Available online at www.sciencedirect.com

ScienceDirect

Transportation Research Procedia 10 (2015) 735 – 744

Transportation Research Procedia

www.elsevier.com/locate/procedia

18th Euro Working Group on Transportation, EWGT 2015, 14-16 July 2015, Delft, The Netherlands

Periodic capacitated vehicle routing for retail distribution of fuel oils

Pasquale Carotenuto [a,*], Stefano Giordani [b], Simone Massari [b], Fabrizio Vagaggini [b]

[a] *Consiglio Nazionale delle Ricerche - Istituto per le Applicazioni del Calcolo "M. Picone", via dei Taurini 19, 00185 Roma, Italy*
[b] *Università di Roma "Tor Vergata" – Dipartimento di Ingegneria dell'Impresa, via del Politecnico 1, 00133 Roma, Italy*

Abstract

In this paper we consider the final distribution of fuel oil from a storage depot to a set of petrol stations faced by an oil company, which has to decide the weekly replenishment plan for each station, and determine petrol station visiting sequences (vehicle routes) for each day of the week, assuming a fleet of homogeneous vehicles (tankers). The aim is to minimize the total distance travelled by tankers during the week, while loading tankers possibly near to their capacity in order to maximize the resource utilization. The problem is modelled as a generalization of the Periodic Vehicle Routing Problem (PVRP). Due to the large size of the real instances which the company has to deal with, we solve the problem heuristically. We propose a hybrid genetic algorithm that successfully address the problem inspired to a known hybrid genetic algorithm from the literature for the PVRP. However, the proposed algorithm adopts some techniques and features tailored for the particular fuel oil distribution problem, and it is specifically designed to deal with real instances derived from the fuel oil distribution in the European context that are profoundly different from the PVRP instances available from literature. The proposed algorithm is evaluated on a set of real case studies and on a set of randomly generated instances that hold the same characteristics of the former.

Peer-review under responsibility of Delft University of Technology

Keywords: Fuel Oil Distribution; Freight Transport;Transportation Planning; Vehicle Routing; Metaheuristics; Genetic Algorithm.

* Corresponding author. Tel.: +39-06-49270926; fax: +39-06-4404306.
E-mail address: carotenuto@iac.cnr.it

Peer-review under responsibility of Delft University of Technology
doi:10.1016/j.trpro.2015.09.027

1. Introduction

This paper deals with the problem of planning the final distribution of petrol products from storage depots to a set of petrol stations, addressed by the logistic department of a major European oil company. In such a context, many variables come into play: some of the most relevant are demand uncertainty and seasonality, agreements for exchange of products with other oil companies, contracts with carriers, etc.

In order to reduce the complexity of the problem, the whole decision process is subdivided by the oil company into three phases. In the first (strategic) phase each petrol station is assigned to a given depot from which it will be refuelled during the next medium-large term planning horizon. The second phase, the tactical-operational one, consists in defining the weekly delivery plan to refurnish the set of petrol stations assigned to a given depot, by determining the service days when each petrol station has to be served, along with the delivery amount of petrol products, and the routes for each specific day of the week that tankers have to perform in order to refurnish the petrol stations, considering some specific operational constraints.

As for the second phase, the real scenario faced by the oil company consists of a set of petrol stations that in general are not directly owned by the company. This implies that deciding how much and when to replenish the petrol stations are decisions that the oil company cannot make autonomously, but in accordance with the petrol stations' owners. Moreover, typically the budget available to a petrol station owner is very limited (specially for a very small petrol station), implying that the petrol order sent to the oil company by a petrol station often covers only its petrol demand for a couple of days. These operational conditions force the oil company to fulfill the estimated weekly petrol demand of a petrol station (typically for distinct product typologies) with a number of replenishments during the week (i.e., the weekly visit frequency), with the chance to select one out of a set of replenishment or visiting patterns established in accordance with the petrol station owner, where a visiting pattern specifies the visiting days along with the (possibly distinct) delivery petrol amounts for these days.

The oil company addresses this complex petrol distribution problem in two sub-phases. In the first one, let us say at tactical level, the oil company defines replenishment weekly plans for each petrol station by assuming for simplicity a single undifferentiated product, and determines petrol stations visiting sequences (vehicle routes) for each day of the week, assuming a fleet of homogeneous vehicles. The main aim at this phase is to minimize the total route length traveled by the vehicles during the considered week. In the second (operational) sub-phase, on a daily basis, the company plans in detail the routes minimizing the total route length, while considering all the operational issues and constraints, including those related to the vehicles characteristics (whose capacity is typically subdivided into compartments), to the distinct products to be delivered (e.g., gasoline and diesel fuel), and fulfilling the actual replenishment demands of petrol stations for the specific day.

In this paper, we address the above tactical problem faced by the oil company by modeling it as a generalization of the Periodic Vehicle Routing Problem (PVRP). Due to the large size of the real instances which the company has to deal with, we solve the problem heuristically. We propose a hybrid genetic algorithm that successfully addresses the problem. The proposed algorithm is evaluated on a set of problem instances derived from real case studies of a European oil distribution company, and on a set of randomly generated instances that hold the same characteristics of the former. Such a kind of optimization tool can be used not only in the optimization process of a given oil distribution network, but also for performing scenario analyses and economic assessments simulating variations of existing networks (acquisition or disposal of petrol stations).

The paper is organized as follows. In Section 2 we survey the relevant literature, in Section 3 we formally define the problem addressed. The proposed algorithm is detailed in Section 4. Section 5 is devoted to the experimental analysis, and finally Section 6 gives some conclusions.

2. Literature review

The problem we address belongs to the class of multi-period petrol station replenishment problems (see, e.g., Cornillier *et al.*, 2008b), where the aim is to optimize the delivery of several petrol products to a set of petrol stations over a given planning horizon. They can be viewed as Inventory Routing Problems (IRP) with specific additional constraints such as the use of heterogeneous vehicle with compartments, also known as IRP in fuel delivery (see, e.g., Vidović *et al.*, 2014). Malépart *et al.* (2003) propose four heuristics for constructing replenishment plans over a horizon of several working days, where some petrol stations manage their own inventories sending their orders to vendor company whenever they want, and for other stations the inventory is managed by the vendor company who decides the replenishment plans. Ng *et al.* (2008) study two small petrol distribution networks in Hong Kong, proposing a model for simultaneously assigning trips to tankers and stations, assuming stations inventories being managed by the vendor. Cornillier *et al.* (2008b) propose a heuristic approach to solve the case where the number of stations on any given route is limited to two. Popović *et al.* (2012) propose a variable neighborhood search (VNS) heuristic for solving a multi-product multi-period IRP in fuel delivery with multi-compartment homogeneous vehicles, given a distinct deterministic petrol consumption for each fuel type and for each petrol station; they limit the number of stations per route to three. Vidović *et al.* (2014) extend this limit to four and propose a mixed integer formulation that can be solved at optimum by commercial solvers only for very small instances (10 petrol stations and 3 days); they also propose some heuristics for solving larger instances up to 50 petrol stations and a period of 5 days. The single-period case has been also studied (in this case the problem is no longer an IRP but a Vehicle Routing Problem (VRP) with special vehicles): Avella *et al.* (2004) and Cornillier *et al.* (2008a) study this case proposing exact and heuristic algorithms; Cornillier *et al.* (2009) propose two heuristics for the more general case with time windows constraints for the petrol delivering to petrol stations; the same authors extend this study by considering also multiple depots (Cornillier *et al.*, 2012).

The PVRP is a generalization of the capacitated VRP (see, e.g.: Toth and Vigo, 2002; Hoff *et al.*, 2010) that takes into account several planning days with customers that require service on multiple days during the planning period.

After the very early works that propose constructive heuristics for the PVRP, Chao *et al.* (1995) are the first to provide a two-phase heuristic that allows escaping from poor local optima, enhancing the exploration of the solution space, through the use of deteriorating moves and the relaxation of vehicle capacity. Cordeau *et al.* (1997) propose a tabu search heuristic to solve the PVRP and two variants (i.e., the Multi-Depot Periodic Vehicle Routing Problem (MDPVRP) and the periodic travelling salesman problem). Another tabu search heuristic is proposed by Angelelli and Speranza (2002) to solve a generalization of PVRP where vehicles can renew their capacity at some intermediate facilities. Drummond *et al.* (2001) propose an island-based parallel evolutionary algorithm which evolves individuals representing visit combinations and generates routes with a saving heuristic. Alegre *et al.* (2007) propose a scatter search procedure for periodic pick up of raw materials of auto parts, using a two-phase approach that assigns visit combinations and design routes for each period with a neighbourhood-based procedure. Pirkwieser and Raidl (2010) introduce an extension of VNS metaheuristics with multilevel refinement strategy particularly suitable for larger PVRP instances. A two-phase GRASP and path relinking metaheuristic is proposed by Pacheco *et al.* (2012). A significant contribution is given by Vidal *et al.* (2012) that propose a hybrid genetic algorithm to tackle the MDPVRP and PVRP with heterogeneous capacitated vehicles and constrained route duration. It combines the breadth exploration of evolutionary algorithms, the improvement capabilities of VNS, and an advanced population-diversity management mechanism which allows a broader access to reproduction, preserving at the same time the memory of the characteristics of good solutions represented by the elite individuals. More recently, Nguyen *et al.* (2014) develop a genetic algorithm for the variant of the PVRP with time windows (PVRPTW), while Rahimi-Vahed *et al.* (2015) use a modular heuristic algorithm to solve a particular application of the MDPVRP and the PVRP with the objective of determining the optimal fleet sizing, considering budget constraints in addition to standard constraints on vehicle capacity and route duration.

3. Problem definition

The PVRP addressed in this paper extends the Capacitated VRP (CVRP) which is defined as follows. Let $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ be a complete direct graph with $|\mathcal{N}| = n + 1$ nodes, with node $0 \in \mathcal{N}$ representing the depot where the product to be distributed to customers is stored, and the other nodes $i = 1, \ldots, n$ representing the customers. Each customer i is characterized by product demand q_i and serving time τ_i. Let arc $a_{ij} \in \mathcal{A}$ represents the direct-travel from customer (depot) i to customer (depot) j with travel cost (length) c_{ij}, and travel time t_{ij}. The product distribution to customers is done by a set of identical vehicles, each one of limited capacity $Q_{\max}$. Each vehicle performs a route, i.e., a cycle over $\mathcal{G}$ starting and ending at the depot node, and visiting a subset of the customer nodes; therefore, the total demand of the visited customers by a vehicle route cannot exceed the capacity $Q_{\max}$ of the vehicle. Moreover, the duration of a route, being the sum of the total route travel time and the total service time of the visited customers, cannot be greater than $T_{\max}$. The goal is to find a set of vehicle routes serving all the customers exactly once, such that vehicle-capacity and route-duration constraints are fulfilled, and the total travel cost (length) is minimized.

The PVRP extends the CVRP adding a time dimension and considering route planning to be performed over a planning horizon of p periods. Each customer i is characterized by a frequency f_i, representing the number of required visits during the planning horizon, and a list of possible visit-period combinations called (visiting) patterns. For each customer i, the demand in each visit-period is assumed to be fixed and equal to q_i with a total demand over the planning horizon equal to $q_i^{tot} = f_i\, q_i$. The PVRP aims at selecting a pattern for each customer and finding the feasible routes for each period to fulfil the selected visiting patterns, minimizing the total route travel cost (length) l.

We consider a generalization of the PVRP where the total demand q_i^{tot} of customer i can be also non-equally subdivided among the visit-periods: in this more general case, a pattern π_i of customer i is formed by a list of possible visit-period combinations and a related list of specific demand combinations; let Π_i be the set of possible patterns of customer i. We also extend the objective function in order to evaluate solutions not only on the basis of the total travelled distance but also with respect to the unused capacity of the vehicles. In particular, we assume that vehicles should be loaded with at least $Q_{\min} \leq Q_{\max}$ amount of goods (petrol product), otherwise we incur in a sort of penalty proportional to the waste capacity index $c = \sum_{r\in R} \max\left(0, Q_{\min} - \sum_{i\in\mathcal{N}(r)} q_i\right)^2 \Big/ \left(Q_{\max}\,|R|\right)$, where $\mathcal{N}(r)$ is the set of customers visited by route $r \in R$, with R being the set of routes performed by the vehicles. Consequently, the objective is to minimize the objective function $z = w_1 l + w_2 c$, where w_1 and w_2 are weight parameters.

4. The proposed hybrid genetic algorithm

The solution approach, called *Periodic Capacitated Vehicle Routing Problem Genetic Search* (PCVRPGS), we propose to solve the considered generalization of the PVRP, is similar to the *Hybrid Genetic Search with Adaptive Diversity Control* (HGSADC) algorithm provided by Vidal *et al*. (2012), which achieves excellent results on PVRP instances known in the literature. However, PCVRPGS adopts some techniques and features tailored for the particular tactical problem of periodical petrol distribution to petrol stations. The general scheme of the hybrid genetic algorithm we propose is reported next.

```
Algorithm PCVRPGS
Initialize subpopulations
while number of iterations without improvement < It_ni and time < MaxTime do
  Select parent solutions P1 and P2 through binary tournament
  Generate offspring C from P1 and P2 through crossover
  Improve C through education procedure with probability p_ed
  if C is infeasible then
    Insert C into infeasible subpopulation
    Repair C and insert C into feasible subpopulation with probability p_rep
  if C is feasible then
    Insert C into feasible subpopulation
  if maximum subpopulation size is reached then
    Select survivors
  Adjust penalty parameters for violating feasibility conditions
  if best solution not improved for It_div iterations then
    Diversify populations
end while
return best feasible solution
```

Differently from HGSADC, the algorithm proposed in this paper is designed to deal with real instances of the considered petrol distribution problem that are profoundly different from the PVRP instances available from the literature. First of all, in petrol distribution instances the ratio between the vehicle capacity and the average customer demand is very small (between 3 and 4). This means that feasible solutions are characterized by a large number of routes each one visiting few petrol stations: consequently, some solving techniques for the VRP are not effective for the problem under consideration. Moreover, algorithm HGSADC is designed to work with constant demand for each customer during the planning horizon, but this assumption appears to be inappropriate for the practical application we are considering in this paper, because there may be a strong variability in the amount demand of a customer between the different periods of the planning horizon. Working with different customer demands over the planning horizon, algorithm PCVRPGS is able to tackle with this more general case of the PVRP, that adds a further degree of complexity to the original problem and provides greater flexibility to the final decision maker.

Algorithm HGSADC considers a population of feasible and infeasible solutions (individuals) which are kept in two separate subpopulations. The algorithm selects two parent individuals from the entire population and combines them through *crossover*, creating a new solution (*offspring*) which is enhanced through local search procedures (*education* and *repair*). The individual yielded is then included into the correct subpopulation and evaluated on the basis of its fitness function. The reproductive cycle iterates from one generation to the next one, each time selecting the survivors, until a stopping criteria is met.

After having presented its general outline, we describe the proposed algorithm in detail, starting from the representation and evaluation of solutions (Section 4.1), and proceeding with the illustration of the basic algorithm components such as parent selection and crossover (Section 4.2), education and repair (Section 4.3), and population management (Section 4.4).

4.1. Solution representation and evaluation

The literature on meta-heuristics algorithms shows that a controlled exploration of unfeasible solutions can improve the search performance, facilitating the transition between two structurally different eligible solutions (Glover and Hao, 2011).

Let G be the set of the periods of the planning horizon, Let S be the solution space that includes both feasible and unfeasible individuals, the latter obtained by relaxing the constraints on vehicle capacity and on maximum route duration. Let $s \in S$ be a solution and $R(s)$ the set of the planned routes of s for the planning horizon. Each route $r \in R(s)$ is characterized by the (vehicle) load $q(r) = \sum_{i \in \mathcal{N}(r)} q_i^{g(r)}$ and the duration $t(r) = \sum_{i \in \mathcal{N}(r) \cup \{0\}} t_{i,succ(i,r)} + \sum_{i \in \mathcal{N}(r)} \tau_i$, where $\mathcal{N}(r)$ is the set of customers visited by route r starting at depot node 0 with $succ(i, r)$ being the successive node visited in route r after node $i \in \mathcal{N}(r)$, $g(r)$ is the planning period of route r, and q_i^g is the demand of customer i in period $g \in G$. Let ω_q and ω_t be the penalties for violating the load capacity constraint and the route duration constraint, respectively. The penalized cost of route r is the sum of the distance travelled and the cost for violating load capacity and/or route duration constraints in case of unfeasibility, and it is defined by the following expression:

$$\psi(r) = \sum_{i \in \mathcal{N}(r) \cup \{0\}} c_{i,succ(i,r)} + \omega_q \max\{0, q(r) - Q_{\max}\} + \omega_t \max\{0, t(r) - T_{\max}\}.$$

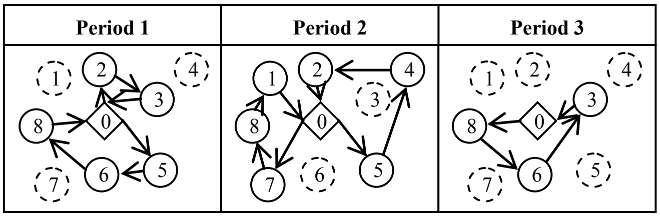
The penalized cost of solution s consists of the sum of penalized cost of each route of the solution, and is therefore equal to $\varphi(s) = \sum_{r \in R(s)} \psi(r)$.

Each solutions $s \in S$ is characterized by the assignment of exactly one demand pattern $\pi_i \in \Pi_i$ for each customer i, among its feasible demand patterns represented by set Π_i, and by the collection of generated routes. A (demand) pattern π_i corresponds to a particular set of periods of visits for customer i during the planning horizon, indicating also the quantity to be delivered in each period. Individual s are represented by 2 distinct chromosomes: i) the *pattern chromosome* that indicates the pattern π_i selected for each customer i; ii) the *giant tour chromosome* that contains a tour (route) for each period of the planning horizon, i.e. p sequences (i.e., sequence $\sigma_g(s)$ for each period $g = 1, \ldots, p$) of customers to be served, without trip delimiters and depot node. As an example, Figure 1 shows a solution s and the related pattern chromosome and giant tour chromosome representations.

Let $\theta_g(s)$ be the giant tour cost of planning period g of solution s, calculated as the total route distance travelled by a truck having infinite capacity. The "giant tour" cost of a solution s is then equal to the sum of all giant tour costs for all the period of the planning horizon, i.e., $\theta(s) = \sum_{g \in G} \theta_g(s)$.

Representing routes as giant tours allows to perform crossover operation in a simple and efficient way, working on permutations of customers, but requires the use of an efficient algorithm to find the optimal segmentation of a giant tour into routes to get the solution and its cost: PCVRPGS use the Split Algorithm of Prins (2004), which reduces the problem of extracting routes from a giant tour to a shortest path problem on an auxiliary acyclic graph.

Figure 1. A solution (individual) of the PVRP and its chromosome representations.



Pattern chromosome

Customer *i*	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Periods	{2}	{1, 2}	{1, 3}	{2}	{1, 2}	{1, 3}	{2}	{1, 2, 3}
Delivery quantities	{5}	{6, 3}	{4, 3}	{6}	{3, 1}	{4, 4}	{3}	{2, 1, 2}

Giant tour chromosome

Period *g*	**1**	**2**	**3**
Customer sequences	{5, 6, 8, 2, 3}	{5, 4, 2, 7, 8, 1}	{8, 6, 3}
Delivery quantities	{3, 4, 2, 6, 4}	{1, 6, 3, 3, 1, 5}	{2, 4, 3}

4.2. Parent selection and crossover

The offspring generation scheme starts with the selection of two parents P_1 and P_2 from which individual C is yielded through crossover using a binary tournament that randomly (with uniform probability) extracts 2 individuals from the entire population and keeps the solution with lower penalized cost with probability 0.8, whereas it selects the worst one with probability 0.2. The binary tournament is performed twice in order to select P_1 and P_2. The chance of combining feasible and unfeasible solutions allows extending the search to the limits of eligibility where we expect to find the best solutions, while the chance of selecting individuals with lower quality is an attempt to get diversification in the solution generation by exploiting the genetic information of the entire population.

The crossover procedure is composed by four basic steps. The first one defines the rules for the inheritance of the genetic material, through the extraction of two random integers between 0 and the number p of periods: being n_1 the lower extracted integer and n_2 the higher one, the inheritance rule states that n_1 periods will be entirely inherited at random from P_1 and form the set Λ_1 of periods, $n_2 - n_1$ periods from P_2 to form the set Λ_2, while the remaining periods will be inherited partly from P_1 and partly from P_2 to form the set Λ_{mix} periods.

In the second step the genetic material is transmitted from parent P_1 to offspring C. For each period $g \in \Lambda_1$, the customer visit sequence (giant tour) $\sigma_g(P_1)$ is entirely copied from P_1 to C without modification. For each $g \in \Lambda_{mix}$, two random cuts α_g and β_g are done on $\sigma_g(P_1)$: if $\alpha_g < \beta_g$, the subsequence of $\sigma_g(P_1)$ from position α_g to position β_g is transmitted to C; if $\alpha_g > \beta_g$, C will inherit from P_1 the subsequences of $\sigma_g(P_1)$ from α_g to the last position and from the first position to β_g; finally, if $\alpha_g = \beta_g$, with probability 0.5 the whole customer sequence $\sigma_g(P_1)$ is transmitted starting from the cut point (i.e., from α_g to the last position and from the first position to α_g).

The third step involves the transmission of genetic material from P_2 to offspring C. For each $g \in \Lambda_2 \cup \Lambda_{mix}$, we consider each customer $i \in \sigma_g(P_2)$ not yet inserted in the current sequence $\sigma_g(C)$ and copy it at the end of $\sigma_g(C)$ if i has at least one eligible pattern containing period g. If there are eligible patterns that also contain demand $q_i^g(P_2)$, then one of these pattern will be inherited into the chromosome pattern of C, otherwise an eligible pattern with another demand q'^g_i is transmitted in accordance with the eligible pattern.

Finally, the forth step completes the crossover through transmission to C of customers with service frequencies not yet satisfied. This is done by calculating the best eligible pattern (with related specific demand) and the best position in the tour through cheapest insertion technique for each customer i not completely served: this may require to modify the amount of demand $q_i^g(P_2)$ inherited from P_2 in order to guarantee the existence of at least one eligible pattern π_i for customer i. Note that, as a result of crossover, it is possible to generate customer's sequences that neither comes from genetic material of P_1 nor from that of P_2.

4.3. Education and repair

The *education* phase consists in a set of local search procedures used to improve the quality of a given solution. If the educated solution is unfeasible a *repair* procedure is possibly used to get a feasible solution. Two types of *local search* are used in the education phase: *Route_Improvement* (RI) and *Pattern_Improvement* (PI).

The education phase is applied with a probability p_{ed} to an individual s in the sequence RI-PI-RI. It is worth to notice that the education stage returns an educated solution s' not worse than s, i.e., $\varphi(s') \leq \varphi(s)$.

The RI procedure works on a solution s, trying to improve its routes by moving customers within the routes of each period (i.e., within each giant tour) with a set of moves. Specifically, given a giant tour ρ of a solution and denoting with $\mathcal{N}(\rho)$ the related set of visited customers, the following two moves M_1 and M_2 are applied over a customer $u \in \mathcal{N}(\rho)$ and one of his neighbour $v \in N(u)$:

- M_1: remove u and place it after v;
- M_2: swap u and v.

The *neighbourhood* $N(u)$ is defined as the set of $h \cdot |\mathcal{N}(\rho)|$ customers closest to u in terms of distance among those belonging to the same giant tour ρ of u. Parameter h is the *granularity index* (Toth and Vigo, 2003), and is a percentage of the number of customers visited by a tour.

Starting from a given solution s, RI applies moves M_1 and M_2 in random order for each period of the planning horizon. For each randomly chosen customer u of the giant tour of a given period and for each randomly chosen neighbour $v \in N(u)$ of u, RI evaluates the obtained neighbour solution s' of s, breaking the search and restarting it from solution s' if the latter is better than s. Solution improvement is evaluated by computing the change in cost of the (one or two) modified routes after the move execution. This process is repeated until all possible moves have been successively evaluated without improvement. Note that at the end of the local search improvement we get possibly a new giant tour for each period; therefore, we apply the Split Algorithm in order to compute the actual cost of the new local optimal solution obtained.

After the second execution of RI procedure, a 2-opt local search on the giant tour of each planning period is applied; finally, an exhaustive search is applied at the single route level if the number of visited customers is very small (e.g., no more than 5): we evaluate all possible sequences of the customers of each single route (all permutations) and select the shortest. To obtain all the possible permutations we use the "plain changes algorithm" (see, e.g., Sedgewick, 1977). The basic idea of this algorithm is that all permutations of a given number k of elements can be obtained from the sequence of permutations for $k-1$ elements by placing the element k into each possible position in each of the permutations of $k-1$ elements.

The PI procedure is executed with probability p_{PI}. It iterates in random order on each customer i and for each pattern $\pi_i \in \Pi_i$, looking for the best visit combinations along with the related customer demands. For each customer i, let us consider feasible patterns different from that (let us say π_i) contained in the current solution s. Let $\pi'_i \in \Pi_i$ be the new pattern considered, the following moves are performed until first improvement is achieved:

- if both π_i and π'_i contain period g and $q_i^g = q'^g_i$, do nothing;
- if both π_i and π'_i contain period g but $q_i^g \neq q'^g_i$, consider q'^g_i in place of q_i^g in the new solution;
- if only π_i contains period g, remove i from the giant tour of period g;
- if only π'_i contains period g, insert i with quantity q'^g_i into the giant tour of g through cheapest insertion.

This process is repeated until all possible moves have been successively evaluated without improvement.

The educated solution might be unfeasible: in that case the *repair* procedure is applied with probability p_{rep} to make the solution feasible. This procedure consists in temporarily multiplying the penalty parameters by 10 and reapplying the RI-PI-RI sequence. If the individual is still unfeasible, penalty parameters are multiplied by 10 again and the sequence is restarted until a feasible solution is obtained. Finally, the resulted individual is inserted into the feasible subpopulation without deleting the unfeasible one from the unfeasible subpopulation.

4.4. Population management

In order to initialize subpopulations, 4μ individuals are generated randomly assigning one pattern to each customer and creating customer visit sequences for each period in random order. Every solution is subjected to education, followed by repair in case of infeasibility (with probability $p_{\mathrm{rep}} = 0.5$), and inserted into correct subpopulation. At the end of initialization, one of the subpopulations may contain less than μ individuals.

The two (feasible and unfeasible) subpopulations are managed independently. Both subpopulations are programmed to hold a number of individuals between μ and $\mu + \lambda$, which represent the minimum and the maximum size of each subpopulation, respectively. Whenever an individual is yielded, it is added directly to the appropriate subpopulation based on its feasibility. When the maximum size $\mu + \lambda$ is reached in a subpopulation, the *survivor selection* mechanism is activated, that brings the population size back to μ discarding the worst λ individuals.

Penalty parameters are initially set to $\omega_q = \bar{c} / \bar{q}$ and $\omega_t = \bar{v}$, where $\bar{c}$ is the average (cost) distance between two customers, $\bar{q}$ is average amount of demanded product to be delivered, and $\bar{v}$ is the average vehicle speed. These parameters are dynamically adjusted in order to make the generation of naturally feasible solutions easier. Let ξ^{ref} be the reference percentage of naturally feasible solutions and ξ the same percentage evaluated over the last 100 iterations. Then, similarly to Vidal *et al.* (2012), the parameters are updated as follows:

- if $\xi \le \xi^{\mathrm{ref}} - 0.05$, then $\omega_q = \omega_q \cdot 1.2$, and $\omega_t = \omega_t \cdot 1.2$;
- if $\xi \ge \xi^{\mathrm{ref}} + 0.05$, then $\omega_q = \omega_q \cdot 0.85$, and $\omega_t = \omega_t \cdot 0.85$.

The *diversification* method is executed if the best solution is not improved for It_{div} iterations. It eliminates the $2\mu/3$ worst individuals of both the subpopulations. The aim of this step is to reintroduce a significant quantity of new genetic material when the population has lost most of its diversity.

When the stopping criteria is met, before returning the *"best solution",* a *post optimization* method is applied: we apply the same exhaustive search method used in the *education* operator to all routes of all individuals of the feasible population, when the number of visited customers of a route is very small. Then the best solution is updated.

5. Experimentation

Any evolutionary algorithm needs a calibration phase to set the right parameters, enabling the algorithm to perform at its best. There is not an optimal set of parameters right for any metaheuristic algorithm. In our study the calibration phase has been conducted with a meta-calibration approach: we evaluated the goodness of the outcome of the algorithm when changing the parameters of interest. The main parameters to be set are the population dimension μ and the granularity index h. Their values were fixed on the basis of the results obtained on a real instance with 194 customers (petrol stations) over a planning horizon of 6 periods (days), and with tanker (vehicle) capacity $Q_{\max}$ equal to 39 Kiloliters. The maximum runtime (*MaxTime*) has been set at 2, 5, 10, 15 and 20 minutes, and the maximum number of iterations without improvement $It_{\mathrm{ni}} = 2500$. For each maximum runtime we run 10 tests. Table 1 lists the results of the average solution values. Accordingly to the results we fix $h = 40\%$ and $\mu = 25$.

Table 1. h and μ calibration (results are total route length, l in Km).

	$h = 0.2$	$h = 0.3$	$h = 0.4$	$h = 0.5$	$\mu = 15$	$\mu = 20$	$\mu = 25$	$\mu = 30$	$\mu = 35$
MaxTime = 2	25745.17	26011.52	25214.93	26123.29	25940.78	25874.34	25214.93	25335.94	25516.76
5	25377.64	25476.42	25196.64	25665.93	25451.80	25390.66	25196.64	25244.77	25267.58
10	25078.78	25103.28	24787.95	25446.48	25140.76	25080.81	24787.95	24993.08	25007.32
15	24916.53	24887.52	24702.31	25255.73	24919.09	24955.41	24702.31	24838.91	24854.45
20	24846.27	24807.19	24621.46	25136.22	24951.44	24860.98	24621.46	24738.06	24786.58

Table 2. Algorithm parameters.

Parameter	Description	Value
μ	population dimension	25
λ	max dimension of a generation	40
p_{ed}	education probability	1.0
p_{rep}	repair probability	0.5
p_{PI}	pattern improvement probability	0.2
h	granularity index	40%
It_{ni}	number of iterations without improvement	2500
It_{div}	number of iterations before diversification	40% It_{ni}
ξ^{ref}	reference proportion for the penalty adjustment	20%

Other parameters are fixed according to Vidal *et al.* (2012). Table 2 lists the set of values used in our algorithm.

The second phase of experimentation is aimed to evaluate the tradeoff between the two terms of the objective function z (i.e., the total route length l and the waste capacity index c) with $Q_{\min}$ equal to 35 Kiloliters (i.e., approx.

90% of Q_{max}), and we run tests to evaluate the goodness of the solutions when changing the value of the weights w_1 and w_2. In particular, we consider $w_1 = (1 - w)/D_{tot}$ and $w_2 = w$, with D_{tot} being the total customer demand in order to make the magnitude of *l* similar to *c*, and test with $0 \leq w \leq 1$. The performance evaluation of algorithm PCVRPGS were done on three different real case studies with 49, 194 and 200 petrol stations, respectively, over a planning horizon of 6 periods. We experimented our algorithm with *MaxTime* = 2, 5, 10, 15, 20 minutes and weight *w* = 0, 0.25, 0.5, 0.75, 1. For each (*MaxTime*, *w*) combination we executed 10 runs: Tables 3–5 list the average results.

Table 3. Real case study 1: 49 petrol stations, 6 periods.

	total route length, *l* (Km)					waste capacity index, *c*				
MaxTime	*w* = 0	*w* = 0.25	*w* = 0.5	*w* = 0.75	*w* = 1	*w* = 0	*w* = 0.25	*w* = 0.5	*w* = 0.75	*w* = 1
2	3179.03	3211.03	3246.99	3280.74	3260.90	0.4786	0.2143	0.1957	0.2128	0.2068
5	3159.46	3222.93	3225.22	3243.76	3256.97	0.2413	0.2443	0.1540	0.1411	0.2027
10	3188.94	3214.30	3241.22	3276.38	3320.00	0.3531	0.2005	0.2161	0.1813	0.2605
15	3176.03	3188.77	3260.16	3262.66	3301.20	0.3431	0.1854	0.2263	0.1834	0.2650
20	3178.96	3238.74	3243.90	3207.52	3284.10	0.3716	0.2190	0.2005	0.1401	0.1783

Table 4. Real case study 2: 194 petrol stations, 6 periods.

	total route length *l* (Km)					waste capacity index, *c*				
MaxTime	*w* = 0	*w* = 0.25	*w* = 0.5	*w* = 0.75	*w* = 1	*w* = 0	*w* = 0.25	*w* = 0.5	*w* = 0.75	*w* = 1
2	25195.82	25194.11	25312.29	25681.21	25798.95	0.6655	0.4936	0.3966	0.4536	0.4461
5	25151.03	25082.46	25276.66	25377.42	25771.64	0.5890	0.4463	0.4208	0.4000	0.3865
10	24711.10	24943.21	25196.22	25325.41	25545.98	0.5029	0.3869	0.3633	0.3796	0.3453
15	24811.48	25172.62	25529.98	25405.36	25633.83	0.5421	0.4540	0.4272	0.3717	0.3749
20	24849.46	24925.99	25202.24	25699.01	25550.19	0.5623	0.3841	0.3672	0.4424	0.3966

Table 5. Real case study 3: 200 petrol stations, 6 periods.

	total route length *l* (Km)					waste capacity index, *c*				
MaxTime	*w* = 0	*w* = 0.25	*w* = 0.5	*w* = 0.75	*w* = 1	*w* = 0	*w* = 0.25	*w* = 0.5	*w* = 0.75	*w* = 1
2	24563.52	24654.79	24601.23	24636.43	24823.12	0.5707	0.3956	0.3843	0.4006	0.3607
5	24311.48	24323.93	24540.95	24647.66	24691.48	0.4238	0.3523	0.3660	0.3580	0.3629
10	24273.35	24389.88	24500.73	24549.11	24661.46	0.4362	0.3729	0.3383	0.3302	0.3575
15	24278.60	24391.29	24379.83	24452.74	24908.42	0.4341	0.3485	0.3495	0.3330	0.3596
20	24199.03	24292.77	24442.3	24522.94	24570.39	0.3574	0.2971	0.3238	0.3516	0.3301

The tables show how setting values for *w* higher than 0.25 leads to “unbalanced” solutions where the increase in distance is not compensated by a significant waste capacity index decrease.

Moreover, we conducted a comparison among the experimented results and the solution adopted by the logistic team of the oil company that calculates delivery/routing plans by using a strongly customized commercial decision support system and team’s expertise. For the real case study with 49 petrol stations, the delivery/routing plan adopted presents a total length equal to 3443.85 Km, while our algorithm always returned better solutions with an average improvement of 7.76%, considering only the total route length as objective function (i.e., *w* = 0).

Table 6. Results on random instances with *n* = 100, 150, 200 petrol stations and 6 periods; *MaxTime* = 10 min.

	total route length *l* (Km)					waste capacity index, *c*				
n	*w* = 0	*w* = 0.25	*w* = 0.5	*w* = 0.75	*w* = 1	*w* = 0	*w* = 0.25	*w* = 0.5	*w* = 0.75	*w* = 1
100	12357.55	12448.74	12463.93	12500.27	12555.79	0.3692	0.3469	0.3318	0.3033	0.2990
150	17888.49	18216.26	18059.67	18244.95	18173.64	0.4750	0.4686	0.3700	0.3797	0.4201
200	25241.05	25341.88	26038.86	25737.50	26203.90	0.5312	0.4265	0.4958	0.4603	0,5090

Finally, we experimented the proposed algorithm also on random realistic instances. In particular, we consider random instances with *n* = 100, 150, 200 petrol stations, obtained from the third real case study with 200 petrol stations: the first two instances are obtained by deleting at random 100 and 50 stations, respectively, and the third one with 200 petrol stations is obtained by rearranging at random the demand patterns. As for Q_{max} and Q_{min} we assume the same data of the real instances. For each weight *w* we performed 10 runs. Table 6 lists the results of the average solution values. Performance results on random instances show a similar trend of those for the real cases, with total route length that tends to increase while waste capacity index tends to decrease by increasing *w*.

6. Conclusions

In this work, we consider the planning of the final distribution of fuel oil from a storage depot to a set of petrol stations faced by a European oil company, which has to decide the weekly replenishment plan for each station, and determine petrol station visiting sequences (vehicle routes) for each day of the week. At tactical level, it is reasonable to assume a fleet of homogeneous vehicles (tankers) of given capacity and the problem is modeled as a particular periodic vehicle routing problem. We propose a hybrid genetic algorithm to solve this problem. Computational results both on real instances and random realistic instances show the effectiveness of the proposed approach. Further researches may be devoted to extend the model and the algorithm to the case with multiple depots.

References

Alegre, J., Laguna, M., Pacheco, J., 2007. Optimizing the periodic pick-up of raw materials for a manufacturer of auto parts. *European Journal of Operational Research* 179 (3), 736–746.

Angelelli, E., Speranza, M., 2002. The period vehicle routing problem with intermediate facilities. *European Journal of Operational Research*, 137(2), 233–247.

Avella, P., Boccia, M., Sforza, A., 2004. Solving a fuel delivery problem by heuristic and exact approaches. *European Journal of Operational Research* 152, 170–179

Chao, I.M., Golden, B.L., Wasil, E., 1995. An improved heuristic for the period vehicle routing problem. *Networks* 26(1), 25–44.

Cornillier, F., Boctor, F.F., Laporte, G., Renaud, J., 2008a. An exact algorithm for the petrol station replenishment problem. *Journal of the Operational Research Society* 59, 607–615.

Cornillier, F., Boctor, F.F., Laporte, G., Renaud, J., 2008b. A heuristic for the multi-period petrol station replenishment problem. *European Journal of Operational Research* 191, 295–305.

Cornillier, F., Laporte, G., Boctor, F.F., Renaud, J., 2009. The petrol station replenishment problem with time windows. *Computers and Operations Research* 36, 919–935.

Cornillier, F., Boctor, F.F., Renaud, J., 2012. Heuristics for the multi-depot petrol station replenishment problem with time windows. *European Journal of Operational Research* 220, 361–369.

Cordeau, J.F., Gendreau, M., Laporte, G., 1997. A tabu search heuristic for multi-depot vehicle routing problems. *Networks* 30, 105–119.

Drummond, L., Ochi, L., Vianna, D., 2001. An asynchronous parallel metaheuristic for the period vehicle routing problem. *Future Generation Computer Systems* 17(4), 379–386.

Glover, F., Hao, J.K., 2011. The case for strategic oscillation. *Annals of Operations Research* 183, 163–173.

Hoff, A., Andersson, H., Christiansen, M., Hasle, G., Løkketangen, A., 2010. Industrial aspects and literature survey: Fleet composition and routing. *Computer & Operations Research* 37(12), 2041-2061.

Malépart, V., Boctor, F.F., Renaud, J., Labilois, S., 2003. Nouvelles approches pour l'approvisionnement des stations d'essence. *Revue Française de Gestion Industrielle* 22, 15–31.

Ng, W., Leung, S., Lam, J., Pan, S., 2008. Petrol delivery tanker assignment and routing: a case study in Hong Kong. *Journal of the Operational Research Society* 59, 1191–1200.

Nguyen, P.K., Crainic, T.G., Toulouse, M., 2014. A hybrid generational genetic algorithm for the periodic vehicle routing problem with time windows. *Journal of Heuristics* 20(4), 383-416.

Pacheco, J., Alvarez, A., Garcia, I., Angel-Bello, F., 2012. Optimizing vehicle routes in a bakery company allowing flexibility in delivery dates. *Journal of the Operational Research Society* 63, 569-581.

Pirkwieser, S., Raidl, G., 2010. Multilevel variable neighborhood search for periodic routing problems. Evolutionary Computation in Combinatorial Optimization. *Lecture Notes in Computer Science*, vol. 6022. Springer-Verlag, Berlin/Heidelberg, pp. 226–238

Popović, D., Vidović, M., Radivojević, G., 2012. Variable Neighborhood Search heuristic for the Inventory Routing Problem in fuel delivery. *Expert Systems with Applications* 39, 13390–13398.

Prins, C., 2004. A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers and Operations Research* 31, 1985–2002.

Rahimi-Vahed, A., Crainic, T.G., Gendreau, M., Rei, W., 2015. Fleet-sizing for multi-depot and periodic vehicle routing problems using a modular heuristic algorithm. *Computers & Operations Research* 53, 9-23.

Sedgewick, R., 1977. Permutation generation methods, *ACM Comput. Surv.* 9(2), 137–164.

Toth, P., Vigo, D., 2002. *The Vehicle Routing Problem*. SIAM Monographs on Discrete Mathematics and Applications. SIAM, Philadelphia, PA.

Toth, P., Vigo, D., 2003. The granular tabu search and its application to the vehicle-routing problem. *INFORMS Journal of Computing* 15(4), 333–346.

Vidal, T., Crainic, T.G., Gendreau M., Lahrichi, N., Rei, W., 2012. A hybrid genetic algorithm for multi-depot and periodic vehicle routing problems. *Operations Research* 60(3), 611-624.

Vidović, M., Popović, D., Ratković, B., 2014. Mixed integer and heuristics model for the inventory routing problem in fuel delivery. *International Journal of Production Economics*,147, 593–604.