Computers & Operations Research 40 (2013) 475–489

Contents lists available at SciVerse ScienceDirect

Computers & Operations Research

journal homepage: www.elsevier.com/locate/caor

A hybrid genetic algorithm with adaptive diversity management for a large class of vehicle routing problems with time-windows

Thibaut Vidal [a,b,*], Teodor Gabriel Crainic [c], Michel Gendreau [d], Christian Prins [a]

[a] Institut Charles Delaunay - LOSI, Université de Technologie de Troyes, 12 rue Marie Curie, BP 2060, 10010 Troyes Cedex, France
[b] CIRRELT et Département d'informatique et de recherche opérationnelle, Université de Montréal, Montréal, Canada H3C 3J7
[c] CIRRELT et Département de management et technologie, École des sciences de la gestion, UQAM, Montréal, Canada H3C 3P8
[d] CIRRELT et Département de mathématiques et génie industriel, École Polytechnique, Montréal, Canada H3C 3A7

ARTICLE INFO

Available online 31 July 2012

Keywords:
Vehicle routing problems
Time windows
Hybrid genetic algorithm
Diversity management
Neighbourhood search
Decomposition

ABSTRACT

The paper presents an efficient Hybrid Genetic Search with Advanced Diversity Control for a large class of time-constrained vehicle routing problems, introducing several new features to manage the temporal dimension. New move evaluation techniques are proposed, accounting for penalized infeasible solutions with respect to time-window and duration constraints, and allowing to evaluate moves from any classical neighbourhood based on arc or node exchanges in amortized constant time. Furthermore, geometric and structural problem decompositions are developed to address efficiently large problems. The proposed algorithm outperforms all current state-of-the-art approaches on classical literature benchmark instances for any combination of periodic, multi-depot, site-dependent, and duration-constrained vehicle routing problem with time windows.

1. Introduction

Vehicle routing problems (VRP) with time constraints and requirements relative to customer assignments to vehicle types, depots, or planning periods constitute a class of difficult optimization problems. These settings are linked with numerous practical applications including logistics, goods transportation, refuse collection, maintenance operations, and relief supply (see [1] for a large variety of application cases). Much has already been dedicated to specific VRP with additional "attributes" such as time windows, multiple depots, or delivery-period choices. As illustrated by numerous reviews ([2–6], for the most recent), almost every prominent meta-heuristic paradigm, including evolutionary methods, ant colony optimization, tabu search, simulated annealing, other improved local search approaches or ruin-and-recreate, has been applied to at least one of the previous settings. Yet, besides highly problem-tailored methods, the literature critically lacks good generalist approaches able to efficiently address a wide range of problem variants, and VRP combining several problem attributes, also called multi-attribute VRP in Crainic et al. [7] and rich VRP in Hartl et al. [8], still constitute major challenges for both research and applications. Within these settings, it is well known that imposing time-window constraints on customer services (and depot availability) raises significant challenges related to the smaller proportion of feasible solutions, the increased computation burden required to evaluate moves in neighbourhood search, and the antagonist influence of temporal and spatial characteristics.

This paper introduces a new *Hybrid Genetic Search with Advanced Diversity Control* (*HGSADC*), which addresses some of these challenges. In particular, it addresses efficiently a wide range of large-scale *vehicle routing problems with time windows* (*VRPTW*), route-duration constraints, and additional attributes involving requirements for customer assignments to particular vehicles types, depots or planning periods. The main characteristic of HGSADC stands in a different approach to population diversity management, the contribution of a particular individual to the diversity of the population appearing as a proper objective to be optimized [9]. We also introduce a number of new algorithmic features targeting specifically the temporal characteristics of the problems. We thus propose simple move evaluation procedures that accommodate penalized infeasibility with regard to duration and time-window constraints, and work in amortized $O(1)$ for any neighbourhood based on a bounded number of arc exchanges or node relocations. We also develop neighbourhood pruning procedures based on the temporal dimension, and decomposition principles to address efficiently large problem instances involving up to 1000 customers and 4500 services. The resulting algorithm is simple and efficient,

* Corresponding author at: Institut Charles Delaunay - LOSI, Université de Technologie de Troyes, 12 rue Marie Curie, BP 2060, 10010 Troyes Cedex, France. Tel.: +33 3 25 71 58 54; fax: +33 3 25 71 56 49.
E-mail addresses: thibaut.vidal@cirrelt.ca (T. Vidal), TeodorGabriel.Crainic@cirrelt.ca (T.G. Crainic), michel.gendreau@cirrelt.ca (M. Gendreau), christian.prins@utt.fr (C. Prins).

0305-0548/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cor.2012.07.018

outperforming all existing approaches on classical benchmarks for the VRPTW, its variants with multiple depots (*MDVRPTW*), multiple periods (*PVRPTW*), and vehicle-site dependencies (*SDVRPTW*).

The main contributions of this paper are thus (1) a generalization of the concept of HGSADC to a large class of VRPTW variants presenting mixed temporal and geometrical characteristics; (2) new procedures to efficiently search neighbourhoods when considering infeasible solutions with regards to duration and time-window constraints; (3) decomposition principles within the genetic framework allowing to address efficiently large instances; and (4) a state-of-the-art meta-heuristic for four classes of vehicle routing problems with time windows.

The remainder of this paper is organized as follows: Section 2 states the notation and formally defines the problems. Summarized elements of the literature are presented in Section 3. The proposed meta-heuristic is described in Section 4, while extensive computational experiments are reported in Section 5. Section 6 concludes.

2. Problem statement

We first formally state the VRPTW. A general PVRPTW is then defined, including other notable variants, such as the MDVRPTW and the SDVRPTW, as special cases. Route-duration constraints are included in all cases but do not appear in the acronyms.

Let $G=(\mathcal{V},\mathcal{A})$ be a complete directed graph. Vertex $v_0 \in \mathcal{V}$ represents a single depot, where a fleet of m identical vehicles with capacity Q is located, and a product to be delivered is kept. Each other vertex $v_i \in \mathcal{V}^{CST}$, with $\mathcal{V}^{CST}=\mathcal{V}\backslash\{v_0\}$ and $i \in \{1,\ldots,n\}$, stands for a customer to be serviced, characterized by a non-negative demand q_i, a service duration τ_i, as well as an interval of allowable visit times $[e_i,l_i]$, called time window. By definition, $q_0=\tau_0=0$. Arcs $(i,j) \in \mathcal{A}$ represent the possibility to travel from v_i to v_j with a distance c_{ij} and a duration δ_{ij}. A feasible route r is defined as a circuit in G that starts and ends at v_0, such that the total demand of customers in r is smaller than or equal to Q. While performing its route, a vehicle may stop and wait in order to reach the next customer within its time window, but the route duration, computed as the difference between the start time and the return time at v_0, is limited to D. The VRPTW aims to construct up to m vehicle routes, to visit each customer vertex once within its time window, while minimizing the total distance.

In the generalized PVRPTW, route planning is performed for a horizon of t periods. Distances c_{ijl} and durations δ_{ijl} can be dependent upon the period. Each customer v_i is characterized by a frequency f_i, representing the total number of services requested on the planning horizon, and a list L_i of allowable visit combinations, called patterns. The objective is to select a pattern for each customer, and construct the associated routes to minimize the total distance over all periods. A mathematical integer programming formulation of this problem is given in Appendix.

We also consider two related problem classes. The MDVRPTW involves $d>1$ depots. It is also generally assumed that any route originates and returns to the same depot. This problem was shown in Cordeau et al. [10] to constitute a special case of the generalized PVRPTW , where depots are assimilated to periods ($t=d$), any customer $v_i \in \mathcal{V}^{CST}$ has a frequency $f_i=1$ and can be serviced in any period $L_i=\{\{1\},\ldots,\{t\}\}$, and δ_{0il} and c_{0il} values are set in each period to correctly account for the distance to the assimilated depots. The SDVRPTW involves w vehicle types, with compatibility constraints between customers and vehicles. Each customer $v_i \in \mathcal{V}^{CST}$ can be serviced only by a subset $R_i \in \{1\ldots w\}$ of vehicle types. As shown in Cordeau and Laporte [11], this problem constitutes another particular case of PVRPTW, where each vehicle type is assimilated to a different period ($t=w$) and $L_i=\{\{k\}:k \in R_i\}$.

Vidal et al. [9] demonstrated that any VRP with multiple depots and periods (MDPVRP) can be transformed into an equivalent PVRP, by associating a different period to each (period, depot) pair from the former problem. In the same spirit, we can transform a problem combining multi-depot, site-dependent, multi-period, and time-window attributes into a PVRPTW, by associating a period for each (depot, period, vehicle type) in the original problem. This transformation thus enables to address all the previous VRPTW variants and their combinations by means of a single algorithm for the PVRPTW. The inherent difficulty related to combined (depot, period, vehicle type) choices leads to a large number of periods in the new problem. The proposed algorithm has thus been designed to successfully tackle large PVRPTW instances.

3. Literature review

We initiate this review surveying proposed meta-heuristics for the VRPTW, which is one of the most intensively studied NP-hard combinatorial optimization problems in the last thirty years. Exact methods are still not able to address most large-size applications, and their performance strongly varies with the time-window characteristics. Heuristic and meta-heuristic approaches have thus been the methodology of choice [2,3,6], and have been mostly evaluated and compared on standard benchmark instances introduced by Solomon [12] and Gehring and Homberger [13] relative to their computational efficiently and the quality of the solutions obtained. Most authors have focused on primarily minimizing fleet size, and then distance, but a few exceptions exist [14,15]. As a consequence, state-of-the-art VRPTW heuristics are generally based on two stages, dedicated first to minimizing fleet size and then distance.

Most successful approaches involve local search improvement procedures based on arc- and node-exchange neighbourhoods, and are coupled with various other concepts listed in the following:

- Evolution strategies: Alvarenga et al. [14], Labadi et al. [15], Mester and Bräysy [16], Hashimoto et al. [17], Repoussis et al. [18], Nagata et al. [19];
- Solutions recombinations: Alvarenga et al. [14], Labadi et al. [15], Hashimoto et al. [17], Repoussis et al. [18], Nagata et al. [19];
- Ruin and recreate: Repoussis et al. [18], Pisinger and Ropke [20], Prescott-Gagnon et al. [21];
- Ejection chains: Nagata et al. [19], Lim and Zhang [22], Nagata and Bräysy [23];
- Guidance and memories: Mester and Bräaysy [16], Repoussis et al. [18], Le Bouthillier and Crainic [24];
- Parallel and cooperative search: Le Bouthillier and Crainic [24,25];
- Mathematical programming hybrids: Prescott-Gagnon et al. [21].

The most competitive results are currently offered by the hybrid genetic algorithm of Nagata et al. [19]. The method combines powerful route minimization procedures, with a very effective *edge assembly crossover*, and extremely efficient local search procedures. The Iterated Local Search (ILS) of Ibaraki et al. [26], the Adaptive Large Neighbourhood Search (ALNS) of Pisinger and Ropke [20], and the Unified Tabu Search (UTS) of Cordeau et al. [10,27], Cordeau and Laporte [11], Cordeau et al. [28] are also worth mentioning. These methods stand out in terms of simplicity and wider applicability, as both have been extended to address various VRP variants.

Variants of the VRPTW, combining time windows with multi-period, multi-depot, or site dependency, arise in many practical applications such as maintenance operations [29,30], refuse collection [31,32], or product distribution [33–37]. Although the interest in these problem settings is growing, a somewhat restricted number of contributions have been proposed in the literature addressing them. Most of these implemented some form of neighbourhood-based meta-heuristic search. UTS [10,11,27,28] is currently the only method addressing all variants considered in this paper. UTS exploits long-term memories to penalize frequently encountered solution features. A parallel variant of this approach has been recently proposed by Cordeau and Maischberger [38].

Specific to the PVRPTW, Pirkwieser and Raidl [39] propose a Variable Neighborhood Search (VNS), further enhanced by means of multi-start strategies, column generation hybridizations, or multi-level strategies [40–42]. Yu and Yang [43] propose a coarse-grained parallel Ant Colony Optimization (ACO) algorithm, and Nguyen et al. [44] develop a hybrid genetic approach. The latter method combines the strength of population-based search, and Tabu and VNS improvement methods applied to the offspring. For the MDVRPTW, we report the hybrid Tabu search and savings approach of Tamashiro et al. [45], the parallel VNS approaches of Polacek et al. [46,47], and a genetic algorithm and ACO hybrid by Ostertag [48]. Noteworthy is also the approach of Chiu et al. [36], which considers total duty time minimization (including waiting times). Other than UTS, a single hybrid Tabu search and VNS approach by Belhaiza [49] may be reported for the SDVRPTW.

The literature is extremely scarce on VRPTW variants combining multiple features. We mention the Tabu search of Parthanadee and Logendran [37], able to address multi-depot periodic VRPTW (MDPVRPTW). Crainic et al. [7] introduced the Integrative Cooperative Search (ICS) framework to target highly complex combinatorial optimization settings. ICS is a central-memory cooperative multi-search involving problem decompositions by decision sets, integration of elite partial solutions yielded by the subproblems, and adaptive guidance mechanisms. An MDPVRPTW application was presented, but no definitive results have been published yet.

We conclude this section focusing on the design of efficient local-search methods, because most methods presented in this section dedicate the greatest part of their computational effort to the serial exploration of neighbourhoods, basically edge exchanges. Efficient move evaluation procedures are thus determining for both algorithmic speed and scalability.

Many methods rely on search spaces that include infeasible solutions with respect to time-window constraints, aiming to explore a wider diversity of structurally different feasible solutions. Several relaxation alternatives have been proposed in the literature. Cordeau et al. [27] and Repoussis et al. [18], among others, allow penalized late services to customers, while Ibaraki et al. [26] also allow penalized early services. Using such relaxations however leads to less efficient move-evaluation procedures, working in $O(n)$ or $O(\log n)$. A different relaxation was recently used by Hashimoto et al. [17] and Nagata et al. [19]. An assumption is made that upon a late arrival, a penalized return in time can be employed to reach the time window. The authors demonstrated that several classical neighbourhood moves can be evaluated in amortized $O(1)$ in this relaxation scheme. However, neither intra-route moves, nor duration constraints are actually managed in $O(1)$.

This review clearly underlines several gaps in the actual state of the art. Thus, for example, while population-based methods have shown their worth on the classic VRPTW, there is a lack of really efficient methods of this type for more complex variants such as PVRPTW, MDVRPTW, SDVRPTW and their combinations. Also, most current efficient methods for VRPTW are intricate, hard to reproduce, and largely rely on specific problem-tailored procedures. Hence, there is a need for more general and simple methods, broadly applicable to a large variety of practical settings with combined features. Finally, the temporal aspects lead to important challenges regarding infeasible-solution management and neighbourhood-evaluation procedures, which have a strong impact on the efficiency and scalability of VRPTW meta-heuristics. The concepts developed in this paper contribute towards addressing these issues.

4. The HGSADC methodology

This section describes the proposed Hybrid Genetic Search with Adaptive Diversity Control algorithm for time window-constrained VRP variants. For matters of presentation clarity, we describe the approach for the VRPTW and PVRPTW, the latter encompassing the MDVRPTW, SDVRPTW and other problems as special cases.

HGSADC [9] is a hybrid meta-heuristic combining the exploration capabilities of genetic algorithms with efficient local search-based improvement procedures and diversity management mechanisms. In HGSADC, population diversity is considered as an objective to be optimized along with solution quality through individual evaluations and selections. The general behavior of HGSADC is sketched in Algorithm 1.

Algorithm 1. HGSADC

1: Initialize population
2: **while** *number of iterations without improvement* $< It_{NI}$, *and time* $< T_{max}$ **do**
3: Select parent solutions P_1 and P_2
4: Create offspring C from P_1 and P_2 (crossover)
5: Educate C (local search procedure)
6: **if** C infeasible **then**
Insert C into infeasible subpopulation,
Repair with probability P_{rep}
7: **if** C feasible **then**
Insert C into feasible subpopulation
8: **if** *maximum subpopulation size reached* **then**
Select survivors
9: **if** *best solution not improved for* It_{div} *iterations*, **then**
Diversify population
10: Adjust penalty parameters for infeasibility
11: **if** *number of iterations* $= k \times It_{dec}$ where $k \in \mathbb{N}^*$, **then**
Decompose the master problem
Use HGSADC on each subproblem
Reconstitute three solutions, and insert them in the population
12: **end while**
13: Return best feasible solution

The method evolves feasible and infeasible solutions in two separate subpopulations. Genetic operators are iteratively applied to select two parents from the subpopulations (Line 3 of Algorithm 1), combine them into an offspring (Line 4), which undergoes a local search-based *Education*, is *Repaired* if infeasible, and is finally inserted into the suitable subpopulation (Lines 5–7). Each subpopulation is managed separately to trigger a *Survivor Selection* phase when a maximum size is reached, adapt infeasibility penalties, and call a *Diversification* mechanism (after each It_{div} successive iterations without improvement, Lines 8–10) whenever the search stagnates. In this application, structural

and geometrical decompositions phases are also performed (after each It_{dec} iterations, Line 11) to tackle large problems, the subproblems being addressed by means of recursive calls to HGSADC. The method terminates when It_{NI} successive iterations have been performed without improvement.

The main components of the method are described in the following subsections. The search space is presented in Section 4.1. Sections 4.2–4.4 briefly recall the solution representation, individual evaluation, selection, and crossover operators which, as the population management of Section 4.6, remain unchanged from Vidal et al. [9]. We then detail, in Section 4.5, the new neighbourhood-based evaluation procedures within *Education* and *Repair*, specifically developed for the temporal characteristics of the problems. Finally, Section 4.7 presents structural and geometrical problem decompositions that enable to address efficiently large instances. All these components together lead to a highly efficient algorithm for VRPTW variants.

4.1. Search space

The efficient exploitation of penalized infeasible solutions is known to contribute significantly to the performance of heuristics [9,50]. The search space of HGSADC involves infeasible solutions with respect to route constraints: load, duration, and time windows. The fleet-size limit is always respected, as a solution with too many vehicles may require sophisticated and computationally costly route-reduction methods to be repaired. Time windows are relaxed following the lines of Nagata [51] and Nagata et al. [19]. Upon a late arrival to a customer, one pays for a "time warp" to reach the end of the time window. This choice of relaxation is motivated by the availability of efficient penalty evaluation procedures within neighbourhood searches (Section 4.5).

Fig. 1 (inspired by Nagata et al. [19]) illustrates the previous assumptions on a route with five stops, which are represented from bottom to top with their time windows. The time dimension corresponds to the horizontal axis, while the vertical axis represents the progression on the route. A possible schedule is represented in bold line. This schedule presents some waiting time before service to v_2, and a time warp, triggered by a late arrival to v_4. Time warps are in some sense symmetric to waiting times, although waiting times are not penalized. Let r be a route, which starts from depot v_0 ($\sigma^r_0=0$), visits n_r customers $(\sigma^r_1,\ldots,\sigma^r_{n_r})\in\mathcal{V}^{CST}$, and returns to the depot $\sigma^r_{n_r+1}=0$. Let $\mathbf{t}^r=(t^r_0,\ldots,t^r_{n_r+1})$ be the visit times associated to each stop. On the way from a vertex σ^r_i to σ^r_{i+1}, the incurred time warp is given by $tw_{i,i+1}=\max\{t^r_i+\tau_{\sigma^r_i}+\delta_{\sigma^r_i\sigma^r_{i+1}}-t^r_{i+1},0\}$. The following quantities characterize route r:

- Load $q(r)=\sum_{i=1,\ldots,n_r} q_{\sigma^r_i}$;
- Distance $c(r)=\sum_{i=0,\ldots,n_r-1} c_{\sigma^r_i\sigma^r_{i+1}}$;

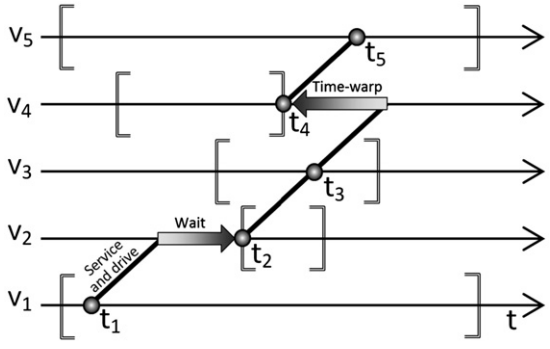

Fig. 1. Illustration of waiting times and time warps.

- Time-warp use $tw(r)=\sum_{i=0,\ldots,n_r-1} tw_{i,i+1}$;
- Duration $\tau(r)=t^r_{n_r+1}-t^r_0+tw(r)$

The *penalized cost* $\phi(r)$ of route r with schedule $\mathbf{t}^r$, presented in Eq. (1), is defined as its total distance plus the weighted sum of its excess duration, load, and time-warp use.

$$\phi(r)=c(r)+\omega^{D}\max\{0,\tau(r)-D)\}+\omega^{Q}\max\{0,q(r)-Q\}+\omega^{TW}\times tw(r) \quad (1)$$

Finally the penalized cost $\phi(s)$ of solution s, involving a set of routes $\mathcal{R}(s)$, is given by the sum of the penalized costs of all its routes.

4.2. Solution representation

The solution representation defined previously in HGSADC [9] is fairly general, and can be applied without change in the present context. A solution is represented as a two-chromosome *individual without trip delimiters*. The chromosomes account for the visit-period choices for each customer and the sequences of services for each period, respectively. The representation without trip delimiters, introduced in Prins [52], allows for simple recombination operators working on sequences, without the need to explicitly account for the individual routes. Then, to obtain a full solution from an individual representation, a polynomial *Split* algorithm, based on a shortest path procedure, is applied for each period to optimally partition the sequence of customers into routes. In our context, we use a *Split* algorithm that respects the maximum number of routes [53], and includes in the auxiliary graph penalized infeasible routes r regarding duration, load, and time-window constraints, and such that $q(r)\leq 2Q$. Reversely, any PVRPTW solution, represented by its routes for each period, can be transformed into an individual by removing visits to the depot.

4.3. The diversity and cost objective for evaluating individuals

Any individual P in the population is characterized by its solution cost $\phi(P)$ (Section 4.1), and its *diversity contribution* $\Delta(P)$ defined as the average distance from P to its closest neighbours $\mathcal{N}_{close}$ in the subpopulation:

$$\Delta(P)=\frac{1}{|\mathcal{N}_{close}|}\sum_{P_2\in\mathcal{N}_{close}}\delta(P,P_2) \quad (2)$$

For the PVRPTW and SDVRPTW, we rely on a Hamming distance δ measuring the proportion of customers with identical patterns or vehicle type assignments [9]. For the VRPTW and MDVRPTW, experiments led to choose the *broken pairs* distance (see [54]), which evaluates the amount of common arcs.

The evaluation, *biased fitness*, $BF(P)$ of an individual P (Equation 3) is then a "diversity and cost objective" that involves both the rank $fit(P)$ of P in the subpopulation with regards to solution cost $\phi(P)$, and its rank $dc(P)$ in terms of diversity contribution $\Delta(P)$ (Equation 2). $BF(P)$ depends upon the actual number of individuals in the subpopulation N_{indiv}, and a parameter N_{elite} ensuring elitism properties during survivor selection. This trade-off between diversity and elitism is critical for a thorough and efficient search

$$BF(P)=fit(P)+\left(1-\frac{N_{elite}}{N_{indiv}}\right)dc(P) \quad (3)$$

4.4. Parent selection and crossover

An iteration of HGSADC corresponds to the generation of a new individual by a succession of genetic operations. Two parents are first selected by binary tournament in the union of both

feasible and infeasible populations, and used as input to the crossover operator. The PIX crossover [9] is used for the PVRPTW. PIX enables to inherit good sequences of visits from both parents, and also to recombine visit patterns. For the VRPTW, we rely on the simple Ordered Crossover (OX) (see [52], for instance). These crossovers allow both small solution refinement and more important structural changes.

4.5. Neighbourhood search for VRPTW education and repair

An offspring resulting from the crossover operator undergoes the *Split* procedure to extract its routes. A neighbourhood search-based improvement operator, called *education*, is then systematically applied, followed by a *repair* phase, called with probability P_{rep}, when the resulting solution is infeasible. *Repair* increases the penalty values by a factor of 10 and calls *education*, aiming to restore the solution feasibility. This process is repeated with a penalty increase of 100 if the offspring remains infeasible.

Education and repair are essential for a fast progression toward high-quality solutions. Yet, these procedures tend inevitably to make for the largest part (90–95%) of the overall computational effort, such that high computational efficiency is required. Three basic aspects are decisive for performance: (1) a suitable choice of neighbourhood, restricted to relevant moves while being large enough to allow some structural solution changes; (2) memory structures to evade redundant move computations; and (3) highly efficient neighbour cost and feasibility evaluations. We introduce new methodologies to address these aspects relatively to the specificities of time-constrained VRPs.

4.5.1. Neighbourhood choices and restrictions

Following Vidal et al. [9], *education* is performed by means of two local search-based procedures. The *route improvement* procedure (*RI*) is dedicated to optimize services from each period separately, while the *pattern improvement* procedure (*PI*) relies on a quick and simple move to improve assignment choices. These local searches, called in the sequence RI,PI,RI, provide the means to address efficiently both service sequencing and assignment characteristics.

The PI procedure evaluates for each customer in random order the best combination of re-insertions within periods. Any improving re-insertion is directly performed until no more improvement can be found. For a customer v_i, the number of possible places of insertion is $O(N+m\times t)$, N representing the total number of customer services in all periods, and $m\times t$ representing the total number of routes, to account for insertions after the depot. Once insertion costs at all different places are known (and thus also the best cost insertion for each period), the best visiting period combination is computed for each customer v_i in $O(f_i|L_i|)$.

The RI procedure explores for each period a neighbourhood based on relocations and exchanges of customer visit sequences, with eventual inversions. A broader range of moves than in Vidal et al. [9] is exploited to cope with the increased variety of VRPTW solution structures, along with more advanced neighbourhood pruning procedures. The following neighbourhoods are evaluated:

- $\mathcal{N}_1$ (Swap and relocate): Swap two disjoint visit sequences $(\sigma_i^r,\ldots,\sigma_j^r)$ and $(\sigma_{i'}^{r'},\ldots,\sigma_{j'}^{r'})$, containing between 0 and 2 visits. Combine this with the reversal of one or both sequences.
- $\mathcal{N}_2$ (2-opt*): Swap two visit sequences $(\sigma_i^r,\ldots,\sigma_{n_r}^r)$ and $(\sigma_{i'}^{r'},\ldots,\sigma_{n_{r'}}^{r'})$, involving the extremities of two distinct routes.
- $\mathcal{N}_3$ (2-opt): Reverse a visit sequence $(\sigma_i^r,\ldots,\sigma_j^r)$.

Neighbourhoods $\mathcal{N}_1$ and $\mathcal{N}_2$ can involve one empty sequence. $\mathcal{N}_1$ involves eventually the same route or different routes. The size of these neighbourhoods is $O(n^2)$. One example of "swap" move from $\mathcal{N}_1$ is illustrated in Fig. 2. This move exchanges one visit from a route r with two visits from a distinct route r'.

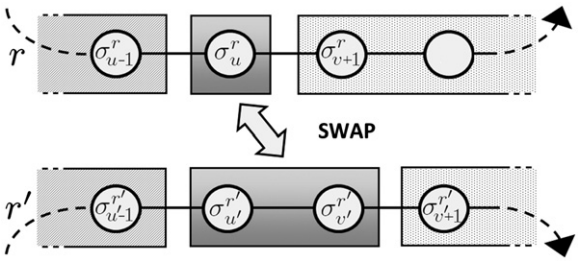

Fig. 2. Example of "swap" move in $\mathcal{N}_1$.

In the context of time-window constraints, moves in RI are explored in two phases: first the moves between "new" routes not existing in the parents, and then, the other moves. Each neighbourhood subset is searched in random order. The best improving move, when existing, is applied as soon as 5% of the neighbourhood has been explored since last move acceptance. This strategy is motivated by the need to perform quick improvements, focus on the new solution elements, and remain close to the characteristics of the original individuals, but aim for a better performance than that usually offered by first-improvement heuristics.

To further increase the computational efficiency of education, neighbourhoods are also pruned by means of *customer correlation* measures. The set of correlated neighbours is usually defined in the context of traveling salesman and classical vehicle routing problems in relation to a spatial proximity measure [55,56]. Time-constrained VRP involve another dimension, however, related to time proximity, as well as additional asymmetry issues. Correlation relationships are thus harder to define [57]. We define for customer v_i a set $\Gamma(v_i)$ of correlated customers as the $|\Gamma|$ closest customers v_j in relation to the correlation measure $\gamma(v_i,v_j)$ of Eq. (4). These customers can be viewed as the most relevant options for a direct visit from v_i, and thus arcs (v_i,v_j) for $v_j\in\Gamma(v_i)$ can be seen as a subset of "promising" arcs.

$$\gamma(v_i,v_j)=c_{ij}+\gamma^{WT}\max\{e_j-\tau_i-\delta_{ij}-l_i,0\}+\gamma^{TW}\max\{e_i+\tau_i+\delta_{ij}-l_j,0\} \tag{4}$$

This correlation measure corresponds to a weighted sum of the distance, the minimum waiting time, and the minimum penalty on a direct service from v_i to v_j. Values for the γ^{TW} and γ^{WT} coefficients, which balance the role of these spatial and temporal components, are discussed in Section 5.1. Neighbourhoods $\mathcal{N}_1$ and $\mathcal{N}_2$ are then restricted to sequences $(\sigma_i^r,\ldots,\sigma_j^r)$ and $(\sigma_{i'}^{r'},\ldots,\sigma_{j'}^{r'})$ such that $\sigma_i^r\in\Gamma(\sigma_{i'-1}^{r'})$ or $\sigma_{i'}^{r'}\in\Gamma(\sigma_{i-1}^{r})$, while $\mathcal{N}_3$ is restricted to sequences $(\sigma_i^r,\ldots,\sigma_j^r)$ such that $\sigma_j^r\in\Gamma(\sigma_{i-1}^r)$ or $\sigma_{j+1}^r\in\Gamma(\sigma_i^r)$. This restriction ensures that at least one "promising" arc is introduced within each move. The resulting neighbourhood size becomes $O(|\Gamma|n)$.

4.5.2. Memories

Memories are used in PI to store for each customer v_i the minimum cost insertion $\psi(i,r,l)$ in each route r and each period l. For RI, the cost of the best move is stored for each pair of customers. These values are valid until the routes under consideration are modified. These techniques lead to notable reductions in the overall computational effort.

4.5.3. Move evaluations

When infeasible solutions are used, evaluating moves implies to compute the change in total arc costs, as well as the variation

of duration, load, and time-window infeasibility of the routes. Calculation of cost and load variation is straightforward to perform in amortized $O(1)$ for moves based on a constant number of arc exchanges [58]. Nagata et al. [19] also provided the means to compute infeasibility in $O(1)$ for some neighbourhoods, including 2-opt*, inter-route swaps, and inter-route inserts. This method can not address as efficiently intra-route moves or more complex neighborhoods, however, and does not actually manage duration features. We thus introduce new procedures to evaluate combined duration and time-window infeasibility in amortized $O(1)$. The proposed approach is widely applicable to any classical neighbourhood based on a constant number of arc exchanges or sequence relocations.

We first observe that any such move can be viewed as a separation of routes into subsequences, which are then concatenated into new routes. This simple property is formalized in Kindervater and Savelsbergh [58], Irnich [59], and Vidal et al. [60]. In the example of Fig. 2, the move produces indeed two new routes, $(\sigma_0^r,\ldots,\sigma_{i-1}^r)\oplus(\sigma_{i'}^{r'},\sigma_{j'}^{r'})\oplus(\sigma_{j+1}^r,\ldots,\sigma_{n_r}^r)$ and $(\sigma_0^{r'},\ldots,\sigma_{i'-1}^{r'})\oplus(\sigma_i^r)\oplus(\sigma_{j'+1}^{r'},\ldots,\sigma_{n_{r'}}^{r'})$, where $\oplus$ represents the concatenation operator. Our move evaluation approach follows from this observation, and uses induction on the concatenation operation to develop suitable *re-optimization data* on subsequences of consecutive visits in the incumbent solution.

For each such subsequence σ, containing visits to depots or customers, we compute the minimum duration $D(\sigma)$, minimum time-warp use $TW(\sigma)$, earliest $E(\sigma)$ and latest visit $L(\sigma)$ to the first vertex allowing a schedule with minimum duration and minimum time-warp use, as well as the cumulated distance $C(\sigma)$ and load $Q(\sigma)$. This data is straightforward to compute for a sequence σ^0 involving a single vertex v_i, as $D(\sigma^0)=\tau_i$, $TW(\sigma^0)=0$, $E(\sigma^0)=e_i$, $L(\sigma^0)=l_i$, $C(\sigma^0)=0$ and $Q(\sigma^0)=q_i$. Proposition 1 then enables to compute the same data on concatenations of sequences. Eqs. (9)–(10) are frequently used in the VRP literature to calculate loads and costs. The other statements, which target the temporal aspects of the problem, are proven in Appendix B.

Proposition 1 (*Concatenation of two sequences*). *Let* $\sigma=(\sigma_i,\ldots,\sigma_j)$ *and* $\sigma'=(\sigma_{i'}',\ldots,\sigma_{j'}')$ *be two subsequences of visits. The concatenated subsequence* $\sigma\oplus\sigma'$ *is characterized by the following data*:

$$D(\sigma\oplus\sigma')=D(\sigma)+D(\sigma')+\delta_{\sigma_j\sigma_{i'}'}+\Delta_{WT} \tag{5}$$

$$TW(\sigma\oplus\sigma')=TW(\sigma)+TW(\sigma')+\Delta_{TW} \tag{6}$$

$$E(\sigma\oplus\sigma')=\max\{E(\sigma')-\Delta,E(\sigma)\}-\Delta_{WT} \tag{7}$$

$$L(\sigma\oplus\sigma')=\min\{L(\sigma')-\Delta,L(\sigma)\}+\Delta_{TW} \tag{8}$$

$$C(\sigma\oplus\sigma')=C(\sigma)+C(\sigma')+c_{\sigma_j\sigma_{i'}'} \tag{9}$$

$$Q(\sigma\oplus\sigma')=Q(\sigma)+Q(\sigma') \tag{10}$$

where $\Delta=D(\sigma)-TW(\sigma)+\delta_{\sigma_j\sigma_{i'}'}$, $\Delta_{WT}=\max\{E(\sigma')-\Delta-L(\sigma),0\}$ *and* $\Delta_{TW}=\max\{E(\sigma)+\Delta-L(\sigma'),0\}$.

The neighbourhood evaluation procedure we propose relies on Proposition 1 to first develop data on relevant consecutive visit subsequences (and their reversal) in a preprocessing phase, and then to evaluate the penalties and costs of routes issued from the moves. Classical neighbourhoods in the literature correspond to a concatenation of less than five subsequences. Hence, given the data on subsequences, any move evaluation is performed in constant time. As shown in Vidal et al. [60], this property stands for any move issued from a constant number of arc exchanges or customer visit relocations.

Experiments showed that, for some problem instances with long routes with more than 50 customers, data preprocessing on all $O(n^2)$ subsequences can play a non-negligible role in the overall computation effort. The 1-level or 2-level strategy of Irnich [59] can be employed to limit this preprocessing to $O(n^{4/3})$ or $O(n^{8/7})$ subsequences, while maintaining the constant time evaluation of moves. To make it even simpler, we limited the preprocessing to "prefix" (and "suffix") subsequences containing the first (the last) customer of a route, and subsequences of size smaller than 20. This data enables to evaluate inter-route moves in constant time, and allows an evaluation of intra-route moves as a concatenation of less than seven subsequences for the instances considered.

4.6. Population management and search guidance

The main components of HGSADC regarding population management remain unchanged from Vidal et al. [9]. The two subpopulations are set up to contain between μ and $\mu+\lambda$ individuals. To initialize the populations, 4μ individuals are created by randomly choosing the patterns and routes, using *Education* and, when infeasible, *Repair*. These individuals are then included in the appropriate subpopulations, which then evolve through iterative generation, education, and selections of individuals. Any solution produced by the education and repair operators is transformed into an individual by removing visits to depots (Section 4.2), and is included in the appropriate subpopulation with respect to its feasibility. It can thus be selected for mating immediately after education. Any subpopulation reaching the size $\mu+\lambda$ undergoes a *survivor-selection* phase, where λ individuals are discarded. Let a "clone" be an individual with the same solution cost, or null distance to another with respect to the metric defined in Section 4.3. Survivor selection removes iteratively λ times the worst clone in terms of biased fitness (Eq. (3)), or the worst individual when no clone exists. The use of the biased fitness for survivor selection promotes both elitism and innovation during the search.

The proportion of feasible solutions following education with regards to duration, capacity, and time-window constraints is monitored during the search on the last 100 generated individuals. Penalty coefficients $\omega^{\mathrm{D}},\omega^{\mathrm{Q}}$, and ω^{TW} (Section 4.1) are also adjusted each 100 iterations. Let parameter ξ^{REF} stand for the target proportion of feasible individuals. If the proportion of feasible individuals relatively to one type of constraint (duration, load or time-window) falls below $\xi^{REF}-5\%$ (or, rises to more than $\xi^{REF}+5\%$), then the corresponding penalty is increased (or, decreased).

A *diversification* phase finally occurs whenever $It_{div}=0.4It_{NI}$ iterations are performed without improving the best solution. Diversification retains the best $\mu/3$ individuals of each subpopulation, to be completed with 4μ new individuals as in the Initialization phase, thus introducing new genetic material.

All these components contribute towards a more thorough search, and complement advantageously the aggressive local improvement abilities of education and repair operators.

4.7. Decomposition phases

Variants of vehicle routing problems with time windows lend themselves well to various decomposition approaches, mostly based on geometry [48,61–63], temporal aspects [63], or problem structure [7], which enable to address large instances more efficiently.

We introduce a simple decomposition framework for population-based methods, which takes full advantage of their associated pool of solutions. The approach proceeds in four steps: (1) features

from one elite solution are exploited to define subproblems; (2) initial individuals for the subproblems are created from the genetic material of the complete problem population; (3) the algorithm, here HGSADC, is called to address these subproblems; (4) a set of complete solutions is finally reconstructed.

We rely on simple route-based geometrical decompositions in the VRPTW case. For the PVRPTW, fixing assignments to periods naturally decomposes the problem. It is worth noticing that the "integration" of partial solutions into solutions of the complete problem is here straightforward, as it simply involves gathering the routes. Also, any improvement in any subproblem leads to an improvement of the elite complete solution, as subproblems are built from the features of this solution.

In the current implementation, decomposition phases occur every It_{dec} iterations, and are only used on problems with more than 120 customers. The elite solution is randomly selected from the 25% best feasible individuals (or infeasible if no feasible solution has been found). In the VRPTW case, routes from this elite solution are swept circularly around the depot by polar angle of barycentre, and included in a set $\mathcal{R}_{dec}$. Each time the number of customers in routes from $\mathcal{R}_{dec}$ becomes larger than 120, or when all the routes have been swept, a VRPTW subproblem is created with the customers from $\mathcal{R}_{dec}$ and the set is emptied. Subproblems of this size can be very efficiently handled by HGSADC. In the case of the PVRPTW, we fix the periods of service, leading to a subproblem for each period.

Initial populations of subproblem solutions are created from the complete solutions population, by retaining from the route chromosomes of complete individuals only the services that occur in the subproblem at the right period. For the PVRTPW, visits from solutions with patterns different from those of the elite solution can be missing. These visits are completed by means of a least cost insertion heuristic. HGSADC is then run on each of these subproblems until $It_{dec}/2$ iterations without improvement are performed. Combining the best solution of each subproblem yields an elite solution, to be added to the population of the complete problem. The same process is repeated with the second and the third best solutions to produce two additional elite individuals.

5. Computational experiments

Extensive computational experiments were performed to analyze the impact of the parameter settings (Section 5.1), assess HGSADC performance when compared to state-of-the-art methods for each problem (Section 5.2), and evaluate the role of the new decomposition phases (Section 5.3). The algorithm is coded in C++, compiled with "g++ -O3", and run on a Intel Xeon 2.93 GHz processor.

We rely on a variety of classical benchmark instance sets: Solomon and Desrosiers [64] and Gehring and Homberger [13] VRPTW instances, Cordeau et al. [27] and Cordeau and Laporte [11] instances for PVRPTW, MDVRPTW, SDVRPTW with duration constraints and the PVRPTW instances of Pirkwieser and Raidl [41] without duration constraints. These instances involve from 48 to 1008 customers, up to nine depots, ten periods, and six vehicle types. It should be noted that the distances of Pirkwieser and Raidl [41] instances have been truncated to the first digit by previous authors. We used this convention exclusively in this case to perform a fair comparison. We also introduce new larger-dimension instances for PVRPTW, MDVRPTW, and SDVRPTW, involving 360–960 customers, and up to 4608 total services, following the generation procedure of Cordeau et al. [10,27]. These instances are available upon request.

Finally, the traditional objective for VRPTW is fleet minimization in priority, and then route length minimization. To apply HGSADC with this objective, we first constrain the fleet size to a large value of 100, and iteratively interrupt the run and reduce the fleet size whenever a feasible solution is found. As soon as HGSADC fails to find a feasible solution, we return to the last feasible fleet value and perform a final run. It is noteworthy that both objectives are here addressed in single-stage-single-algorithm mode contrasting with current state-of-the-art VRPTW methods, which generally rely on two distinct procedures and concepts.

5.1. Parameter calibration

In order to study the applicability and generality of the HGSADC framework with limited changes, we voluntarily limited the role of parameter tuning, to focus exclusively on the new parameters related to the neighbourhood evaluation procedures (γ^{WT}, γ^{TW}), and the decomposition phases. The remaining parameter values are kept the same as in [9], where an extensive meta-calibration procedure had been used to generate good parameter values on closely related PVRP and MDVRP instances. Thus, $N_{elite}=8$ and $|\mathcal{N}_{close}|=3$; $P_{rep}=0.5$ and $|\Gamma|=40$; $(\mu,\lambda)=(25,40)$ and $\xi^{REF}=0.2$ (Section 4.6). Finally, the termination criteria $It_{NI}=5000$ is used to compare with other authors in similar run times, while $It_{dec}=2000$ to balance the computation time dedicated to decomposition phases and the regular run.

The nature of VRPTW solutions tends to strongly vary in relation to the distribution and tightness of time windows. For some problems, high-quality solutions involve some long arcs and relatively small waiting times, while for less tightly constrained problems, closer to the classical VRP, long arcs become very unlikely. The parameters γ^{TW} and γ^{WT}, balancing the role of geometrical and temporal aspects during neighbourhood pruning, are thus critical, and need to be calibrated relatively to the instances at hand. To that extent, we selected ten problems with various structures (R1-2, R1-4, R2-1, R2-3, RC1-1, RC1-2, RC2-3, RC2-4, C1-2, and C2-4) from the 200-customer instances of Gehring and Homberger [13]. 20 runs were performed for each instance and each of the 36 combinations of parameters $(\gamma^{TW},\gamma^{WT})\in\{0;0.2;0.5;1;2;5\}^2$. To reduce computation time and amplify the impact of good move choices, we also reduced $|\Gamma|$ to 20, $It_{NI}=2000$, and turned off the decomposition phases. The minimum number of vehicles has been reached on all runs but five out of $20\times36=720$ total experiments. These five marginal results have been discarded in order to simply compare on the basis of distance. Table 1 presents the gap of HGSADC to the best known solution (BKS) in the literature for each parameter setting, averaged on the 20 (19 in some cases) experiments and ten instances. The line and column indicated in boldface corresponds to the best mean for each parameter taken independently.

Table 1
Performance of HGSADC on a selection of 200-customer VRPTW instances for various γ^{TW} and γ^{WT} settings.

γ^{TW}	γ^{WT}					
	0.0	0.2	0.5	1.0	2.0	5.0
0.0	0.70	**0.64**	0.69	0.72	0.71	0.82
0.2	0.59	**0.59**	0.65	0.57	0.71	0.72
0.5	0.63	**0.53**	0.56	0.63	0.65	0.67
1.0	**0.63**	**0.59**	**0.51**	**0.59**	**0.59**	**0.63**
2.0	0.60	**0.51**	0.61	0.63	0.62	0.61
5.0	0.56	**0.56**	0.57	0.56	0.70	0.61

HGSADC performance appears to increase with parameter values close to $(\gamma^{TW},\gamma^{WT})=(1.0,0.2)$. To state the statistical significance of these observations, we analyzed the distribution of the average deviation to BKS among the 20×36 experiments. The Shapiro-Wilk test rejects the distribution normality assumption with high confidence ($p=0.000$). Noteworthy is the fact that Levene's test also rejects ($p=0.011$) the equality of variances among the 36 groups of experiments, hence these parameters affect both the algorithm performance and stability.

Following these observations, a classic ANOVA is not relevant. We thus compared the settings $(\gamma^{TW},\gamma^{WT})=(0,0)$ and $(\gamma^{TW},\gamma^{WT})=(1.0,0.2)$ on 50 new runs, using new random number generator seeds. Average deviations of +0.68% and +0.53% were retrieved. A Wilcoxon test on these paired samples yields $p=0.000$, thus rejecting with very high confidence the null hypothesis that "both parameter settings lead to the same solution quality". The new setting (1.0,0.2), which accounts for the temporal aspect in neighbourhood pruning, leads to a significant increase in quality when compared to (0,0), which corresponds to the classic distance-driven granular search policy.

5.2. Comparison of performances

We compare HGSADC with state-of-the art methods for the PVRPTW in Tables 2 and 3, for the MDVRPTW in Table 4, for the SDVRPTW in Table 5, and for the VRPTW in Tables 6 and 7. We also report the performance of HGSADC on the new large scale instances in Table 8. The first group of columns displays the instance identifier, number of customers n, maximum fleet size m, number of periods t, depots d, and vehicle types w when applicable. The next group of columns compares the average and best results, as well as the average run time of HGSADC with state-of-the-art methods for each problem:

- B: Hybrid VNS and Tabu of Tabu of Belhaiza [49]
- CLM: UTS of Cordeau et al. [28]
- CM-8P and CM-64: Parallel iterative UTS of Cordeau and Maischberger [38] on 8 and 64 processors
- NB-100 and NB-f(n): Hybrid GA based on EAX crossover of Nagata et al. [19] with a population size of 100 and $f(n)=n/20\,000$.

Table 2
Results on Cordeau et al. [27] PVRPTW instances.

Inst	n	m	t	CLM	PR08	CM-8P	CM-64P	HGSADC			prev BKS	HGSADC
				1 run	Best X	Avg 10	Avg 5	Avg 10	Best 10	T (min)	–	All exp.
p01a	48	3	4	2915.58	**2909.02**	**2909.02**	**2909.02**	2909.05	**2909.02**	1.13	2909.02	2909.02
p02a	96	6	4	5094.39	5036.27	5046.78	5037.60	5031.50	**5026.57**	3.28	5026.57	5026.57
p03a	144	9	4	7284.32	7138.70	7134.11	7097.55	7091.51	**7050.72**	8.11	7062.00	7023.90
p04a	192	12	4	8087.06	7882.06	7923.48	7857.08	7818.75	**7791.93**	17.93	7807.32	7755.77
p05a	240	15	4	8752.72	8492.45	8518.20	8434.02	8368.98	**8341.93**	31.03	8358.96	8311.17
p06a	288	18	4	10 961.78	10 713.75	10 756.53	10 664.56	10 595.85	**10 477.01**	65.36	10 542.10	10 473.24
p07a	72	5	6	6891.76	6787.72	6799.12	6799.73	6788.67	**6783.23**	3.76	6782.68	6783.23
p08a	144	10	6	9990.46	9721.25	9729.13	9684.86	9623.72	**9593.43**	17.00	9603.92	9574.80
p09a	216	15	6	13 796.75	13 463.96	13 459.28	13 371.16	13 285.89	**13 247.38**	45.94	13 299.80	13 201.06
p10a	288	20	6	18 135.6	17 650.89	17 503.69	17 365.50	17 058.89	**16 999.88**	95.96	17 261.30	16 920.96
p01b	48	3	4	2297.21	**2277.44**	2278.41	**2277.44**	**2277.44**	**2277.44**	0.83	2277.44	2277.44
p02b	96	6	4	4335.11	4137.45	4200.75	4139.10	4130.64	**4122.03**	4.88	4124.76	4121.50
p03b	144	9	4	5699.78	5575.27	5601.34	5571.88	5555.77	**5521.71**	8.44	5489.84	5489.33
p04b	192	12	4	6619.56	6476.67	6482.60	6433.16	6400.55	**6352.28**	27.80	6383.28	6347.77
p05b	240	15	4	7138.28	6970.33	6902.39	6846.96	6838.54	**6790.44**	47.47	6800.45	6777.54
p06b	288	18	4	9039.29	8819.32	8760.22	8695.84	8647.15	**8595.10**	77.48	8659.44	8582.72
p07b	72	5	6	5580.22	5504.67	5514.35	5494.67	5491.08	**5481.61**	3.64	5481.61	5481.61
p08b	144	10	6	7914.39	7729.32	7772.38	7726.38	7665.10	**7619.95**	16.75	7656.13	7599.01
p09b	216	15	6	11 269.13	10 885.93	10 871.81	10 750.42	10 653.60	**10 589.68**	68.10	10 579.50	10 532.51
p10b	288	20	6	14 145.37	13 943.61	13 778.33	13 576.78	13 502.65	**13 442.57**	109.98	13 490.80	13 406.89
Avg Gap to BKS (%)				+3.54	+1.28	+1.31	+0.64	+0.17	−0.24	–		
Avg Time (min)				16.0	NC	8 × 7.6	64 × 11.32	32.74	10 × 32.74	–		
Processor				P4-2G	Opt-2.2G	Xe-2.93G	Xe-2.93G		Xe-2.93G			

Table 3
Results on Pirkwieser and Raidl [41] PVRPTW instances without duration constraints, distances truncated to the first digit.

Inst	n	t	PR09	PR10	NCT	HGSADC		
			Avg 30	Avg 30	Avg 10	Avg 10	Best 10	T (min)
R4	100	4	3454.50	3467.08	3441.86	3441.34	**3434.18**	3.03
C4	100	4	2787.14	2828.83	2778.19	2768.76	**2766.22**	2.68
RC4	100	4	3641.61	3659.66	3628.41	3630.81	**3620.70**	3.92
R6	100	6	4475.85	4474.04	4445.81	4443.48	**4428.88**	4.52
C6	100	6	3777.97	3807.68	3742.74	3728.94	**3723.20**	3.96
RC6	100	6	5030.33	5021.79	4967.34	4971.14	**4952.94**	4.85
R8	100	8	–	5526.47	5443.08	5456.67	**5428.80**	5.08
C8	100	8	–	4971.18	4860.52	4827.88	**4809.46**	4.23
RC8	100	8	–	5994.37	5902.67	5876.73	**5840.84**	5.88
Avg Gap to BKS			NC	+1.75%	+0.38%	+0.19%	−0.09%	
Avg Time (min)			0.86	0.61	97.51	4.24	10 × 4.24	
Processor			Qd-2.83G	Qd-2.83G	C2-2.4G		Xe-2.93G	

Table 4
Results on Cordeau et al. [27] MDVRPTW instances.

Inst	n	m	d	CLM	PBDH	CM-8P	CM-64P	HGSADC			prev BKS	HGSADC
				1 run	Best X	Avg 10	Avg 5	Avg 10	Best 10	T (min)	–	All exp.
p01a	48	2	4	**1074.12**	**1074.12**	**1074.12**	**1074.12**	**1074.12**	**1074.12**	0.31	1074.12	1074.12
p02a	96	3	4	1766.94	1763.66	1762.80	**1762.21**	1762.61	**1762.21**	1.15	1762.21	1762.61
p03a	144	4	4	2420.89	2388.73	2394.77	2380.24	2374.27	**2373.65**	1.75	2373.65	2373.65
p04a	192	5	4	2868.64	2847.56	2841.51	2822.80	2817.39	**2815.75**	5.89	2815.48	2815.11
p05a	240	6	4	3059.40	3015.27	3007.80	2987.01	2968.71	**2964.65**	8.68	2965.18	2962.25
p06a	288	7	4	3701.08	3674.60	3638.40	3616.69	3598.77	**3588.78**	13.43	3590.58	3588.78
p07a	72	2	6	1425.87	**1418.22**	**1418.22**	**1418.22**	**1418.22**	**1418.22**	0.51	1418.22	1418.22
p08a	144	3	6	2118.50	2103.21	2111.16	2101.50	2097.35	**2096.73**	2.39	2096.73	2096.73
p09a	216	4	6	2777.91	2753.61	2739.40	2723.81	2716.15	**2712.56**	5.20	2717.69	2712.56
p10a	288	5	6	3546.24	3541.01	3505.30	3481.58	3477.56	**3465.92**	15.22	3469.29	3464.65
p01b	48	2	4	1025.14	1011.65	**1005.73**	**1005.73**	**1005.73**	**1005.73**	0.51	1005.73	1005.73
p02b	96	3	4	1486.26	1488.32	1473.65	1468.30	1466.49	**1464.50**	1.68	1464.50	1464.50
p03b	144	4	4	2033.75	2012.37	2004.69	2001.83	2001.82	**2001.81**	2.94	2001.81	2001.81
p04b	192	5	4	2228.64	2239.02	2212.38	2199.70	2197.41	**2195.33**	6.55	2195.33	2195.33
p05b	240	6	4	2555.95	2498.85	2476.87	2443.94	2454.28	**2433.15**	12.56	2434.94	2433.15
p06b	288	7	4	2978.60	2909.45	2875.70	2862.38	2844.06	**2836.67**	15.97	2852.25	2836.67
p07b	72	2	6	1250.18	1247.51	1238.51	1237.00	1237.78	**1236.24**	1.05	1236.24	1236.24
p08b	144	3	6	1870.34	1809.25	1793.90	1790.44	1789.76	**1788.18**	3.30	1788.18	1788.18
p09b	216	4	6	2338.74	2294.19	2283.35	2276.32	2264.56	**2261.08**	8.59	2263.74	2257.13
p10b	288	5	6	3147.79	3093.51	3060.46	3016.73	3006.18	**2993.31**	22.18	2995.08	2984.01
Avg Gap to BKS				+2.32%	+1.28%	+0.74%	+0.27%	+0.10%	−0.06%	–		
Avg Time (min)				14.9	146.94	8 × 4.15	64 × 6.57	6.49	10 × 6.49	–		
Processor				P4-2G	P4-3.6G	Xe-2.93G	Xe-2.93G		Xe-2.93G			

Table 5
Results on Cordeau and Laporte [11] SDVRPTW instances.

Inst	n	m	w	CLM	B	CM-8P	CM-64P	HGSADC			prev BKS	HGSADC
				1 run	Best X	Avg 10	Avg 5	Avg 10	Best 10	T (min)	–	All exp.
p01a	48	2	4	1666.47	**1655.42**	**1655.42**	**1655.42**	**1655.42**	**1655.42**	0.23	1655.42	1655.42
p02a	96	3	4	2915.55	2938.37	**2904.13**	**2904.13**	**2904.13**	**2904.13**	0.70	2904.13	2904.13
p03a	144	4	4	3406.21	3338.58	3322.38	3317.25	3320.44	**3304.13**	1.60	3304.13	3304.13
p04a	192	5	4	4532.58	4560.96	4486.13	4448.48	4437.19	**4427.25**	5.85	4438.97	4427.25
p05a	240	6	4	5859.03	5908.41	5722.89	5666.23	5681.48	**5647.76**	11.64	5620.56	5626.42
p06a	288	7	4	5773.24	5966.33	5767.97	5710.96	5666.20	**5637.48**	12.68	5670.66	5627.82
p07a	72	2	6	2181.77	2169.06	**2168.88**	**2166.88**	**2166.88**	**2166.88**	0.42	2166.88	2166.88
p08a	144	3	6	3983.08	3944.33	3908.93	3885.31	3880.58	**3873.40**	2.35	3874.32	3873.40
p09a	216	4	6	5008.54	4985.19	4854.70	4838.82	4797.72	**4777.61**	5.60	4801.47	4772.55
p10a	288	5	6	6171.16	6118.20	5935.56	5881.95	5876.38	**5858.82**	11.58	5868.03	5817.28
p11a	1008	27	4	–	16 784.67	–	–	15 198.10	**15 080.68**	120.46	16 418.21	14 982.35
p12a	720	14	6	–	12 485.43	–	–	11 475.15	**11 402.01**	112.06	12 106.20	11 330.23
p01b	48	2	4	1433.24	1429.37	**1429.35**	**1429.35**	**1429.35**	**1429.35**	0.22	1429.35	1429.35
p02b	96	3	4	2516.83	2494.34	2489.83	2482.48	**2479.56**	**2479.56**	0.99	2479.56	2479.56
p03b	144	4	4	2814.61	2801.51	2785.84	2780.40	2779.09	**2775.61**	2.28	2775.61	2774.30
p04b	192	5	4	3762.38	3746.99	3695.27	3673.01	3660.66	**3649.72**	6.57	3655.48	3649.72
p05b	240	6	4	4955.04	4730.63	4678.58	4654.10	4625.79	**4611.16**	8.06	4613.09	4609.20
p06b	288	7	4	5008.27	5019.64	4823.92	4777.44	4755.59	**4729.96**	15.29	4752.04	4716.36
p07b	72	2	6	1864.11	**1837.94**	1839.08	**1837.94**	**1837.94**	**1837.94**	0.51	1837.94	1837.94
p08b	144	3	6	3215.06	3163.99	3161.03	3149.49	3152.69	**3149.77**	2.15	3144.91	3144.91
p09b	216	4	6	4033.63	4033.21	3959.74	3940.15	3894.67	**3883.94**	8.90	3894.64	3883.94
p10b	288	5	6	5158.89	5114.38	5014.48	5009.00	4962.62	**4932.40**	12.03	4967.59	4927.95
p11b	1008	27	4	–	14 655.00	–	–	13 226.60	**13 067.52**	120.32	14 015.50	12 998.63
p12b	720	14	6	–	10 864.70	–	–	9857.89	**9777.44**	120.17	10 267.50	9708.45
Gap p01-10				+2.76%	+2.23%	+0.82%	+0.39%	+0.12%	−0.13%			
Gap p11-12				–	+3.94%	–	–	−5.57%	−6.38%			
T (min) p01-10				13.3	2.94	8 × 4.53	64 × 5.60	5.48	10 × 5.48			
T (min) p11-12				–	45.73	–	–	118.25	10 × 118.25			
Processor				P4-2G	Qd-2.66G	Xe-2.93G	Xe-2.93G		Xe-2.93G			

A limit of 2 h has been set for HGSADC runs. The average gaps and time are reported separately for instances sets p01-10 and p11-12, which are of very different sizes.

- NCT: GA+VNS+Tabu of Nguyen et al. [44]
- PBDH: Cooperative VNS of Polacek et al. [47]
- PDR: Branch-and-price based LNS of Prescott-Gagnon et al. [21]
- PisR: ALNS of Pisinger and Ropke [20]
- PR08: VNS of Pirkwieser and Raidl [39]
- PR09: Multiple VNS+ILP (15,10) of Pirkwieser and Raidl [41]
- PR10: VNS+ILP of Pirkwieser and Raidl [42]

Table 6
Results on Solomon and Desrosiers [64] VRPTW instances.

Inst	n	PisR	PDR	RTI	NB-100	NB-f(n)	HGSADC	
		Best 10	Best 5	Best 3	1 run	Best 5	Avg 5	Best 5
R1	100	11.92|1212.39	**11.92|1210.34**	11.92|1210.82	**11.92|1210.34**	**11.92|1210.34**	11.92|1211.49	11.92|1210.69
R2	100	2.73|957.72	2.73|955.74	2.73|952.67	2.73|952.08	**2.73|951.03**	2.73|952.05	2.73|951.51
C1	100	**10.0|828.38**	**10.0|828.38**	**10.0|828.38**	**10.0|828.38**	**10.0|828.38**	**10.0|828.38**	**10.0|828.38**
C2	100	**3.0|589.86**	**3.0|589.86**	**3.0|589.86**	**3.0|589.86**	**3.0|589.86**	**3.0|589.86**	**3.0|589.86**
RC1	100	11.5|1385.78	**11.5|1384.16**	11.50|1384.30	11.50|1384.72	**11.5|1384.16**	11.5|1384.81	11.5|1384.17
RC2	100	3.25|1123.49	3.25|1119.44	3.25|1119.72	3.25|1119.45	**3.25|1119.24**	3.25|1119.40	**3.25|1119.24**
CNV		405	405	405	405	**405**	405	405
CTD		57 332	57 240	57 216	57 205	**57 187**	57 218	57 196
T (min)		10 × 2.5	5 × 30	3 × 17.9	3.2	5 × 5.0	2.68 min	5 × 2.68 min
Processor		P4-3G	Opt-2.3G	P4-3G	Opt-2.4G	Opt-2.4G	Xe-2.93G	

Table 7
Results on Gehring and Homberger [13] large-scale VRPTW instances.

Inst	n	LZ	PDR	RTI	NB-100	NB-f(n)	HGSADC	
		1 run	Best 5	Best 3	1 run	Best 5	Avg 5	Best 5
R1	200	18.2|3639.60	18.2|3615.69	18.2|3640.11	18.2|3615.15	**18.2|3612.36**	18.20|3621.72	18.2|3613.16
R2	200	4.0|2950.09	4.0|2937.67	4.0|2941.99	4.0|2930.04	**4.0|2929.41**	4.00|2933.19	**4.0|2929.41**
C1	200	18.9|2726.11	18.9|2718.77	18.9|2721.90	18.9|2718.44	**18.9|2718.41**	18.90|2720.13	**18.9|2718.41**
C2	200	6.0|1834.24	6.0|1831.59	6.0|1833.36	6.0|1831.64	6.0|1831.64	6.00|1832.08	**6.0|1831.59**
RC1	200	18.0|3205.51	18.0|3192.56	18.0|3224.63	18.0|3182.48	**18.0|3178.68**	18.00|3195.26	18.0|3180.48
RC2	200	4.3|2574.10	4.3|2559.32	4.3|2554.33	4.3|2536.54	4.3|2536.22	4.30|2538.29	**4.3|2536.20**
CNV		694	694	694	694	**694**	694	694
CTD		169 296	168 556	169 163	168 143	**168 067**	168 407	168 092
Time		93.2 min	5 × 53 min	90 min	4.7 min	5 × 4.1 min	8.40 min	5 × 8.40 min
R1	400	36.4|8489.53	36.4|8420.52	36.4|8514.11	36.4|8413.23	36.4|8403.24	36.40|8423.07	**36.4|8402.57**
R2	400	8.0|6271.57	8.0|6213.48	8.0|6258.82	8.0|6149.49	**8.0|6148.57**	8.00|6168.98	8.0|6152.92
C1	400	37.6|7229.04	37.6|7182.75	37.6|7273.90	37.6|7179.71	37.6|7175.72	37.60|7184.65	**37.6|7170.47**
C2	400	11.7|3942.93	11.9|3874.58	11.7|3941.70	11.7|3898.02	11.7|3899.00	11.68|3916.83	**11.6|3952.95**
RC1	400	36.0|8005.25	36.0|7940.65	36.0|8088.46	36.0|7931.66	36.0|7922.23	36.00|7942.81	**36.0|7907.14**
RC2	400	8.5|5431.15	8.6|5269.09	8.4|5516.59	**8.4|5293.74**	8.4|5297.86	8.52|5233.33	8.5|5215.21
CNV		1382	1385	1381	1381	1381	1382	**1381**
CTD		393 695	389 011	395 936	388 548	388 466	388 697	**388 013**
Time		295.9 min	5 × 89 min	180 min	34.0 min	5 × 16.2 min	34.1 min	5 × 34.1 min
R1	600	54.5|18 381.28	54.5|18 252.13	54.5|18 781.79	54.5|18 194.38	54.5|18 186.24	54.50|18 111.58	**54.5|18 023.18**
R2	600	11.0|12 847.31	11.0|12 808.59	11.0|12 804.60	**11.0|12 319.75**	11.0|12 330.49	11.00|12 385.20	11.0|12 352.38
C1	600	57.4|14 103.61	57.4|14 106.03	**57.3|14 236.86**	57.4|14 054.70	57.4|14 067.34	57.40|14 078.12	57.4|14 058.46
C2	600	17.4|7725.86	17.5|7632.37	17.4|7729.80	17.4|7601.94	17.4|7605.07	17.40|7635.68	**17.4|7594.41**
RC1	600	55.0|16 274.17	55.0|16 266.14	55.0|16 767.72	55.0|16 179.39	55.0|16 183.95	55.00|16 156.47	**55.0|16 097.05**
RC2	600	11.5|10 935.91	11.7|10 990.85	11.4|11 311.81	11.4|10 591.87	**11.4|10 586.14**	11.50|10 568.26	11.5|10 511.86
CNV		2068	2071	**2066**	2067	2067	2068	2068
CTD		802 681	800 797	**816 326**	789 420	789 592	789 353	786 373
Time		646.9 min	5 × 105 min	270 min	80.4 min	5 × 25.3 min	99.4 min	5 × 99.4 min
R1	800	72.8|31 755.57	72.8|31 797.42	72.8|32 734.57	72.8|31 486.74	72.8|31 492.81	72.80|31 385.55	**72.8|31 311.38**
R2	800	15.0|20 601.22	15.0|20 651.81	15.0|20 618.21	**15.0|19 873.04**	15.0|19 914.97	15.00|19 995.82	15.0|19 933.39
C1	800	75.4|25 026.42	75.4|25 093.38	75.2|25 911.44	75.3|24 990.42	**75.2|25 151.83**	75.50|24 898.96	75.4|24 876.38
C2	800	23.4|11 598.81	23.5|11 569.39	23.4|11 835.72	23.4|11 438.52	23.4|11 447.27	23.38|11 474.16	**23.3|11 475.05**
RC1	800	72.0|31 267.84	72.0|33 170.01	72.0|33 795.61	72.0|31 020.22	72.0|31 278.28	72.0|29 655.52	**72.0|29 404.32**
RC2	800	15.6|16 992.79	15.8|16 852.38	15.5|17 536.54	**15.4|16 438.90**	15.4|16 484.31	15.50|16 513.49	15.4|16 495.82
CNV		2742	2745	2739	2739	**2738**	2741.8	2739
CTD		1 372 427	1 391 344	1 424 321	1 352 478	**1 357 695**	1 339 235	1 334 963
Time		1269.4 min	5 × 129 min	360 min	126.8 min	5 × 27.6 min	215 min	5 × 215 min
R1	1000	91.9|48 827.23	91.9|49 702.32	91.9|51 414.26	91.9|48 287.98	91.9|48 369.71	91.90|47 928.13	**91.9|47 759.66**
R2	1000	19.0|30 164.60	19.0|30 495.26	19.0|30 804.79	**19.0|28 913.40**	19.0|29 003.42	19.00|29 159.07	19.0|29 076.45
C1	1000	94.4|41 699.32	94.3|41 783.27	94.2|43 111.60	94.1|41 683.29	94.1|41 748.60	94.42|41 550.55	**94.1|41 572.86**
C2	1000	29.3|16 589.74	29.5|16 657.06	29.3|16 810.22	29.1|16 498.61	29.1|16 534.36	28.90|16 723.59	**28.8|16 796.45**
RC1	1000	90.0|44 818.54	90.0|45 574.11	90.0|46 753.61	90.0|44 743.18	90.0|44 860.60	90.00|44 448.97	**90.0|44 333.40**
RC2	1000	18.3|25 064.88	18.5|25 470.33	18.4|25 588.52	18.3|23 939.62	18.3|24 055.31	18.24|24 209.03	**18.2|24 131.13**
CNV		3429	3432	3428	3424	3424	3424.6	**3420**
CTD		2 071 643	2 096 823	2 144 830	2 040 661	2 045 720	2 040 193	**2 036 700**
Time		1865.4 min	5 × 162 min	450 min	186.4 min	5 × 35.3 min	349 min	5 × 349 min
Processor		P4-3G	P4-2.8G	Opt-2.3G	P4-3G	Opt-2.4G	Xe-2.93G	

Table 8
Results on new large-scale PVRPTW, MDVRPTW, and SDVRPTW instances.

Inst	n	m			t	**PVRPTW**			**MDVRPTW**			**SDVRPTW**)		
		P	MD	SD		Avg 5	T (min)	Best 5	Avg 5	T (min)	Best 5	Avg 5	T (min)	Best 5
pr11a	360	24	10	11	4	21 120.94	61.74	20 937.29	6772.00	16.81	6720.71	9958.05	13.91	9924.11
pr12a	480	30	13	14	4	26 677.56	192.16	26 483.68	8259.57	30.00	8179.80	12 371.95	30.44	12 251.66
pr13a	600	38	16	17	4	31 909.12	297.03	31 808.00	9751.22	54.85	9667.20	14 562.53	44.93	14 491.25
pr14a	720	44	19	21	4	37 066.65	302.25	36 954.39	11 235.13	65.65	11 124.01	16 620.70	70.12	16 547.86
pr15a	840	50	22	25	4	41 847.30	301.05	41 699.07	13 078.26	132.44	13 013.97	19 283.90	111.09	19 090.19
pr16a	960	58	26	29	4	48 855.14	307.29	48 375.16	14 415.89	133.63	14 299.87	21 803.57	176.25	21 413.65
pr17a	360	22	7	8	6	28 889.82	65.28	28 818.04	6340.66	17.23	6304.30	10 581.02	12.20	10 547.07
pr18a	520	30	10	12	6	37 491.40	263.63	37 385.82	8381.71	44.25	8308.32	14 009.53	21.56	13 963.49
pr19a	700	38	13	16	6	49 103.78	300.39	48 993.72	10 734.60	74.42	10 677.61	18 998.48	95.29	18 855.51
pr20a	880	48	16	20	6	60 474.34	302.59	60 144.66	12 142.60	107.37	11 963.91	22 655.72	150.56	22 513.44
pr21a	420	22	4	6	12	54 562.68	213.11	54 257.26	6321.20	28.00	6260.53	13 775.90	16.22	13 758.32
pr22a	600	30	6	8	12	73 226.99	297.44	72 978.33	8047.87	76.05	7985.37	17 661.05	45.00	17 572.09
pr23a	780	38	8	10	12	91 424.98	300.02	90 951.34	9984.75	137.72	9937.43	21 974.90	97.06	21 793.32
pr24a	960	48	10	12	12	114 892.01	308.38	114 712.30	11 971.74	197.17	11 923.72	26 875.82	148.26	26 775.76
pr11b	360	18	8	9	4	16 102.27	86.18	15 992.20	4852.67	18.04	4839.44	8011.50	15.89	7962.22
pr12b	480	24	11	11	4	20 822.71	177.36	20 753.17	6084.33	29.09	6063.26	9566.13	33.45	9508.68
pr13b	600	30	14	14	4	25 050.30	291.83	24 972.94	7282.25	70.99	7254.17	11 609.39	74.81	11 562.67
pr14b	720	36	17	17	4	29 976.52	301.40	29 790.14	8796.77	98.92	8732.29	13 693.79	157.09	13 623.28
pr15b	840	48	20	20	4	41 715.58	300.02	41 609.04	10 496.39	129.48	10 439.72	15 589.83	191.21	15 437.52
pr16b	960	56	23	23	4	49 558.36	306.31	49 470.50	11 565.39	170.31	11 483.22	17 920.79	252.35	17 834.61
pr17b	360	18	6	6	6	23 138.63	94.09	22 989.05	4847.58	15.78	4806.01	8629.90	13.20	8562.99
pr18b	520	24	9	9	6	32 201.55	274.33	32 093.04	6555.95	39.45	6526.72	11 525.05	53.61	11 477.72
pr19b	700	32	12	12	6	42 467.74	300.53	42 332.28	8295.30	80.55	8227.25	14 918.54	92.15	14 894.65
pr20b	880	42	15	15	6	53 119.63	302.15	52 863.23	10 378.54	150.74	10 325.80	18 666.10	228.78	18 566.66
pr21b	420	18	4	4	12	43 195.88	261.51	43 098.26	4887.57	36.75	4866.57	11 309.41	27.17	11 246.57
pr22b	600	24	6	6	12	58 942.49	303.13	58 814.76	6537.15	73.28	6488.50	14 354.29	57.21	14 288.26
pr23b	780	30	7	8	12	74 755.07	302.99	74 357.84	8603.85	163.99	8523.41	17 635.56	125.08	17 576.39
pr24b	960	40	8	10	12	94 551.24	303.18	94 395.56	10 997.66	298.51	10 890.08	23 420.33	227.26	23 176.90
Avg. Gap & T (min)						+0.42%	254.19	+0.00%	+0.71%	88.98	+0.00%	+0.60 %	92.22	+0.00%
Processor							Xe-2.93G			Xe-2.93G			Xe-2.93G	

To save computation time, the termination criterion was reduced to $It_{NI} = 2500$ iterations without improvement, and a time limit of 5 h is imposed. On the PVRPTW instances, which are very large, the population size was divided by two to speed up the convergence.

- LZ: Two-phase ejections chains and iterated local search of Lim and Zhang [22]
- RTI: Arc-guided evolutionary algorithm of Repoussis et al. [18]

We indicate in boldface for each problem instance the best performing method. In the last two columns for each instance the previous best-known solution (BKS) ever reported in the literature, and the best solution obtained by HGSADC during all our experiments are given. New BKS produced by HGSADC are underlined. Finally, the last three lines provide average measures over all instances: the percentage of error relative to the previous BKS, the computation time, and the type of processor used by each author. For concision matters, we report only average results by group of instances for the PVRPTW instances of Pirkwieser and Raidl [41], the VRPTW instances of Solomon and Desrosiers [64] and Gehring and Homberger [13]. Calculation of the average deviation to BKS is also based on groups. Detailed results are available upon request. Finally, notice that results are presented in the format "Fleet Size|Distance" for VRPTW benchmarks.

HGSADC appears to be highly competitive in terms of solution quality for all the problem settings considered. The average computation time remains short, similar to other methods, and suitable for many operational decisions. The proposed approach outperforms all other algorithms in the literature for the PVRPTW, MDVRPTW, and SDVRPTW, including the parallel iterative UTS of Cordeau and Maischberger [38], which requires a large overall computational effort distributed on 64 processors. For the VRPTW, HGSADC is comparable to the hybrid genetic algorithm of Nagata et al. [19] in terms of solution quality, is less computationally efficient, but relies on less problem-tailored components, and does not necessitate dedicated route minimization procedures.

The average standard deviation, measured on the instances of Cordeau et al. [27], ranges between 0.17% for the MDVRPTW and 0.27% for the PVRPTW, thus illustrating the good stability of the algorithm. HGSADC performance appears to be higher on problems with tight time windows and short routes. This observation goes in accordance with Bent and Van Hentenryck [63], which showed that geometrical decomposition tends to perform better for this kind of instances.

These experiments retrieved or improved 105/109 BKS for the PVRPTW, MDVRPTW, and SDVRPTW, and 75/109 of those have been strictly improved. For the VRPTW, HGSADC retrieved or improved 292/356 BKS, and strictly improved 158/356. Five new BKS with one less vehicle have been produced on the large scale VRPTW instances. In particular, on the VRPTW instances of Pirkwieser and Raid [64], HGSADC found the same best solutions as Nagata et al. [19]. The new BKS on the instances of Pirkwieser and Raidl [41] and Gehring and Homberger [13] are presented in Appendix C.

5.3. Sensitivity analysis on method components

Addressing large-scale time-constrained VRP with the HGSADC methodology led to several challenges, which were answered in this paper by means of new procedures for neighbourhood evaluation and pruning, and problem decompositions. This section analyses the role of several of these components. We measure the impact of the decomposition phases, the contribution of infeasible solutions to the search, which required new move evaluation procedures, and the diversity and cost objective.

Table 9
Sensitivity analysis on the role of diversity and cost objective, time window-infeasible solutions, and decomposition phases.

Benchmark	No diversity objective			No TW infeasibility			No decomposition			HGSADC		
	Fleet (%)	Dist (%)	T (min)	Fleet	Dist (%)	T (min)	Fleet (%)	Dist (%)	T (min)	Fleet (%)	Dist (%)	T (min)
PVRPTW	–	+1.23	17.6	–	+0.59	26.0	–	+0.21	22.56	–	+0.17	33.2
MDVRPTW	–	+1.24	4.73	–	+0.80	5.79	–	+0.11	5.29	–	+0.10	6.49
SDVRPTW	–	+1.33	3.59	–	+0.67	4.84	–	+0.10	4.06	–	+0.12	5.48
PVRPTW new	–	+1.68	232	–	+0.97	252	–	+1.45	238	–	+0.42	254
MDVRPTW new	–	+3.91	62.9	–	+1.83	85.9	–	+0.52	78.3	–	+0.60	92.2
SDVRPTW new	–	+2.91	44.3	–	+1.56	74.2	–	+0.65	99.6	–	+0.71	89.9
VRPTW $n=200$	+0.00	+0.55	5.93	–	–	–	+0.00	+0.21	6.21	+0.00	+0.18	8.40
VRPTW $n=400$	+0.48	+0.53	27.9	–	–	–	+0.32	+0.22	29.1	+0.22	+0.15	34.1

Three versions of the algorithm were thus derived by removing in turn a different element from the method. The first does not rely on infeasible solutions with respect to time windows, setting high penalties to time-window violations, and relies on the "feasible" subpopulation only. The second does not apply decomposition phases. The last one uses a "traditional" evaluation of individuals driven exclusively by solution cost. Table 9 compares the average results on five runs, as an average deviation to the BKS, of these derived methods on various benchmark instances studied in this paper. The objective of fleet minimization being not straightforward to tackle without relying on TW-infeasible solutions, some instances were not considered in this case.

These experiments confirm the pertinence of the HGSADC framework for this class of problems, as the diversity and cost objective contributes largely to the performance of the proposed method. They also underline the major role of the time window-infeasible solutions, thus giving full meaning to the new move evaluation procedures.

Decomposition phases contribute significantly to the search performance on large PVRPTW instances, for which subproblems can involve up to 560 customers. This impact is less substantial on MDVRPTW and SDVRPTW instances, which present few customer visits (between 90 and 240) per resource (depot or vehicle type), and generate only small- or medium-size VRPTW sub-problems. In this latter case, VRPTW routes were already efficiently optimized by the algorithm, without the need for decomposition, solution improvements being more likely to come from combined route and assignment optimizations.

6. Conclusions

We presented a new Hybrid Genetic Search with Advanced Diversity Control to efficiently address a large class of VRPTW variants, including MDVRPTW, PVRPTW, and SDVRPTW. Several new features were introduced to efficiently evaluate and prune neighbourhoods, and decompose large instances. Their important contribution to the performance of the algorithm in terms of solution quality and computing efficiency, as well as that of the HGSADC methodology combining cost and contribution-to-diversity factors in evaluating and selecting individuals, has been demonstrated by extensive sensitivity analysis.

Comprehensive computational experiments and comparisons to state-of-the-art methods showed that the proposed algorithm performs impressively, in terms of both solution quality and computational efficiency, outperforming all current state-of-the-art methods for the considered problems, and producing numerous new best known solutions on classical literature benchmark instances.

Among the future developments we intend to undertake, there is the development of more advanced mixed structural and spatial decompositions, in order to decompose a PVRPTW into smaller PVRPTW subproblems thus providing the means to keep on working on service assignments during the decomposition phases. We also plan to keep on generalizing the method, progressing towards VRP variants with even more attributes and "rich" settings.

Acknowledgments

We thank the referees for their valuable comments. While working on this project, T.G. Crainic was the NSERC Industrial Research Chair, Collaborative Research and Development Grant, in Logistics Management, ESG UQAM. and Adjunct Professor with the Department of Computer Science and Operations Research, Université de Montréal, and the Department of Economics and Business Administration, Molde University College, Norway, M. Gendreau was the NSERC/Hydro-Québec Industrial Research Chair on the Stochastic Optimization of Electricity Generation, MAGI, École Polytechnique. and Adjunct Professor with the Department of Computer Science and Operations Research, Université de Montréal.

Partial funding for this project has been provided by the Champagne-Ardenne regional council in France, the Natural Sciences and Engineering Council of Canada (NSERC), through its Industrial Research Chair and Discovery Grant programs, by our partners CN, Rona, Alimentation Couche-Tard, la Fédération des producteurs de lait du Québec, the Ministry of Transportation of Québec, and by the Fonds québécois de la recherche sur la nature et les technologies (FQRNT) through its Team Research Project program.

Appendix A. Mathematical model for the generalized PVRPTW

Eqs. (A.1)–(A.13) introduce a mathematical formulation of the generalized periodic vehicle routing problem with time windows. For convenience, the depot node v_0 has been modeled by two nodes v_0 and v_{n+1} representing the origin and destination nodes, respectively. We also identify the set of customer vertices as $\mathcal{V}^{CST}=\mathcal{V}\backslash\{v_0,v_{n+1}\}$. The model relies on binary constants a_{pl}, equal to 1 if and only if period l belongs to pattern p. Binary decision variables x_{ijkl} take value 1 if and only if vehicle k in period l visits v_j immediately after v_i. Binary variables y_{ip} take value 1 if and only if customer i is assigned to pattern p. Finally, the continuous variables t_{ikl} stand for the start of service of customer i, when serviced by vehicle k during period l.

$$\text{Minimize} \quad \sum_{v_i\in\mathcal{V}}\sum_{v_j\in\mathcal{V}}\sum_{k=1}^{m}\sum_{l=1}^{t} c_{ijl}x_{ijkl} \tag{A.1}$$

$$\text{Subject to}: \quad \sum_{p \in L_i} y_{ip} = 1 \quad v_i \in \mathcal{V}^{CST} \tag{A.2}$$

$$\sum_{v_j \in \mathcal{V}} \sum_{k=1}^{m} x_{ijkl} - \sum_{p \in L_i} a_{pl} y_{ip} = 0 \quad v_i \in \mathcal{V}^{CST};\ l = 1 \ldots t \tag{A.3}$$

$$\sum_{v_j \in \mathcal{V} \setminus \{v_{n+1}\}} x_{jikl} - \sum_{v_j \in \mathcal{V} \setminus \{v_0\}} x_{ijkl} = 0 \quad v_i \in \mathcal{V}^{CST};\ k = 1 \ldots m;\ l = 1 \ldots t \tag{A.4}$$

$$\sum_{v_j \in \mathcal{V} \setminus \{v_0\}} x_{0jkl} = 1 \quad k = 1 \ldots m;\ l = 1 \ldots t \tag{A.5}$$

$$\sum_{v_j \in \mathcal{V} \setminus \{v_{n+1}\}} x_{j,n+1,kl} = 1 \quad k = 1 \ldots m;\ l = 1 \ldots t \tag{A.6}$$

$$\sum_{v_i \in \mathcal{V}} \sum_{v_j \in \mathcal{V}} q_i x_{ijkl} \le Q \quad k = 1 \ldots m;\ l = 1 \ldots t \tag{A.7}$$

$$x_{ijkl}(t_{ikl} + \delta_{ijl} + \tau_i - t_{jkl}) \le 0 \quad (v_i, v_j) \in \mathcal{V}^2;\ k = 1 \ldots m;\ l = 1 \ldots t \tag{A.8}$$

$$e_i \le t_{ikl} \le l_i \quad v_i \in \mathcal{V};\ k = 1 \ldots m;\ l = 1 \ldots t \tag{A.9}$$

$$t_{n+1,kl} - t_{0kl} \le D \quad k = 1 \ldots m;\ l = 1 \ldots t \tag{A.10}$$

$$x_{ijkl} \in \{0,1\} \quad (v_i, v_j) \in \mathcal{V}^2;\ k = 1 \ldots m;\ l = 1 \ldots t \tag{A.11}$$

$$y_{ip} \in \{0,1\} \quad v_i \in \mathcal{V};\ p \in L_i \tag{A.12}$$

$$t_{ikl} \in \Re^+ \quad v_i \in \mathcal{V}; k = 1 \ldots m;\ l = 1 \ldots t \tag{A.13}$$

Three main groups of constraints constitute the building blocks of the model. The first group, Constraints (A.2)–(A.3), corresponds to the assignment of customers to patterns, and its impact on routes. The next group of constraints (A.4)–(A.7) presents an underlying network flow structure to model the route choices, with flow conservation constraints (A.4), and capacity restrictions (A.7). Finally, the last group, Constraints (A.8)–(A.10), models the service time to customers, and enforces the temporal constraints (duration and time windows). Under the assumption that the graph does not admit a cycle with null total travel time, sub-tours are implicitly eliminated by Eq. (A.8). This constraint can also be linearized using big M values.

Appendix B. Proof of Proposition 1: minimum duration, time-warp use, and concatenation of sequences

When the starting date of a visit sequence is set, an optimal policy for minimizing duration and time-window infeasibility on the remaining vertices involves to service each customer as early as possible, and use time warp only upon a late arrival to reach the end of a customer's time window. To demonstrate Proposition 1, we prove by induction on the concatenation operator the stronger Proposition 2, which characterizes the minimum duration and time-warp use to service a sequence σ of visits, as a function of the service date t of the first sequence's vertex (i.e., the departure date when the first vertex is a depot). This proposition also provides the means to calculate the characteristic functions of the concatenation of two sequences from the functions of the separate sequences.

Proposition 2. *There exists a set of starting dates t that minimize both the time-warp use and the duration to service a sequence σ. This set is a segment, notated $\mathcal{T}^{MIN}(\sigma) = [E(\sigma), L(\sigma)]$. The minimum duration $D(\sigma)(t)$ and time-warp use $TW(\sigma)(t)$ as a function of t can be expressed as follows, where $D(\sigma)$ and $TW(\sigma)$ represent the minimum duration and time-warp use, respectively*:

$$D(\sigma)(t) = D(\sigma) + (E(\sigma) - t)^+ \tag{B.1}$$

$$TW(\sigma)(t) = TW(\sigma) + (t - L(\sigma))^+ \tag{B.2}$$

Furthermore, let $\sigma = (\sigma_i, \ldots, \sigma_j)$ and $\sigma' = (\sigma'_{i'}, \ldots, \sigma'_{j'})$ be two visit sequences, then the sequence $\sigma \oplus \sigma'$ is characterized by the following data:

$$D(\sigma \oplus \sigma') = D(\sigma) + D(\sigma') + \delta_{\sigma_j \sigma'_{i'}} + \Delta_{WT} \tag{B.3}$$

$$TW(\sigma \oplus \sigma') = TW(\sigma) + TW(\sigma') + \Delta_{TW} \tag{B.4}$$

$$E(\sigma \oplus \sigma') = \max\{E(\sigma') - \Delta, E(\sigma)\} - \Delta_{WT} \tag{B.5}$$

$$L(\sigma \oplus \sigma') = \min\{L(\sigma') - \Delta, L(\sigma)\} + \Delta_{TW} \tag{B.6}$$

where $\Delta = D(\sigma) - TW(\sigma) + \delta_{\sigma_j \sigma'_{i'}}$, $\Delta_{WT} = \max\{E(\sigma') - \Delta - L(\sigma), 0\}$ *and* $\Delta_{TW} = \max\{E(\sigma) + \Delta - L(\sigma'), 0\}$.

Proof. For a sequence σ^0 with a single service to v_i, $D(\sigma^0) = \tau_i$, $TW(\sigma^0) = 0$, $E(\sigma^0) = e_i$, $L(\sigma^0) = l_i$. A schedule starting at t means to wait if $t \le e_i$ or use a time warp if $t \ge l_i$, and then perform the service. Starting at t, the minimum duration is $D(\sigma)(t) = \tau_i + (e_i - t)^+$, while the minimum time-warp use is $TW(\sigma)(t) = (t - l_i)^+$, thus satisfying the statements of Proposition 2.

Let σ and σ' be two visit sequences with their characteristic functions $D(\sigma)(t)$, $TW(\sigma)(t)$, $D(\sigma')(t)$ and $TW(\sigma')(t)$. The minimum duration of a schedule starting at t for $\sigma \oplus \sigma'$ is the sum of the duration to process the sequence σ at t, reach the first customer of σ' at time $t + D(\sigma)(t) - TW(\sigma)(t) + \delta_{\sigma_j \sigma'_{i'}}$, and perform the service of σ':

$$\begin{aligned}
D(\sigma \oplus \sigma')(t) &= D(\sigma)(t) + \delta_{\sigma_j \sigma'_{i'}} + D(\sigma')(t + D(\sigma)(t) - TW(\sigma)(t) + \delta_{\sigma_j \sigma'_{i'}}) \\
&= D(\sigma) + \max\{E(\sigma) - t, 0\} + \delta_{\sigma_j \sigma'_{i'}} + D(\sigma') \\
&\quad + \max\{E(\sigma') - t - \Delta + \max\{t - L(\sigma), 0\} \\
&\quad - \max\{E(\sigma) - t, 0\}, 0\} = D(\sigma) + \delta_{\sigma_j \sigma'_{i'}} + D(\sigma') \\
&\quad + \max\{E(\sigma') - \Delta - L(\sigma), 0, E(\sigma') - \Delta - t, E(\sigma) - t\} \\
&= D(\sigma) + \delta_{\sigma_j \sigma'_{i'}} + D(\sigma') + \max\{\delta_{WT}, \max\{E(\sigma') - \Delta, E(\sigma)\} - t\} \\
&= D(\sigma) + \delta_{\sigma_j \sigma'_{i'}} + D(\sigma') + \delta_{WT} + \max\{0, \max\{E(\sigma') \\
&\quad - \Delta, E(\sigma)\} - \delta_{WT} - t\} = D(\sigma \oplus \sigma') + \max\{0, E(\sigma \oplus \sigma') - t\}
\end{aligned} \tag{B.7}$$

The function profile of Proposition (2) and the values of $D(\sigma \oplus \sigma')$ and $E(\sigma \oplus \sigma')$ are thus respected. The minimum time-warp use of a schedule starting at t can also be calculated in a similar manner as follows:

$$\begin{aligned}
TW(\sigma \oplus \sigma')(t) &= TW(\sigma)(t) + TW(\sigma')(t + D(\sigma)(t) - TW(\sigma)(t) + \delta_{\sigma_j \sigma'_{i'}}) \\
&= TW(\sigma) + \max\{t - L(\sigma), 0\} + TW(\sigma') \\
&\quad + \max\{t + \Delta - L(\sigma') + \max\{E(\sigma) - t, 0\} \\
&\quad - \max\{t - L(\sigma), 0\}, 0\} \\
&= TW(\sigma) + TW(\sigma') + \max\{E(\sigma) + \Delta - L(\sigma'), 0, t \\
&\quad + \Delta - L(\sigma'), t - L(\sigma)\} \\
&= TW(\sigma) + TW(\sigma') + \max\{\delta_{TW}, t \\
&\quad + \max\{\Delta - L(\sigma'), -L(\sigma)\}\} \\
&= TW(\sigma) + TW(\sigma') + \delta_{TW} + \max\{0, t - (\min\{L(\sigma') \\
&\quad - \Delta, L(\sigma)\} + \delta_{TW})\} \\
&= TW(\sigma \oplus \sigma') + \max\{0, t - L(\sigma \oplus \sigma')\}
\end{aligned} \tag{B.8}$$

Again, the profile of $TW(\sigma \oplus \sigma')(t)$, and the values of $TW(\sigma \oplus \sigma')$ and $L(\sigma \oplus \sigma')$ of (2) are correct. It only remains to show that $E(\sigma \oplus \sigma') \le L(\sigma \oplus \sigma')$, which is equivalent to Eq. B.9:

$$\begin{aligned}
L(\sigma \oplus \sigma') - E(\sigma \oplus \sigma') &= \min\{L(\sigma') - \Delta, L(\sigma)\} - \max\{E(\sigma') - \Delta, E(\sigma)\} \\
&\quad + \max\{E(\sigma') - \Delta - L(\sigma), 0\} + \max\{E(\sigma) + \Delta - L(\sigma'), 0\} \ge 0
\end{aligned} \tag{B.9}$$

If $\delta_{WT} = 0$, then $L(\sigma \oplus \sigma') - E(\sigma \oplus \sigma') \ge \min\{L(\sigma') - \Delta, L(\sigma)\} - \min\{L(\sigma') - \Delta, L(\sigma)\} \ge 0$.

Table C1
HGSADC best solutions on Pirkwieser and Raidl [41] PVRPTW instances. Distances truncated to the first digit.

#	T4-R1	T4-C1	T4-RC1	T6-R1	T6-C1	T6-RC1	T8-R1	T8-C1	T8-RC1
1	**4082.0**	2907.4	3956.4	**5376.1**	**3981.2**	**5781.5**	**6471.3**	**4679.1**	**6847.2**
2	**3724.3**	**2882.9**	**3755.7**	**5201.6**	3841.7	**5333.3**	**6097.9**	4933.3	**5763.3**
3	3153.1	**2734.5**	**3449.9**	**3940.5**	**3523.6**	**4273.1**	**4687.0**	**4664.0**	**5424.9**
4	**2566.0**	**2419.0**	**2991.5**	**3335.8**	**3206.3**	**4062.0**	**4355.8**	**4591.6**	**4929.5**
5	3638.9	2884.1	**3932.6**	**4272.9**	4052.1	**5227.1**	5476.5	**5134.2**	**6203.4**

Table C2
HGSADC best solutions on Gehring and Homberger [13] large-scale VRPTW instances.

n	#	**R1**		**R2**		**C1**		**C2**		**RC1**		**RC2**	
200	1	20	4784.11	4	4483.16	20	2704.57	6	1931.44	18	**3602.80**	6	3099.53
	2	18	**4040.60**	4	3621.20	18	2917.89	6	1863.16	18	**3249.05**	5	2825.24
	3	18	3381.96	4	2880.62	18	2707.35	6	1775.08	18	3008.33	4	2601.88
	4	18	3057.81	4	1981.30	18	2643.31	6	1703.43	18	2851.68	4	2038.56
	5	18	4107.86	4	3366.79	20	2702.05	6	**1878.85**	18	**3371.00**	4	2911.46
	6	18	**3583.14**	4	2913.03	20	2701.04	6	1857.35	18	**3324.80**	4	2873.12
	7	18	3150.11	4	2451.14	20	2701.04	6	1849.46	18	3189.32	4	2525.83
	8	18	2951.99	4	1849.87	19	2775.48	6	1820.53	18	3083.93	4	**2292.53**
	9	18	**3760.58**	4	3092.04	18	2687.83	6	1830.05	18	3081.13	4	2175.04
	10	18	3301.18	4	2654.97	18	2643.55	6	1806.58	18	**3000.30**	4	2015.61
400	1	40	**10 372.31**	8	9210.15	40	7152.06	12	4116.14	36	**8576.97**	11	6682.37
	2	36	**8926.70**	8	7606.75	36	**7686.38**	12	3930.05	36	**7905.66**	9	**6191.24**
	3	36	**7821.95**	8	**5911.07**	36	**7060.67**	11	**4018.02**	36	**7540.59**	8	4930.84
	4	36	**7282.78**	8	4241.47	36	**6803.24**	11	**3702.49**	36	**7310.35**	8	3631.01
	5	36	**9246.63**	8	7132.14	40	7152.06	12	3938.69	36	**8185.21**	9	5893.84
	6	36	**8387.62**	8	**6125.82**	40	7153.45	12	3875.94	36	**8177.80**	8	5766.61
	7	36	**7641.22**	8	**5018.53**	39	**7417.92**	12	3894.16	36	**7957.64**	8	5360.34
	8	36	**7275.13**	8	4018.01	37	**7349.01**	**11**	**4303.69**	36	**7797.02**	8	4799.02
	9	36	**8719.19**	8	**6400.10**	36	**7043.74**	12	3865.65	36	**7752.77**	8	4551.81
	10	36	**8113.93**	8	5805.87	36	6860.63	11	**3827.15**	36	**7609.21**	8	**4278.61**
600	1	59	**21 408.13**	11	**18 206.80**	60	**14 095.64**	18	7774.16	55	**17 118.70**	14	13 368.77
	2	54	**18 863.43**	11	14 807.67	56	14 163.31	17	**8273.78**	55	**16 044.93**	12	**11 555.51**
	3	54	**17 040.40**	11	11 200.10	56	13 778.75	17	**7523.12**	55	**15 273.98**	11	**9444.99**
	4	54	**15 819.62**	11	**8029.37**	56	13 563.17	17	6911.35	55	**14 839.61**	11	**7076.49**
	5	54	**19 771.90**	11	**15 098.49**	60	**14 085.72**	18	7575.20	55	**16 693.26**	11	13 138.99
	6	54	**18 041.87**	11	12 506.57	60	14 089.66	18	**7471.17**	55	**16 632.03**	11	11 977.46
	7	54	**16 615.13**	11	**10 066.34**	58	14 848.38	18	**7512.07**	55	**16 145.64**	11	10 779.24
	8	54	**15 696.58**	11	7609.96	56	**14 429.48**	17	**7547.67**	55	**15 978.70**	11	10 009.46
	9	54	**18 708.67**	11	13 483.92	56	**13 693.42**	17	**8015.73**	55	**15 922.60**	11	9583.65
	10	54	**17 801.43**	11	12 279.01	56	13 637.34	17	**7255.69**	55	**15 740.26**	11	9078.64
800	1	80	36 860.69	15	**28 117.94**	80	**25 184.39**	24	11 662.08	72	**31 710.68**	18	**21 014.85**
	2	72	**32 598.51**	15	22 811.82	74	25 430.07	23	12 378.53	72	**29 034.99**	16	18 197.14
	3	72	**29 506.45**	15	17 741.68	72	**24 278.18**	23	**11 410.69**	72	**27 905.64**	15	14 467.00
	4	72	**27 931.57**	15	13 219.06	72	**23 841.11**	**22**	**11 154.40**	72	**26 875.90**	15	**11 006.56**
	5	72	**33 861.43**	15	24 350.52	80	**25 166.28**	24	11 425.23	72	**30 277.12**	15	19 147.21
	6	72	**31 154.87**	15	20 480.79	80	**25 160.85**	24	**11 347.35**	72	**30 262.33**	15	18 160.91
	7	72	**29 010.78**	15	16 697.82	78	**25 845.05**	24	**11 370.84**	72	**29 862.44**	15	16 852.17
	8	72	**27 766.11**	15	12 748.16	74	**25 293.09**	23	**11 292.10**	72	**29 194.16**	15	15 808.99
	9	72	**32 629.99**	15	22 423.76	72	**24 389.50**	23	**11 645.22**	72	**28 978.35**	15	15 390.38
	10	72	**31 187.35**	15	20 459.29	72	**24 090.10**	23	**10 981.00**	72	**28 797.79**	15	14 454.62
1000	1	100	53 657.99	19	42 317.54	100	**42 478.95**	30	16 879.24	90	**46 272.07**	20	30 343.11
	2	91	**49 105.21**	19	33 567.91	90	**42 300.76**	29	17 130.76	90	**44 129.42**	**18**	**26 327.92**
	3	91	**45 237.29**	19	25 053.80	90	**40 239.23**	**28**	**16 886.80**	90	**42 487.54**	18	20 053.78
	4	91	**42 845.99**	19	18 039.77	90	**39 501.23**	28	**15 656.75**	90	**41 613.58**	18	15 747.13
	5	91	**51 869.67**	19	36 446.65	100	**42 469.18**	30	**16 561.29**	90	**45 564.81**	18	27 237.68
	6	91	**47 849.05**	19	30 223.14	100	**42 471.28**	29	**16 951.39**	90	**45 303.67**	18	26 986.30
	7	91	**44 525.53**	19	23 452.85	98	42 873.78	**29**	**18 162.46**	90	**44 903.80**	18	25 295.67
	8	91	**42 597.89**	19	17 602.31	93	**42 220.24**	**28**	**16 577.32**	90	**44 366.01**	18	23 787.26
	9	91	**50 490.49**	19	33 231.28	90	**40 570.60**	29	16 432.53	90	**44 280.84**	18	23 116.15
	10	91	**48 578.49**	19	30 598.69	90	**39 933.06**	28	**15 944.72**	90	**43 896.78**	18	22 076.90

Otherwise, if $\delta_{TW}=0$, then $L(\sigma\oplus\sigma')-E(\sigma\oplus\sigma') \geq \min\{L(\sigma')-\Delta, L(\sigma)\}-\min\{E(\sigma')-\Delta, L(\sigma')-\Delta\} \geq 0$.

Finally, if $\delta_{WT}=\delta_{TW}=0$, then $\min\{L(\sigma')-\Delta, L(\sigma)\}-\min\{E(\sigma')-\Delta, E(\sigma)\} \geq 0$, as $L(\sigma')-\Delta \geq E(\sigma')-\Delta$; $L(\sigma) \geq E(\sigma)$; $L(\sigma')-\Delta \geq E(\sigma)$ because of $\delta_{TW}=0$; and $L(\sigma) \geq E(\sigma')-\Delta$ following from $\delta_{WT}=0$.

All the statements of Proposition 2 are thus proven by induction on the concatenation operation, and Proposition 1 follows. □

Appendix C. Best solutions found on VRPTW and PVRPTW instances

Tables C1 and C2 present the best solutions found by HGSADC during all the experiments on the PVRPTW instances of Pirkwieser and Raidl [41], and the VRPTW instances of Gehring and Homberger [13]. New best known solutions are indicated in boldface.

References

[1] Golden B, Raghavan S, Wasil E, editors. The vehicle routing problem: latest advances and new challenges. New York: Springer; 2008.
[2] Bräysy O, Gendreau M. Vehicle routing problem with time windows, part I: route construction and local search algorithms. Transportation Science 2005;39(1):104–18.
[3] Bräysy O, Gendreau M. Vehicle routing problem with time windows, part II: metaheuristics. Transportation Science 2005;39(1):119–39.
[4] Francis P, Smilowitz K, Tzur M. The period vehicle routing problem and its extensions. In: Golden B, Raghavan S, Wasil E, editors. The vehicle routing problem: latest advances and new challenges. Springer; 2008. p. 73–102.
[5] Gendreau M, Potvin J-Y, Bräysy O, Hasle G, Løkketangen A. Metaheuristics for the vehicle routing problem and its extensions: a categorized bibliography. In: Golden B, Raghavan S, Wasil E, editors. The vehicle routing problem: latest advances and new challenges. Springer; 2008. p. 143–69.
[6] Gendreau M, Tarantilis CD. Solving large-scale vehicle routing problems with time windows: the state-of-the-art. Technical report 2010-04, CIRRELT, Montreal, QC, Canada; 2010.
[7] Crainic TG, Crisan GC, Gendreau M, Lahrichi N, Rei W. Multi-thread integrative cooperative optimization for rich combinatorial problems. In: The 12th international workshop on nature inspired distributed computing—NIDISC'09, 25–29 May, Rome; 2009. CD-ROM.
[8] Hartl RF, Hasle G, Janssens GK. Special issue on rich vehicle routing problems. Central European Journal of Operations Research 2006;14(2):103–4.
[9] Vidal T, Crainic TG, Gendreau M, Lahrichi N, Rei W. A hybrid genetic algorithm for multi-depot and periodic vehicle routing problems. Operations Research 2012;60(3):611–24.
[10] Cordeau JF, Gendreau M, Laporte G. A tabu search heuristic for periodic and multi-depot vehicle routing problems. Networks 1997;30(2):105–19.
[11] Cordeau JF, Laporte G. A tabu search algorithm for the site dependent vehicle routing problem with time windows. INFOR 2001;39(3):292–8.
[12] Solomon MM. Algorithms for the vehicle routing and scheduling problems with time window constraints. Operations Research 1987;35(2):254–65.
[13] Gehring H, Homberger J. A parallel hybrid evolutionary metaheuristic for the vehicle routing problem with time windows. In: Proceedings of EUROGEN99, University of Jyväskylä, Finland; 1999. p. 57–64.
[14] Alvarenga GB, Mateus GR, De Tomi G. A genetic and set partitioning two-phase approach for the vehicle routing problem with time windows. Computers & Operations Research 2007;34(6):1561–84.
[15] Labadi N, Prins C, Reghioui M. A memetic algorithm for the vehicle routing problem with time windows. RAIRO—Operations Research 2008;42(3):415–31.
[16] Mester D, Bräysy O. Active guided evolution strategies for large-scale vehicle routing problems with time windows. Computers & Operations Research 2005;32(6):1593–614.
[17] Hashimoto H, Yagiura M, Ibaraki T. An iterated local search algorithm for the time-dependent vehicle routing problem with time windows. Discrete Optimization 2008;5(2):434–56.
[18] Repoussis PP, Tarantilis CD, Ioannou G. Arc-guided evolutionary algorithm for the vehicle routing problem with time windows. IEEE Transactions on Evolutionary Computation 2009;13(3):624–47.
[19] Nagata Y, Bräysy O, Dullaert W. A penalty-based edge assembly memetic algorithm for the vehicle routing problem with time windows. Computers & Operations Research 2010;37(4):724–37.
[20] Pisinger D, Ropke S. A general heuristic for vehicle routing problems. Computers & Operations Research 2007;34(8):2403–35.
[21] Prescott-Gagnon E, Desaulniers G, Rousseau LM. A branch-and-price-based large neighborhood search algorithm for the vehicle routing problem with time windows. Networks 2009;54(4):190–204.
[22] Lim A, Zhang X. A two-stage heuristic with ejection pools and generalized ejection chains for the vehicle routing problem with time windows. INFORMS Journal on Computing 2007;19(3):443–57.
[23] Nagata Y, Bräysy O. A powerful route minimization heuristic for the vehicle routing problem with time windows. Operations Research Letters 2009;37(5):333–8.
[24] Le Bouthillier A, Crainic TG. A guided cooperative search for the vehicle routing problem with time windows. Intelligent Systems, IEEE 2005;20(4):36–42.
[25] Le Bouthillier A, Crainic TG. A cooperative parallel meta-heuristic for the vehicle routing problem with time windows. Computers & Operations Research 2005;32(7):1685–708.
[26] Ibaraki T, Imahori S, Nonobe K, Sobue K, Uno T, Yagiura M. An iterated local search algorithm for the vehicle routing problem with convex time penalty functions. Discrete Applied Mathematics 2008;156(11):2050–69.
[27] Cordeau JF, Laporte G, Mercier A. A unified tabu search heuristic for vehicle routing problems with time windows. Journal of the Operational Research Society 2001;52(8):928–36.
[28] Cordeau JF, Laporte G, Mercier A. Improved tabu search algorithm for the handling of route duration constraints in vehicle routing problems with time windows. Journal of the Operational Research Society 2004;55(5):542–6.
[29] Weigel D, Cao B. Applying GIS and OR techniques to solve Sears technician-dispatching and home-delivery problems. Interfaces 1999;29(1):112–30.
[30] Blakeley F, Bozkaya B, Cao B, Hall W, Knolmajer J. Optimizing periodic maintenance operations for Schindler Elevator Corporation. Interfaces 2003;33(1):67–79.
[31] Teixeira J, Antunes AP, De Sousa JP. Recyclable waste collection planning—a case study. European Journal of Operational Research 2004;158(3):543–54.
[32] Sahoo S, Kim S, Kim BI, Kraas B, Popov Jr. A. Routing optimization for waste management. Interfaces 2005;35(1):24–36.
[33] Golden B, Wasil E. Computerized vehicle routing in the soft drink industry. Operations Research 1987;35(1):6–17.
[34] Privé J, Renaud J, Boctor F, Laporte G. Solving a vehicle-routing problem arising in soft-drink distribution. Journal of the Operational Research Society 2005;57(9):1045–52.
[35] Jang W, Lim HH, Crowe T, Raskin G, Perkins TE. The missouri lottery optimizes its scheduling and routing to improve efficiency and balance. Interfaces 2006;36(4):302–13.
[36] Chiu HN, Lee YS, Chang JH. Two approaches to solving the multi-depot vehicle routing problem with time windows in a time-based logistics environment. Production Planning & Control 2006;17(5):480–93.
[37] Parthanadee P, Logendran R. Periodic product distribution from multi-depots under limited supplies. IIE Transactions 2006;38(11):1009–26.
[38] Cordeau JF, Maischberger M. A parallel iterated tabu search heuristic for vehicle routing problems. Computers & Operations Research 2012;39(9):2033–50.
[39] Pirkwieser S, Raidl GR. A variable neighborhood search for the periodic vehicle routing problem with time windows. In: Proceedings of the 9th EU/meeting on metaheuristics for logistics and vehicle routing, Troyes, France, 2008.
[40] Pirkwieser S, Raidl GR. A column generation approach for the periodic vehicle routing problem with time windows. In: Proceedings of the international network optimization conference 2009. Pisa, Italy; 2009.
[41] Pirkwieser S, Raidl GR. Multiple variable neighborhood search enriched with ILP techniques for the periodic vehicle routing problem with time windows. In: Proceedings of hybrid metaheuristics, HM 2009. Udine, Italy; 2009.
[42] Pirkwieser S, Raidl GR. Matheuristics for the periodic vehicle routing problem with time windows. In: Proceedings of matheuristics 2010. Vienna, Austria; 2010.
[43] Yu B, Yang ZZ. An ant colony optimization model: the period vehicle routing problem with time windows. Transportation Research Part E: logistics and Transportation Review 2011;47(2):166–81.
[44] Nguyen PK, Crainic TG, Toulouse M. A Hybrid genetic algorithm for the periodic vehicle routing problem with time windows. Technical report 2011-25, CIRRELT, Montreal, QC, Canada; 2011.
[45] Tamashiro H, Nakamura M, Okazaki T, Kang D. A tabu search approach combined with an extended saving method for multi-depot vehicle routing problems with time windows. Soft Computing 2010;15(1):31–9.
[46] Polacek M, Hartl RF, Doerner K, Reimann M. A variable neighborhood search for the multi depot vehicle routing problem with time windows. Journal of Heuristics 2004;10(6):613–27.
[47] Polacek M, Benkner S, Doerner KF, Hartl RF. A cooperative and adaptive variable neighborhood search for the multi depot vehicle routing problem with time windows. BuR—Business Research 2008;1(2):207–18.
[48] Ostertag A. Decomposition strategies for large scale multi depot vehicle routing problems. PhD thesis, Universität Wien; 2008.
[49] Belhaiza S. Hybrid variable neighborhood—tabu search algorithm for the site dependent vehicle routing problem with time windows. Technical report, GERAD, Montreal, Canada; 2010.
[50] Glover F, Hao JK. The case for strategic oscillation. Annals of Operations Research 2011;183(1):163–73.
[51] Nagata Y. Effective memetic algorithm for the vehicle routing problem with time windows: edge assembly crossover for the VRPTW. In: Proceedings of the seventh metaheuristics international conference, Montreal, Canada (on CD-ROM); 2007.
[52] Prins C. A simple and effective evolutionary algorithm for the vehicle routing problem. Computers & Operations Research 2004;31(12):1985–2002.
[53] Chu F, Labadi N, Prins C. A Scatter Search for the periodic capacitated arc routing problem. European Journal of Operational Research 2006;169(2):586–605.
[54] Prins C. Two memetic algorithms for heterogeneous fleet vehicle routing problems. Engineering Applications of Artificial Intelligence 2009;22(6):916–28.
[55] Johnson DS, McGeoch LA. The traveling salesman problem: a case study in local optimization. In: Aarts EHL, Lenstra JK, editors. Local search in combinatorial optimization. Princeton University Press; 1997. p. 215–310.
[56] Toth P, Vigo D. The granular tabu search and its application to the vehicle-routing problem. INFORMS Journal on Computing 2003;15(4):333–46.
[57] Ibaraki T, Imahori S, Kubo M, Masuda T, Uno T, Yagiura M. Effective local search algorithms for routing and scheduling problems with general time-window constraints. Transportation Science 2005;39(2):206–32.
[58] Kindervater GAP, Savelsbergh MWP. Vehicle routing: handling edge exchanges. In: Aarts EHL, Lenstra JK, editors. Local search in combinatorial optimization. Princeton University Press; 1997. p. 337–60.
[59] Irnich S. A unified modeling and solution framework for vehicle routing and local search-based metaheuristics. INFORMS Journal on Computing 2008;20(2):270–87.
[60] Vidal T, Crainic TG, Gendreau M, Prins C. A unifying view on timing problems and algorithms. Technical report 2011-43, CIRRELT, Montreal, QC, Canada; 2011.
[61] Taillard E. Parallel iterative search methods for vehicle routing problems. Networks 1993;23(8):661–73.
[62] Reimann M, Doerner KF, Hartl RF. D-Ants: savings based ants divide and conquer the vehicle routing problem. Computers & Operations Research 2004;31:563–91.
[63] Bent R, Van Hentenryck P. Spatial, temporal, and hybrid decompositions for large-scale vehicle routing with time windows. In: Proceedings of the 16th international conference on principles and practice of constraint programming, CP'10, St. Andrews, Scotland. Springer-Verlag; 2010. p. 99–113.
[64] Solomon MM, Desrosiers J. Time window constrained routing and scheduling problems. Transportation Science 1988;22(1):1–13.