	4.20	1606.42	1606.51	0.01	111.70	1606.42	1606.42	0.00	0.59
0	20	10+	2	1606.42	1606.42	0.00	4.30	1606.42	1606.51	0.01	123.00	1606.42	1606.42	0.00	0.78
0	20	20+	2	1608.37	1608.37	0.00	3.50	1608.37	1608.37	0.00	99.00	1608.37	1608.37	0.00	0.54
28	20	0	0	7807.05	7813.40	0.08	3.50	7807.05	7807.25	0.00	75.90	7807.05	7807.05	0.00	0.87
28	20	8	0	7807.05	7807.05	0.00	3.60	7807.05	7807.05	0.00	94.80	7807.05	7807.05	0.00	0.72
28	20	18	0	7807.05	7807.05	0.00	3.10	7807.05	7807.05	0.00	92.50	7807.05	7807.05	0.00	0.71
28	20	2+	0	7807.05	7807.05	0.00	3.60	7807.56	7807.56	0.00	90.60	7807.05	7807.05	0.00	0.81
28	20	10+	0	7807.95	7874.43	0.85	3.70	7807.56	7807.56	0.00	135.10	7807.05	7807.05	-0.01	0.71
28	20	20+	0	7807.05	7807.05	0.00	2.80	7807.05	7807.05	0.00	113.30	7807.05	7807.05	0.00	0.58
28	20	0	1	7672.11	7689.21	0.22	3.70	7670.94	7670.94	0.00	81.10	7670.94	7671.00	0.00	1.42
28	20	8	1	7670.94	7671.44	0.01	3.60	7670.94	7670.94	0.00	89.30	7670.94	7671.13	0.00	1.03
28	20	18	1	7670.94	7670.94	0.00	3.20	7670.94	7670.94	0.00	84.40	7670.94	7671.00	0.00	0.80
28	20	2+	1	7670.94	7678.20	0.09	3.60	7670.94	7670.94	0.00	99.30	7670.94	7671.00	0.00	0.96
28	20	10+	1	7670.94	7670.94	0.00	3.70	7670.94	7670.94	0.00	122.00	7670.94	7671.00	0.00	1.02
28	20	20+	1	7670.94	7670.94	0.00	3.10	7670.94	7670.94	0.00	110.50	7670.94	7671.14	0.00	0.67
28	20	0	2	7717.25	7722.01	0.06	3.40	7713.39	7713.39	0.00	72.50	7709.14	7711.27	-0.03	1.09
28	20	8	2	7715.01	7717.39	0.03	3.30	7709.14	7714.31	0.07	96.30	7709.14	7709.14	0.00	0.71
28	20	18	2	7709.14	7709.14	0.00	3.20	7709.14	7709.14	0.00	104.00	7709.14	7709.73	0.01	0.61
28	20	2+	2	7709.14	7709.14	0.00	3.50	7709.14	7711.08	0.03	88.10	7709.14	7709.63	0.01	0.75
28	20	10+	2	7709.14	7709.14	0.00	3.60	7709.14	7710.76	0.02	130.90	7709.14	7709.14	0.00	0.63
28	20	20+	2	7709.14	7709.14	0.00	3.10	7709.14	7709.14	0.00	119.10	7709.14	7709.14	0.00	0.50
57	20	0	0	13343.32	13343.32	0.00	3.30	13329.82	13329.82	0.00	86.00	13329.82	13329.82	0.00	0.97
57	20	8	0	13343.32	13345.86	0.02	3.50	13343.32	13343.34	0.00	109.60	13343.32	13344.17	0.01	0.92
57	20	18	0	13343.32	13343.32	0.00	3.50	13343.32	13343.32	0.00	132.80	13343.32	13343.32	0.00	0.82
57	20	2+	0	13329.82	13332.52	0.02	3.40	13329.82	13329.82	0.00	107.00	13329.82	13329.82	0.00	0.77
57	20	10+	0	13343.32	13343.32	0.00	3.70	13343.32	13343.32	0.00	144.90	13343.32	13343.32	0.00	0.81
57	20	20+	0	13343.32	13343.32	0.00	3.40	13343.32	13343.32	0.00	161.30	13343.32	13343.32	0.00	0.65
57	20	0	1	13290.27	13299.34	0.07	3.40	13287.61	13287.61	0.00	92.80	13287.61	13289.01	0.01	1.05
57	20	8	1	13287.61	13287.61	0.00	3.60	13287.61	13288.14	0.00	120.70	13287.61	13287.61	0.00	0.85
57	20	18	1	13287.61	13287.61	0.00	3.30	13287.61	13287.61	0.00	122.90	13287.61	13287.61	0.00	0.82
57	20	2+	1	13287.61	13294.01	0.05	3.60	13287.61	13287.61	0.00	111.60	13287.61	13287.61	0.00	0.79
57	20	10+	1	13287.61	13287.61	0.00	3.70	13287.61	13287.61	0.00	159.80	13287.61	13287.61	0.00	0.91
57	20	20+	1	13287.61	13287.61	0.00	3.20	13287.61	13287.61	0.00	170.50	13287.61	13287.61	0.00	0.85
57	20	0	2	13247.33	13247.33	0.00	3.40	13247.33	13247.33	0.00	91.40	13247.33	13250.29	0.02	0.85
57	20	8	2	13247.33	13247.33	0.00	3.20	13247.33	13247.33	0.00	108.40	13247.33	13247.33	0.00	0.82
57	20	18	2	13247.33	13247.33	0.00	3.20	13247.33	13247.33	0.00	134.30	13247.33	13247.33	0.00	0.72
57	20	2+	2	13247.33	13247.59	0.00	3.40	13247.33	13247.33	0.00	102.10	13247.33	13247.33	0.00	0.68
57	20	10+	2	13247.33	13247.33	0.00	3.50	13247.33	13247.33	0.00	143.90	13247.33	13247.33	0.00	0.89
57	20	20+	2	13247.33	13247.33	0.00	2.90	13247.33	13247.33	0.00	156.10	13247.33	13247.33	0.00	0.71
Avg.						0.03	3.61			< 0.01	111.49			< 0.01	0.77

HGS_{HEV} achieved high quality solutions and new improved solutions were found for 26 instances. With respect to the runtime performance, HGS_{HEV} managed to be 9.86 and 331.47 times faster than the versions RO and QO of PVNS, on average, respectively.

Tabela 12: Results obtained on the 50-customer instances.

Instance				P-VNS _{RO}				P-VNS _{QO}				HGS _{HEV}			
D	C	TWs	V	Best	Avg.	Gap (%)	CPU (s)	Best	Avg.	Gap (%)	CPU (s)	Best	Avg.	Gap (%)	CPU (s)
0	50	0	0	3046.44	3071.36	0.82	182.70	2771.98	2819.37	1.71	3848.10	2712.90	2713.65	-2.10	17.12
0	50	23	0	2712.90	2712.90	0.00	169.10	2712.90	2714.44	0.06	5649.00	2712.90	2712.90	0.00	15.25
0	50	48	0	2716.54	2716.54	0.00	158.40	2716.54	2716.54	0.00	4552.60	2716.54	2716.54	0.00	12.00
0	50	2+	0	2843.86	2843.86	0.00	191.60	2761.61	2828.51	2.42	4183.40	2712.90	2712.92	-1.76	20.73
0	50	25+	0	2712.90	2712.90	0.00	198.70	2713.97	2714.53	0.02	11744.30	2712.90	2712.90	0.00	15.57
0	50	50+	0	2716.54	2716.54	0.00	164.80	2716.54	2716.54	0.00	6415.50	2716.54	2716.54	0.00	11.59
0	50	0	1	2727.67	2824.08	3.53	223.20	2574.63	2587.52	0.50	4321.60	2489.24	2489.57	-3.30	19.61
0	50	23	1	2497.43	2497.99	0.02	224.30	2497.78	2508.73	0.44	8163.60	2489.24	2489.24	-0.33	12.84
0	50	48	1	2497.78	2497.78	0.00	200.90	2497.78	2497.78	0.00	5272.10	2497.78	2497.78	0.00	13.05
0	50	2+	1	2750.81	2792.02	1.50	210.10	2626.64	2626.64	0.00	4830.60	2489.24	2490.44	-5.19	20.43
0	50	25+	1	2496.82	2502.71	0.24	224.40	2497.78	2502.45	0.19	12580.20	2489.24	2489.24	-0.30	14.69
0	50	50+	1	2497.78	2497.78	0.00	163.50	2497.78	2497.78	0.00	7449.90	2497.78	2497.78	0.00	13.99
0	50	0	2	2595.56	2623.22	1.07	198.50	2506.76	2506.76	0.00	4117.80	2474.21	2474.80	-1.27	19.44
0	50	23	2	2481.22	2513.84	1.31	206.40	2499.59	2523.86	0.97	7383.30	2474.21	2474.21	-0.28	15.24
0	50	48	2	2497.59	2497.59	0.00	194.10	2497.59	2497.63	0.00	4443.10	2497.59	2497.60	0.00	13.60
0	50	2+	2	2618.41	2618.41	0.00	196.20	2539.65	2656.19	4.59	4765.20	2487.14	2487.35	-2.06	19.20
0	50	25+	2	2505.68	2505.68	0.00	209.20	2531.96	2531.96	0.00	11573.00	2487.14	2487.47	-0.73	16.41
0	50	50+	2	2497.59	2500.92	0.13	136.60	2497.59	2497.86	0.01	7073.70	2497.59	2497.61	0.00	15.09
28	50	0	0	8456.38	8497.22	0.48	102.80	8420.92	8421.19	0.00	1405.40	8338.12	8338.17	-0.98	24.96
28	50	23	0	8387.19	8388.64	0.02	73.70	8363.71	8363.71	0.00	4512.60	8363.71	8363.71	0.00	12.17
28	50	48	0	8363.71	8363.71	0.00	124.60	8363.71	8363.71	0.00	3665.60	8363.71	8363.71	0.00	11.96
28	50	2+	0	8503.34	8535.99	0.38	85.10	8414.32	8418.38	0.05	1797.90	8338.12	8338.12	-0.91	18.58
28	50	25+	0	8387.19	8405.95	0.22	56.40	8386.99	8387.11	0.00	5361.00	8386.99	8386.99	0.00	13.92
28	50	50+	0	8400.11	8400.12	0.00	122.30	8404.00	8404.56	0.01	3607.80	8404.75	8410.26	0.12	18.95
28	50	0	1	8691.60	8691.60	0.00	241.10	8585.05	8586.54	0.02	2963.10	8425.70	8427.87	-1.83	26.09
28	50	23	1	8427.26	8438.68	0.14	153.40	8427.26	8427.26	0.00	5090.30	8425.90	8425.90	-0.02	18.80
28	50	48	1	8427.26	8427.26	0.00	152.60	8425.90	8425.90	0.00	4944.80	8425.90	8425.90	0.00	17.56
28	50	2+	1	8762.09	8762.09	0.00	174.90	8565.78	8576.23	0.12	3648.10	8425.70	8425.70	-1.64	19.93
28	50	25+	1	8427.26	8485.88	0.70	124.30	8427.26	8427.26	0.00	7233.10	8425.90	8425.90	-0.02	17.28
28	50	50+	1	8425.90	8426.17	0.00	145.60	8425.90	8425.90	0.00	7834.00	8425.90	8425.90	0.00	13.19
28	50	0	2	8749.99	8775.47	0.29	162.90	8508.20	8508.20	0.00	2560.30	8422.15	8428.20	-0.94	25.09
28	50	23	2	8446.66	8448.81	0.03	90.70	8428.38	8431.13	0.03	4250.90	8428.38	8428.38	0.00	14.38
28	50	48	2	8428.38	8428.38	0.00	168.60	8428.38	8428.38	0.00	4870.40	8428.38	8428.38	0.00	13.52
28	50	2+	2	8590.87	8590.87	0.00	159.70	8475.79	8497.20	0.25	2890.20	8428.38	8428.38	-0.56	21.62
28	50	25+	2	8437.85	8479.76	0.50	69.10	8428.38	8429.78	0.02	5520.50	8428.38	8428.38	0.00	14.60
28	50	50+	2	8428.38	8428.38	0.00	155.00	8428.38	8428.38	0.00	7255.60	8428.38	8428.38	0.00	11.87
57	50	0	0	14108.54	14108.54	0.00	196.20	13881.72	13942.60	0.44	3263.80	13846.98	13860.74	-0.15	23.21
57	50	23	0	13935.30	13965.74	0.22	175.70	13853.20	13891.55	0.28	5627.70	13846.98	13847.02	-0.04	17.58
57	50	48	0	13846.98	13846.98	0.00	179.70	13846.98	13846.98	0.00	4763.50	13846.98	13847.06	0.00	15.73
57	50	2+	0	14150.30	14150.30	0.00	178.50	13847.37	13907.15	0.43	3978.30	13846.98	13849.67	0.02	18.00
57	50	25+	0	13846.98	13884.15	0.27	212.10	13847.37	13875.43	0.20	9454.10	13846.98	13847.02	0.00	15.62
57	50	50+	0	13846.98	13846.98	0.00	162.70	13846.98	13846.98	0.00	8677.90	13846.98	13847.02	0.00	12.98
57	50	0	1	14135.44	14135.44	0.00	180.70	14043.83	14072.36	0.20	3336.80	13935.63	13946.66	-0.69	24.74
57	50	23	1	13980.33	13984.36	0.03	173.30	13980.34	13980.34	0.00	5815.90	13980.33	13981.33	0.01	15.83
57	50	48	1	13985.22	13985.22	0.00	172.80	13985.22	13985.22	0.00	5224.40	13985.22	13985.22	0.00	16.24
57	50	2+	1	14207.78	14306.90	0.70	148.20	14077.99	14085.20	0.05	3666.30	13985.22	13988.28	-0.64	26.47
57	50	25+	1	13985.22	13985.22	0.00	161.40	13985.22	13985.22	0.00	8367.10	13985.22	13985.22	0.00	14.66
57	50	50+	1	13985.22	13985.22	0.00	162.20	13985.22	13985.22	0.00	8439.60	13985.22	13985.22	0.00	14.28
57	50	0	2	13929.89	13929.89	0.00	188.30	13866.67	13866.67	0.00	3257.60	13805.51	13806.26	-0.44	16.27
57	50	23	2	13875.02	13942.94	0.49	206.20	13821.48	13821.48	0.00	5820.30	13805.96	13805.96	-0.11	16.50
57	50	48	2	13819.39	13819.86	0.00	173.40	13820.18	13820.18	0.00	4637.50	13819.30	13819.30	0.00	14.85
57	50	2+	2	13929.21	13940.95	0.08	183.30	13953.07	13981.12	0.20	4039.30	13805.96	13805.98	-0.88	20.96
57	50	25+	2	13806.14	13864.39	0.42	169.20	13821.48	13821.48	0.00	9018.90	13805.96	13805.96	< 0.00	16.16
57	50	50+	2	13820.18	13820.18	0.00	177.80	13819.30	13819.83	0.00	7868.50	13819.30	13819.30	0.00	13.85
Avg.						0.25	166.99			0.24	5611.78			-0.50	16.93

According to the Wilcoxon signed-rank test, there are no significant differences (p -values > 0.05) for the instances involving 10 customers, as expected, because the

average gaps are zero for the vast majority of the instances. On the other hand, there seems to be a significant difference between HGS and RO, and HGS and QO for the 50-customer instances with p -values < 0.001 in both comparisons. As for the 8- and the 20-customer instances, there was a statistical difference when comparing HGS with RO (p -values < 0.01), but the same did not happen for HGS and QO (p -values > 0.1).

Table 13 summarizes the results found by the HGS_{HEV} in terms of solution quality, when compared with those achieved by the PVNS considering both versions: RO and QO. It can be observed that the gains tend to be more prominent as the size of the instance increases. Note that in many cases our average solution was better than the best known solution attained by PVNS, which ratifies the superior performance of HGS_{HEV} in terms of solution quality.

Tabela 13: Summary of the results found by HGS_{HEV} compared with those of versions RO and QO of PVNS.

	8 customers		10 customers		20 customers		50 customers	
	RO	QO	RO	QO	RO	QO	RO	QO
#Best improved	3	0	0	5	6	4	34	31
#Best equaled	51	54	54	49	48	50	19	22
#Best worse	0	0	0	0	0	0	1	1
#Avg. better than the Best	3	0	0	5	6	4	34	29
#Avg. equal to the Best	51	54	52	47	39	39	13	18
#Avg. improved	8	1	7	7	18	13	38	37
#Avg. equaled	46	53	45	45	30	31	12	13
#Avg. worse	0	0	2	2	6	10	4	4
#Worst better than the Best	3	0	0	5	5	3	34	27
#Worst equal to the Best	51	54	52	47	39	40	13	19
#Worst better than the Avg.	8	1	7	7	15	11	37	36
#Worst equal to the Avg.	46	53	45	45	30	32	12	13

5.3 Impact of the newly proposed offspring generation procedure

In order to demonstrate the effectiveness of the newly proposed offspring generation operators, i.e., MOX and MOXMC, we have also executed the HGS_{HEV} algorithm with the following crossover operators: *order crossover* (OX) [Oliver et al., 1987]; *OX with mode changes* (OXMC); *modified order crossover* (MOX); *sequential constructive crossover* (SCX), described in [Ahmed, 2010]; *SCX with mode changes* (SCXMC); *edge recombination crossover* (ERX) [Whitley et al., 1989]; and *ERX with mode changes* (ERXMC).

For the OX, MOX, SCX and ERX, the modes chromosome of one of the parents is randomly selected and copied to the child. In the case of OXMC, whenever a vertex is copied from one parent to the offspring, the operation mode connecting such vertex to the next one is also copied, regardless of the next customer served in the offspring solution. As for the SCXMC and the ERXMC, the modes chromosome at index 0 of the

child is randomly selected from one of the parents' 0 index modes chromosome. In the SCXMC, the remaining of such chromosome is filled according to the selected customer and parent in a similar way as the MOXMC; whereas in the ERXMC, the remaining modes chromosome is filled randomly from one of the parents based on the selected customer.

Figure 9 exhibits the results of employing the discussed operators on the 50-customer instances, which is definitely the most challenging set and the one where the differences become more evident. It can be observed that crossover operators with mode changes tend to offer better solutions in terms of average gap, but with larger runtimes. The operator ERXMC, for example, resulted in a average gap of approximately -0.44% against -0.38% of ERX, but with an average CPU time approximately 3 seconds worse. Moreover, the figure clearly shows the superiority of both MOX and MOXMC operators, which dominate the remaining ones. While the former improved the runtime performance of HGS_{HEV}, the latter helped the proposed algorithm to consistently obtain better average solutions, especially on larger instances, as summarized in Table 14. Moreover, the Wilcoxon signed-rank test confirmed that there is a statistical difference between the average gaps achieved by MOX and MOXMC for the 8-, 20- and 50-customer instances, with p -values < 0.001 in these three groups. For this reason, it was thought advisable to adopt MOXMC as the crossover operator.

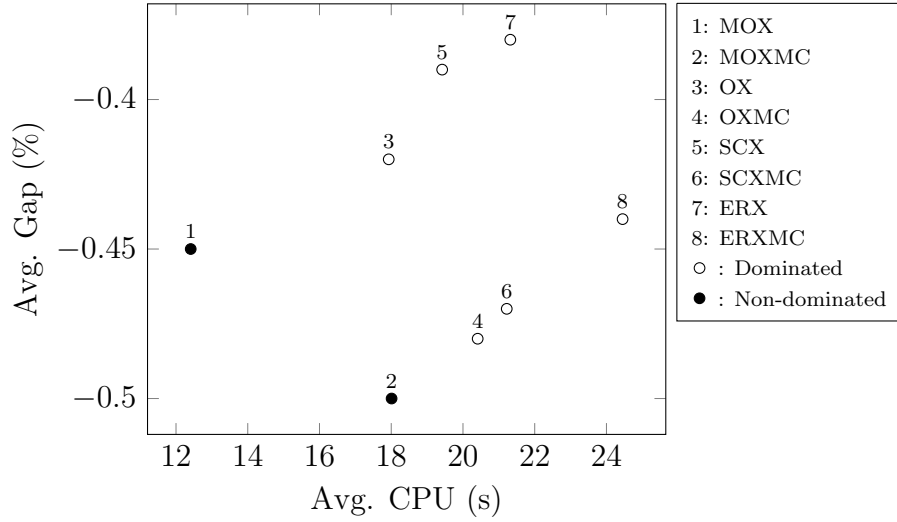


Figure 9: Impact of different crossover operators.

Tabela 14: Number of cases in which the average solution of one operator was better than the other.

	8 customers	10 customers	20 customers	50 customers
MOX	0	0	3	5
MOXMC	14	5	32	43

5.4 Impact of the efficient move evaluation scheme

Table 15 shows the speed-up achieved by employing the proposed move evaluation scheme compared to a straightforward implementation. The speed-up is obtained even for small instances (8 customers), and it becomes increasingly significant with the instance size. Therefore, one can conclude that the proposed approach plays a crucial role on the runtime performance of our algorithm, most notably on larger instances.

Tabela 15: Speed-up achieved by using the efficient move evaluation scheme.

Instance size	Avg. CPU (s)		Speed-up
	Efficient Implementation	Straightforward Implementation	
08	0.08	0.12	1.54
10	0.12	0.20	1.60
20	0.77	1.46	1.90
50	16.93	54.13	3.20

5.5 Impact of restricting the number of arc combinations during local search

Table 16 presents the impact on solution quality and CPU time of restricting the number of arc combinations during local search. The results suggest that a significant speed-up can be attained without loss on the average gaps.

Tabela 16: Impact of restricting the number of arc combinations during local search.

Instance size	Avg. CPU (s)		Speed-up	Avg. Gap (%)	
	Restricted Combinations	All Combinations		Restricted Combinations	All Combinations
08	0.08	0.43	5.59	0.00	0.01
10	0.12	0.75	6.06	< 0.01	< 0.01
20	0.77	3.08	4.00	< 0.00	< 0.00
50	16.93	48.56	2.87	-0.50	-0.49

5.6 Impact of the instance size on the operation modes

As the final step of the numerical experiments, we conducted an analysis on the impact of the number of customers on the selected operation modes of the vehicle. The best solution was considered on this analysis. In addition, the percentage usage of each operation mode was computed for each instance and then the average percentage value was computed based on 4 groups created according to the sizes of the instances.

Figure 10 depicts the results obtained per group. The reason for the missing information about the boost operation mode is because no arc was selected with such mode in any of the best solutions found by HGS_{HEV} . As can be observed, there is an inverse relationship between the usage of the combustion mode c and the size of the instances, that is, as the number of customers increases, the number of arcs traversed in such mode decreases. There is also a direct relationship between the usage of the charging mode ch and the instance size, that is, as one increases the other one also increases. The electric mode e usage, on the other hand, is not significantly changed as the number of customers increases. These observations convey better usage of the electric characteristics of the vehicle as the sizes of the instances increase for the benchmark under consideration.

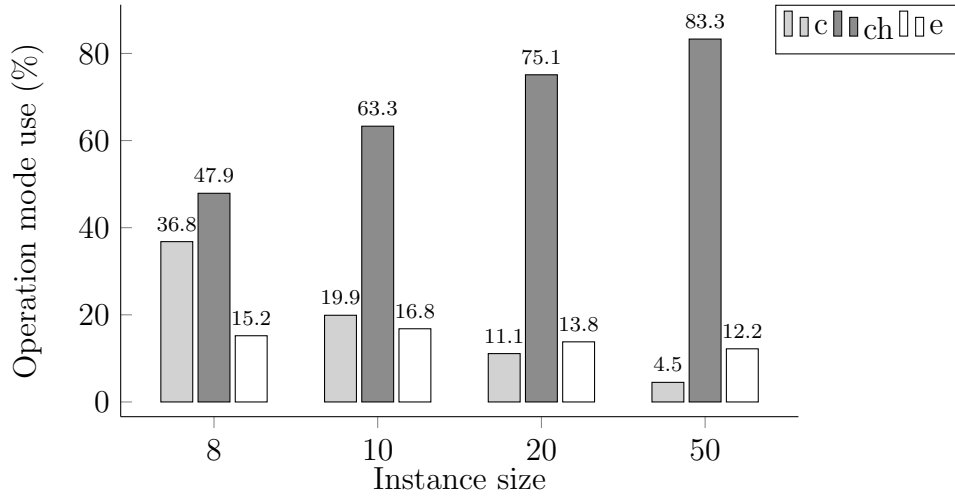


Figura 10: Relation between instance length and the operation mode use.

Moreover, Table 17 shows the average length of the arcs (of the best solutions), in meters, per operation mode and instance size. According to the table, as the size of the instances increases, the average length of the arcs with the charging mode decreases with respect to the average length of the arcs with the combustion and electric modes. For this reason, more arcs have to be used with the charging mode in order to balance the electric mode usage. This fact explains the behavior reported in Figure 10.

Tabela 17: Average length of the arcs, in meters, per operation mode and instance size.

Operation mode	#Customers			
	8	10	20	50
c	15842.20	20067.57	22543.46	21769.84
ch	3633.45	2502.55	930.66	481.96
e	1612.75	1637.42	1376.10	1116.47
ch/c	0.23	0.12	0.04	0.02
ch/e	2.25	1.53	0.68	0.43

6 Concluding remarks and future work

This work presented a *hybrid genetic search* (HGS) algorithm to solve the *traveling salesman problem with time windows* (HEVTSPW). The proposed method includes a *hierarchical randomized variable neighborhood descent* (HRVND) procedure in the local search phase, a limitation strategy to prevent all combinations of arcs associated with the battery operation modes from being inspected, and a novel move evaluation scheme to efficiently compute battery violations in $\mathcal{O}(1)$ time. We also introduced a *modified order crossover procedure with mode changes* (MOXMC) that takes advantage of the battery operation modes of the problem.

Extensive computational experiments were carried out on benchmark instances to show the gains of each of the main components of our HGS, as well as to assess the performance of the method itself. The results ratified the importance of such components and also clearly showed that the proposed HGS clearly outperformed the best known solution approach, both in terms of solution quality and CPU time. All best known solutions were achieved or improved and the difference between the average gaps increased in favor of HGS when compared to the best existing method. Furthermore, we conducted an analysis on the proportional usage of each battery operation mode, and it was observed that the larger the instance size the larger the charging mode tends to be employed and the lesser the combustion mode is likely to be activated.

Promising avenues of research include the extension of the proposed HGS algorithm to solve the version with multiple vehicles and additional attributes such as vehicle capacity and multiple depots. Furthermore, one could also consider scenarios involving uncertainties (e.g., on travel time) in order to produce solutions that are more reliable to be adopted in practice. Such uncertainties can be tackled both under the lenses of stochastic programming or robust optimization, thus requiring the development of further procedures to efficiently cope with the challenges arising from these characteristics.

Bibliography

- Abdelkader Sbihi and Richard W Eglese. Combinatorial optimization and green logistics. *4OR*, 5(2):99–116, 2007.
- Zhu Zhongming, Lu Linong, Zhang Wangqiang, Liu Wei, et al. Trends and projections in Europe 2020 tracking progress towards europe’s climate and energy targets. Technical Report 13/2020A, European Environment Agency (EEA), 2020.
- Yoshinori Suzuki. A new truck-routing approach for reducing fuel consumption and pollutants emission. *Transportation Research Part D: Transport and Environment*, 16(1):73–77, 2011.
- Canhong Lin, King Lun Choy, George TS Ho, Sai Ho Chung, and HY Lam. Survey of green vehicle routing problem: past and future trends. *Expert systems with applications*, 41(4):1118–1138, 2014.
- Simona Mancini. The hybrid vehicle routing problem. *Transportation Research Part C: Emerging Technologies*, 78:1–12, 2017.
- Erfan Ghorbani, Mahdi Alinaghian, Gevork B Gharehpetian, Sajad Mohammadi, and Guido Perboli. A survey on environmentally friendly vehicle routing problem and a proposal of its classification. *Sustainability*, 12(21):9079, 2020.
- ARADEx AG. Retrofit electric drive kit for diesel delivery vehicles, 2021. URL <https://aradex.asia/en/system-solutions/aradex-project-report/light-vehicle-s-small-vans/retrofit-electric-drive-kit-for-diesel-delivery-vehicles/>.
- Christian Doppstadt, Achim Koberstein, and Daniele Vigo. The hybrid electric vehicle—traveling salesman problem with time windows. *European Journal of Operational Research*, 284(2):675–692, 2020.
- Thibaut Vidal, Teodor Gabriel Crainic, Michel Gendreau, and Christian Prins. A unified solution framework for multi-attribute vehicle routing problems. *European Journal of Operational Research*, 234(3):658–673, 2014.
- Tomislav Erdelić and Tonči Carić. A survey on the electric vehicle routing problem: variants and solution approaches. *Journal of Advanced Transportation*, 2019, 2019.
- Hu Qin, Xinxin Su, Teng Ren, and Zhixing Luo. A review on the electric vehicle routing problems: Variants and algorithms. *Frontiers of Engineering Management*, 8(3):370–389, 2021.
- Sevgi Erdoğan and Elise Miller-Hooks. A green vehicle routing problem. *Transportation research part E: logistics and transportation review*, 48(1):100–114, 2012.

- Jane Lin, Wei Zhou, and Ouri Wolfson. Electric vehicle routing problem. *Transportation Research Procedia*, 12:508–521, 2016.
- Michael Schneider, Andreas Stenger, and Dominik Goeke. The electric vehicle-routing problem with time windows and recharging stations. *Transportation science*, 48(4): 500–520, 2014.
- Maurizio Bruglieri, Ferdinando Pezzella, Ornella Pisacane, and Stefano Suraci. A variable neighborhood search branching for the electric vehicle routing problem with time windows. *Electronic Notes in Discrete Mathematics*, 47:221–228, 2015.
- Pierre Hansen, Nenad Mladenović, Jack Brimberg, and José A Moreno Pérez. Variable neighborhood search. In *Handbook of metaheuristics*, pages 57–97. Springer, 2019.
- Angelo Sifaleras and Ioannis Konstantaras. A survey on variable neighborhood search methods for supply network inventory. In *International Conference on Network Analysis*, pages 71–82. Springer, 2020.
- Shaowen Lan, Wenjuan Fan, Shanlin Yang, Panos M Pardalos, and Nenad Mladenovic. A survey on the applications of variable neighborhood search algorithm in healthcare management. *Annals of Mathematics and Artificial Intelligence*, 89(8):741–775, 2021.
- Manuel Laguna. Tabu search. In *Handbook of heuristics*, pages 741–758. Springer, 2018.
- Pierre Hansen, Nenad Mladenović, and Dragan Urošević. Variable neighborhood search and local branching. *Computers & Operations Research*, 33(10):3034–3045, 2006.
- Matteo Fischetti and Andrea Lodi. Local branching. *Mathematical programming*, 98(1): 23–47, 2003.
- Guy Desaulniers, Fausto Errico, Stefan Irnich, and Michael Schneider. Exact algorithms for electric vehicle-routing problems with time windows. *Operations Research*, 64(6): 1388–1405, 2016.
- Merve Keskin and Bülent Çatay. Partial recharge strategies for the electric vehicle routing problem with time windows. *Transportation Research Part C: Emerging Technologies*, 65:111–127, 2016.
- David Pisinger and Stefan Ropke. Large neighborhood search. In *Handbook of metaheuristics*, pages 99–127. Springer, 2019.
- Alejandro Montoya, Christelle Guéret, Jorge E Mendoza, and Juan G Villegas. The electric vehicle routing problem with nonlinear charging function. *Transportation Research Part B: Methodological*, 103:87–110, 2017.

- Helena Ramalhinho Lourenço, Olivier C. Martin, and Thomas Stützle. *Iterated Local Search: Framework and Applications*, chapter 5, pages 129–168. Springer International Publishing, Cham, 2019.
- Kenneth Earl Rosing and CS ReVelle. Heuristic concentration: Two stage solution construction. *European Journal of Operational Research*, 97(1):75–86, 1997.
- Gerhard Hiermann, Jakob Puchinger, Stefan Ropke, and Richard F Hartl. The electric fleet size and mix vehicle routing problem with time windows and recharging stations. *European Journal of Operational Research*, 252(3):995–1018, 2016.
- F Yu Vincent, AAN Perwira Redi, Yosi Agustina Hidayat, and Oktaviyanto Jimat Wibowo. A simulated annealing heuristic for the hybrid vehicle routing problem. *Applied Soft Computing*, 53:119–132, 2017.
- Daniel Delahaye, Supatcha Chaimatanan, and Marcel Mongeau. Simulated annealing: From basics to applications. In *Handbook of metaheuristics*, pages 1–35. Springer, 2019.
- Xiaohui Li, Xuemin Shi, Yi Zhao, Huagang Liang, and Yuan Dong. Svnd enhanced metaheuristic for plug-in hybrid electric vehicle routing problem. *Applied Sciences*, 10(2):441, 2020.
- Keisuke Murakami. Formulation and algorithms for route planning problem of plug-in hybrid electric vehicles. *Operational Research*, 18(2):497–519, 2018.
- Sina Bahrami, Mehdi Nourinejad, Glareh Amirjamshidi, and Matthew J Roorda. The plugin hybrid electric vehicle routing problem: A power-management strategy model. *Transportation Research Part C: Emerging Technologies*, 111:318–333, 2020.
- Gerhard Hiermann, Richard F Hartl, Jakob Puchinger, and Thibaut Vidal. Routing a mix of conventional, plug-in hybrid, and electric vehicles. *European Journal of Operational Research*, 272(1):235–248, 2019.
- Busra Gulnihan Dascioglu and Gulfem Tuzkaya. A literature review for hybrid vehicle routing problem. *Industrial Engineering in the Big Data Era*, pages 249–257, 2019.
- Amira Belhadj Ammar, Taicir Moalla Loukil, and Mohamed Cheikh. A survey on the hybrid vehicle routing problem: Literature review. In *2022 14th International Colloquium of Logistics and Supply Chain Management (LOGISTIQUA)*, pages 1–6. IEEE, 2022.
- Lu Zhen, Ziheng Xu, Chengle Ma, and Liyang Xiao. Hybrid electric vehicle routing problem with mode selection. *International Journal of Production Research*, 58(2):562–576, 2020.

- Majid Seyfi, Mahdi Alinaghian, Erfan Ghorbani, Bülent Çatay, and Mohammad Saeid Sabbagh. Multi-mode hybrid electric vehicle routing problem. *Transportation Research Part E: Logistics and Transportation Review*, 166:102882, 2022.
- Riccardo Poli, James Kennedy, and Tim Blackwell. Particle swarm optimization. *Swarm intelligence*, 1(1):33–57, 2007.
- Chi-Kin Chau, Khaled Elbassioni, and Chien-Ming Tseng. Fuel minimization of plug-in hybrid electric vehicles by optimizing drive mode selection. In *Proceedings of the Seventh International Conference on Future Energy Systems*, pages 1–11, 2016.
- Christian Doppstadt, Achim Koberstein, and Daniele Vigo. The hybrid electric vehicle–traveling salesman problem. *European Journal of Operational Research*, 253(3):825–842, 2016.
- Thibaut Vidal, Teodor Gabriel Crainic, Michel Gendreau, Nadia Lahrichi, and Walter Rei. A hybrid genetic algorithm for multidepot and periodic vehicle routing problems. *Operations Research*, 60(3):611–624, 2012.
- Thomas A Feo and Mauricio G C Resende. Greedy randomized adaptive search procedures. *Journal of global optimization*, 6(2):109–133, 1995.
- Thibaut Vidal, Teodor Gabriel Crainic, Michel Gendreau, and Christian Prins. A hybrid genetic algorithm with adaptive diversity management for a large class of vehicle routing problems with time-windows. *Computers & operations research*, 40(1):475–489, 2013.
- Jordana Mecler, Anand Subramanian, and Thibaut Vidal. A simple and effective hybrid genetic search for the job sequencing and tool switching problem. *Computers & Operations Research*, 127:105153, 2021.
- IM Oliver, DJd Smith, and John RC Holland. Study of permutation crossover operators on the traveling salesman problem. In *Genetic algorithms and their applications: proceedings of the second International Conference on Genetic Algorithms: July 28-31, 1987 at the Massachusetts Institute of Technology, Cambridge, MA*. Hillsdale, NJ: L. Erlbaum Associates, 1987., 1987.
- Anand Subramanian, Lúcia Maria de A Drummond, Cristiana Bentes, Luiz Satoru Ochi, and Ricardo Farias. A parallel heuristic for the vehicle routing problem with simultaneous pickup and delivery. *Computers & Operations Research*, 37(11):1899–1911, 2010.
- Dominik Goeke and Michael Schneider. Routing a mixed fleet of electric and conventional vehicles. *European Journal of Operational Research*, 245(1):81–99, 2015.

- Thibaut Vidal. Node, edge, arc routing and turn penalties: Multiple problems—one neighborhood extension. *Operations Research*, 65(4):992–1010, 2017.
- Jean-François Cordeau and Mirko Maischberger. A parallel iterated tabu search heuristic for vehicle routing problems. *Computers & Operations Research*, 39(9):2033–2050, 2012.
- Güneş Erdoğan. An open source spreadsheet solver for vehicle routing problems. *Computers & operations research*, 84:62–72, 2017.
- Erik Orm Hellsten, David Sacramento, and David Pisinger. An adaptive large neighbourhood search heuristic for routing and scheduling feeder vessels in multi-terminal ports. *European Journal of Operational Research*, 287(2):682–698, 2020.
- Merve Keskin, Bülent Çatay, and Gilbert Laporte. A simulation-based heuristic for the electric vehicle routing problem with time windows and stochastic waiting times at recharging stations. *Computers & Operations Research*, 125:105060, 2021.
- Teobaldo Bulhões, Anand Subramanian, Güneş Erdoğan, and Gilbert Laporte. The static bike relocation problem with multiple vehicles and visits. *European Journal of Operational Research*, 264(2):508–523, 2018.
- Zakir H Ahmed. Genetic algorithm for the traveling salesman problem using sequential constructive crossover operator. *International Journal of Biometrics & Bioinformatics (IJBB)*, 3(6):96, 2010.
- L Darrell Whitley, Timothy Starkweather, and D’Ann Fuquay. Scheduling problems and traveling salesmen: The genetic edge recombination operator. In *ICGA*, volume 89, pages 133–40, 1989.