IMA Journal of Management Mathematics Advance Access published October 24, 2014

IMA Journal of Management Mathematics (2014) Page 1 of 16
doi:10.1093/imaman/dpu020

A general variable neighbourhood search for the multi-product inventory routing problem

Anis Mjirda*
LAMIH, Université de Valenciennes et du Hainaut-Cambrésis, France and Le Mont Houy, 59313 Valenciennes Cedex 9, France
*Corresponding author: anis.mjirda@univ-valenciennes.fr

Bassem Jarboui
MODELIS, Faculté des Sciences Economiques et de Gestion de Sfax, Tunisia and FSEGS, Route de l'aéroport km 4, Sfax 3018, Tunisia

Jelena Mladenović
Faculty of Mathematics, University of Belgrade, Beograd, Serbia

and

Christophe Wilbaut and Saïd Hanafi
LAMIH, Université de Valenciennes et du Hainaut-Cambrésis, France and Le Mont Houy, 59313 Valenciennes Cedex 9, France

[Received on 22 November 2013; accepted on 12 September 2014]

The inventory routing problem (IRP) deals with the study of the inter-relationship between two important activities in the supply chain: transportation of commodities and inventory management. In this paper, we study a variant of the multi-product IRP and we propose a general variable neighbourhood search metaheuristic for solving it. We present several neighbourhood structures based on the movement of quantities between routes and periods. Computational experiments show the efficiency of the proposed metaheuristic when compared with the state-of-the-art based on variable neighbourhood search.

Keywords: inventory; routing; metaheuristic; general variable neighbourhood search.

1. Introduction

During the last decades the operational research community has considered many optimization problems to bring efficient solutions to companies in order to maintain and to improve their competitiveness. This is, in particular, the case in the context of the supply chain management in which several activities can be identified. Practical problems incorporate more and more practical constraints encountered in the industry. In this work, we consider the combination of inventory and routing activities. As mentioned in some recent papers (Andersson *et al.*, 2010; Moin & Salhi, 2007), decisions related to inventory management and routing are critical issues when optimizing the supply chain, and this problematic can be considered as a rich and important field in itself in Andersson *et al.* (2010).

Generally speaking, the inventory routing problem (IRP) is a multi-period integrated distribution problem, in which three (inter-related) kinds of decision can be listed as showed in Coelho *et al.* (2012) and in Michel & Vanderbeck (2012): decide when a given customer will be visited during the planning horizon; decide the quantity to deliver (or picked-up) to a given customer; decide the routes of the vehicles for every period. Therefore, the IRP combines a vehicle routing problem, where the vehicle routes to visit customers or suppliers must be determined, and an inventory problem, concerning the management of the quantities of product to deliver or to collect. The objective is typically to minimize the transportation and the inventory costs over the planning horizon.

In this paper, we consider a variant of the multi-product version of the IRP with multiple suppliers and only one customer (many-to-one structure). This problem was originally introduced by Lee *et al.* (2003), and it was considered more recently by Moin *et al.* (2010). It can be viewed as a (part-supply) network composed by a depot, an assembly plant and a set of suppliers where each supplier provides a distinct product to the assembly plant. Formally, the situation can be represented by a complete graph $G = (V, A)$, where the set of nodes $V = N \cup \{0, n+1\}$ is composed by the depot (node (0)), the set $N = \{1, \ldots, n\}$ of suppliers and the assembly plant (node $(n+1)$). The set A of arcs is defined by: $A = \{(j, j') : j, j' \in V \times V\}$. A travelling cost $c_{jj'}$ is associated with each arc $(j, j') \in A$.

The planning horizon T is finite and it can be divided into τ time periods: $T = \{1, \ldots, \tau\}$. As mentioned previously, each supplier $i \in N$ provides a unique product requested by the assembly plant. In the following, we will denote by i a supplier or the corresponding product. The demand of the assembly plant for a given product i at a given period $t \in T$ is noted d_{it}. It is deterministic but variable over time. In the problem, it is assumed that products are ready at the supplier when a vehicle arrive. In addition, an inventory cost at the assembly plant is associated with each product $i \in N$ and it is noted h_i. However, no inventory cost are considered at the suppliers. To ensure the deliveries, an unlimited fleet of homogeneous vehicles with a capacity C is available. The vehicles are localized at the depot and collect the products from suppliers. After delivering the products at the assembly plant, the vehicles return to the depot. It can be observed that a supplier can be visited by more than one vehicle in a given period (split pick up). In addition, no backorder is allowed since it leads to excessive costs. The objective of the problem is to minimize the total costs to ensure the production at the assembly plant. More precisely, the objective integrates three different costs: the total holding cost h_i of the product i at the assembly plant, the total travelling cost $c_{jj'}$ by the vehicles during the period and an additional fixed cost noted F for using one vehicle in one period.

In this paper, we show the efficiency of the new neighbourhood structures used within general variable neighbourhood search (GVNS). It is assumed that the demand is taken in full or not at all. That corresponds, in fact, to a discretization of the continuous demand variable used in insertion and in exchange moves. It appears that our new GVNS outperform the recent VNS-based heuristic that could be considered as state-of-the-art for the multi-product IRP (see Mjirda *et al.*, 2013).

The rest of the paper is organized as follows. In Section 2, we present some related works concerned with the study of the IRP in general and to this variant in particular. Then, we provide, in Section 3, a description for the discrete quantity management and the proposed neighbourhood structures. Next, in Section 4, we describe our GVNS. Section 5 provides the numerical results, whereas Section 6 reports some conclusions and future works.

2. Related works

Some recent surveys on the IRP can be found in the literature (see Andersson *et al.*, 2010; Coelho *et al.*, 2012; Moin & Salhi, 2007). In these papers, the authors propose different classifications of the IRP variants according to different criteria, in particular the time horizon, the inventory policy, the fleet composition and size, the number of products, and so on. In particular, Coelho *et al.* (2012) propose the classification of the variants according to the time at which information on demand becomes known. With this criterion they distinguish the deterministic IRP in which the information is fully available to the decision maker from the stochastic IRP in which its probability distribution is known and from the dynamic IRP in which information is not fully known in advance, but is gradually revealed over time.

One of the first publication on the IRP is due to Bell *et al.* (1983). They consider a problem of industrial gases distribution with a finite time horizon and stochastic demand. To solve their problem they use knowledge about the number of customers that are normally visited on a route. This hypothesis simplifies the problem. They propose a branch-and-bound in which they use a Lagrangian relaxation. Some other papers related to the industrial gases distribution can be listed. For instance, Golden *et al.* (1984) consider a problem of propane distribution for a company with 3,000 customers and a stochastic demand. The aim of this work was to improve the planning system used by the company, which provides the next dates of visits based on the delivery history and the consumption estimation. They propose heuristic approaches to determine which customers to visit based on their degree of urgency. Then, the routes are determined by using the Clarke & Wright heuristic proposed by Clarke & Wright (1964). The obtained routes are associated to a set of vehicles and a bin-packing problem is solved. Campbell & Savelsbergh (2004) consider a deterministic variant for which they proposed a greedy randomized adaptive search procedure. Their approach is also based on the building of clusters of customers that can be served efficiently by a single vehicle. Then, they use these clusters to determine the quantities to deliver thanks to an allocation problem.

If we consider the applications of the IRP, we can also distinguish maritime applications and road-based applications. An overview on IRPs specifically dedicated to maritime applications can be found in Christiansen *et al.* (2007). Another recent review paper dedicated to shipping and routing in general and to maritime IRPs can also be found in Christiansen *et al.* (2013). Recent contributions in this particular area includes the works of Christiansen & Nygreen (2005), Stalhane *et al.* (2012), Hewitt *et al.* (2013) or Song & Furman (2013) among others. In Christiansen & Nygreen (2005), the authors introduce time window constraints in a combined ship scheduling and inventory management problem to deal with uncertainties in sailing time and time consumption at ports. They propose a column generation method to solve the problem. Stalhane *et al.* (2012) consider a large-scale liquefied natural gas ship routing and inventory model. They develop a multi-start local search heuristic providing solutions that are improved using a first-descent neighbourhood search and a branch-and-bound. In Hewitt *et al.* (2013), the authors propose a formulation of the maritime IRP with the aim to speed up the search for high-quality primal solutions in a branch-and-price framework. They also introduce specific local search techniques to enhance the process. Finally, in Song & Furman (2013), the authors introduce a flexible arc flow model and they propose an algorithmic framework to solve it by defining subproblems that are restricted versions of the original problem, by branch-and-cut. The subproblems are solved iteratively in a large neighbourhood search heuristic.

If we consider non-maritime applications, we can also found many references in the literature. For instance, Bard *et al.* (1998) present an IRP with satellite facilities that correspond to intermediate nodes where vehicles can pick up products. A route for each vehicle starts at the depot, visits a set of customers and when the vehicle is empty, it visits a facility in order to collect products. The solution of the problem is achieved in two phases. In the first phase, they group the customers to be visited during the planning horizon into clusters and in the second phase, they propose a local search method to solve a vehicle routing problem with satellite facilities. Bertazzi *et al.* (2002) analyse and compare two variants of the IRP with different objective functions: the minimization of the transportation cost and/or the holding cost into the retailer and/or into suppliers. They use a heuristic for a simplified problem with a single vehicle and a single product. As observed in Andersson *et al.* (2010), the first papers on the IRP with road-based applications and considering arc flow formulations were more recent. In particular, Savelsvergh & Song (2007, 2008) introduce the IRP with *continuous moves*. In this variant, some customers cannot be served using out-and-back tours, and some routes cannot be done in only 1 day. The authors propose three heuristics in Savelsvergh & Song (2007) that define the routes and the quantities to deliver. They also propose mixed integer programming (MIP) model in order to refine the quantities to deliver. Then, they propose in Savelsvergh & Song (2008) a multi-commodity flow model on which they applied some reductions, and a branch-and-cut framework to improve the previous results.

In this paper, we consider variable neighbourhood search (VNS) introduced in Mladenović & Hansen (1997) to solve a multi-product IRP. VNS is a metaheuristic based on the systematic exploration of different neighbourhood structures, within a local search routine (see Hansen *et al.*, 2010). VNS consist of a stochastic part (the shaking phase) and a deterministic part (the Local Search). It combines three basic steps: a local search phase to converge to efficient solutions, a shaking phase in order to escape from local optima visited during the search and an evaluation phase. VNS has already been applied for solving many hard optimization problems such as parallel machine scheduling problem (Rocha de Paula *et al.*, 2007), location and routing problem (Jarboui *et al.*, 2013), etc.

Some versions of IRP were already solved by VNS. For instance, Hemmelmayr *et al.* (2009) consider a version of the problem modelled as a periodic vehicle routing problem and they propose a VNS algorithm to solve it. Recently, Popovic *et al.* (2012) considered a problem of fuel distribution with a multi-compartment fleet of vehicles. In this problem, different types of fuel are delivered to a set of customers by vehicles with compartments. They propose a VNS to solve the problem since the proposed MIP model could only handle the smallest instance. Vidovic *et al.* (2014) extended the previous work by considering a second formulation when including the fleet size costs. Then, they propose two variable neighbourhood descent (VND) algorithms based on a local intra-period neighbourhood search and a large inter-period neighbourhood search. The results show that the heuristics produce high solution quality with in average $<7\%$ higher total costs compared with the optimal solution.

Lee *et al.* (2003) proposed a MIP model for the variant of IRP considered in this paper, and they developed a hybrid method combining an annealing-based heuristic to generate alternative sets of routes and a linear programme to determine the corresponding optimum inventory levels. Moin *et al.* (2010) derived lower and upper bounds for the problem after solving a MIP formulation with the purpose solver CPLEX. Then, they designed a genetic algorithms to obtain better upper bounds. Recently, Mjirda *et al.* (2013) proposed a two phase approach based on VNS to solve the problem. In the first phase, VNS is used to solve a capacitated vehicle routing problem at each period to find an initial solution without taking into account the inventory. Then, they improved the solution with a VND and a VNS algorithm and by taking into account both transportation and inventory costs. Computational experiments showed the efficiency of the approach compared with Moin *et al.* (2010). In particular, the proposed algorithms provide solutions with a better total cost for all the instances. One of the drawback in this approach is the running time needed to obtain the best solution for some instances. That can be partly explained by the use of an inventory management heuristic developed to determine the quantity of products to be collected. This heuristic is called at each solution move in the VND approach (see Mjirda *et al.*, 2013 for more details). To avoid this problem, we propose in this paper a new VNS algorithm with a different solution representation. This representation will be described in the following section.

3. Solution representation and neighbourhood structures

3.1 *Solution representation*

To deal with the problem, we decompose a solution in a set of τ *sub* or *partial* solutions $S = \{S_1, \ldots, S_\tau\}$. Each partial solution S_t is associated with the corresponding period $t \in T$. Then, for a given period t,

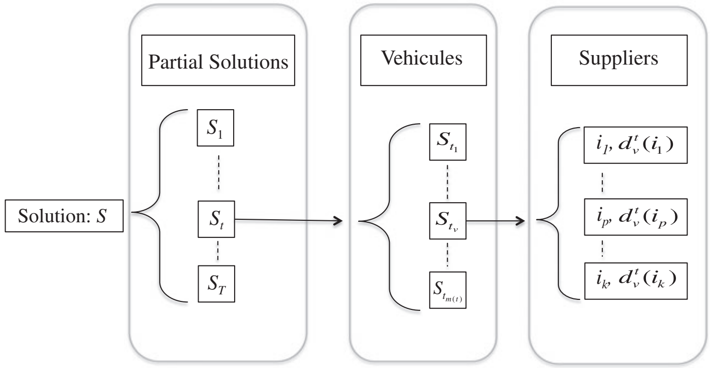

FIG. 1. Solution representation.

the partial solution S_t is decomposed into a set of routes $S_t = \{S_{t_1}, \ldots, S_{t_{m(t)}}\}$, where $m(t)$ is the number of vehicles used during the period t (since one route is associated with one vehicle). A third level of decomposition is used to represent the sequence of k suppliers visited by a given vehicle v at period t, $S_{t_v} = \{i_1, \ldots, i_k\}$. To build the inventory routing policy corresponding to a given solution S, we need to have the information related to the quantity of product to collect from each supplier at each period. For each supplier $i \in S_{t_v}$, let $d_v^t(i)$ be the quantity to collect at period t by vehicle v. Figure 1 illustrates the solution representation used in this work.

3.2 *Discrete management of the quantities to collect*

An important element of the solution is the determination of quantities to collect. In related works dealing with this problem (Lee *et al.*, 2003; Moin *et al.*, 2010; Mjirda *et al.*, 2013), authors manage the quantities $d_v^t(i)$ as continuous variables, which means that, for a given period t, the amount to be collected from a given supplier i is a part of the assembly plant's demand. In this paper, we consider the quantities as integer variables. More precisely, if a supplier is visited at a period h, then the complete demand of the assembly plant at this period, d_h, is collected. We introduce the discrete variable y_{ih}, which is defined as

$$y_{ih} = \begin{cases} 1 & \text{if supplier } i \text{ is visited at period } h \\ 0 & \text{otherwise} \end{cases}$$

Then, the quantity $d_v^t(i)$, for $i \in S_{t_v}$, is determined as follows:

$$d_v^t(i) = \sum_{h=t}^{T} d_{ih} y_{ih} \tag{3.1}$$

The previous definition of $d_v^t(i)$ implies that the corresponding vehicle v can collect quantities of product from supplier i associated with period later than the current one t. However, an important point is the fact that, for a given period $h \in [t, T]$, the vehicle collects all the product associated with the demand d_{ih} or it does not collect the product for this period. This brings that we only manage discrete quantities of products in a given solution S. A consequence of this management of demand is the possible generation of empty routes.

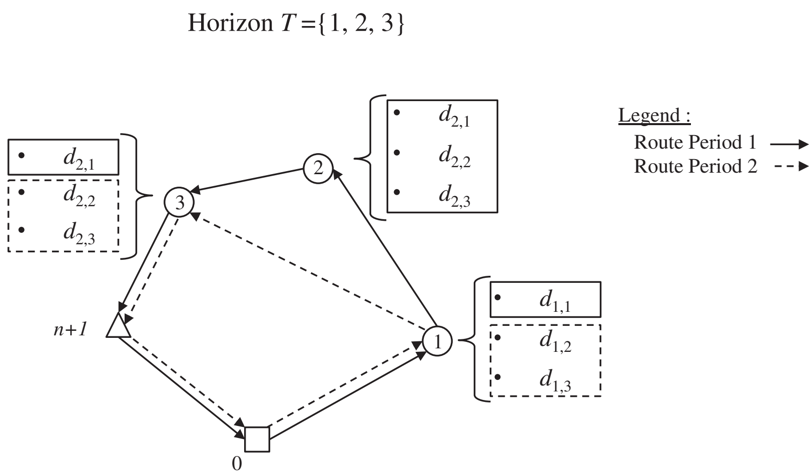

FIG. 2. Illustration of an empty partial solution.

EXAMPLE We consider an example with three suppliers, three periods ($T = \{1, 2, 3\}$) and only one vehicle. Suppose that in Period 1 the vehicle visits the three suppliers. Then, the partial solution for the first period can be defined as: $S_{1_1} = \{1, 2, 3\}$. Suppose that during this route the vehicle collects the quantity of product associated with demand $d_{1,1}$ from Supplier 1 ($d_1^1(1) = d_{1,1}$), all the quantity of product requested from Supplier 2 (i.e $d_1^1(2) = d_{2,1} + d_{2,2} + d_{2,3}$) and the quantity of product for the first period from Supplier 3 ($d_1^1(3) = d_{3,1}$). Then, suppose that the vehicle collects the required demand for Period 2 and 3 from Suppliers 1 and 3 in the second period. In that case, we have $S_{2,1} = \{1, 3\}$ and $d_1^2(1) = d_{1,2} + d_{1,3}$, $d_1^2(3) = d_{3,2} + d_{3,3}$. It is easy to observe that all the required products were collected during the first two periods, which means that for Period 3 there is no routes, $S_{3,1} = \emptyset$. This example is illustrated in Fig. 2.

As mentioned above, the solution representation defined in this paper allows us to define moves based on quantities' modifications without the need of a heuristic (or a mathematical model) to readjust these quantities in the solution. We present the neighbourhood structures used in our VNS algorithm in the next section.

3.3 *Neighbourhood structures*

We define six neighbourhood structures based on the movement of quantities between routes and periods. These neighbourhood structures can be decomposed into two categories: insertion move and exchange move, and they are described in the following.

3.3.1 *Insertion neighbourhood N_1* We define three neighbourhood structures based on an insertion move. They may concern just one route or two routes of the same period or two routes of different periods.

Insertion move in the same route (shift neighbourhood) $N_{1,1}$

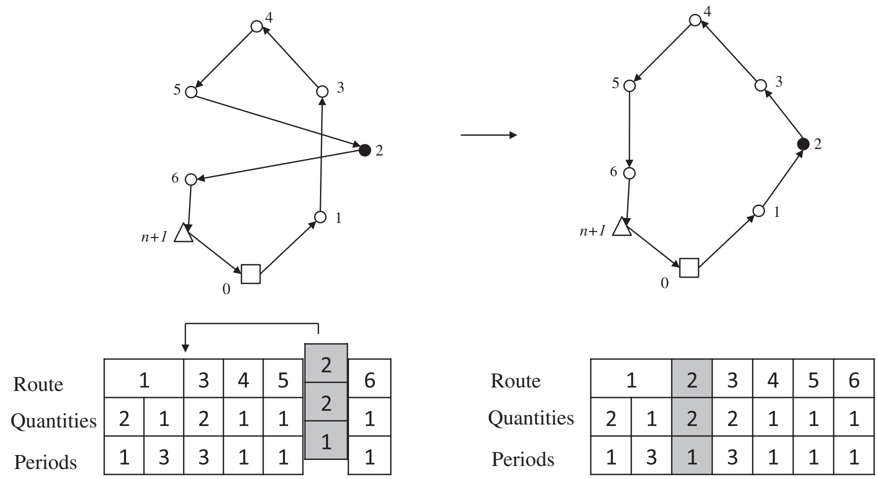

FIG. 3. Insertion move in the same route (shift neighbourhood) $N_{1,1}$.

The shift neighbourhood structure consists of changing the order in which quantities are collected from suppliers in one route. This move corresponds to the shift of a supplier from its current position to a new position. This is a classical move used for routing problems and it is referred as 1-0 move (i.e., supplier relocation). This simple move is illustrated in Fig. 3. In this figure and in the following ones, we present examples of routes and tables that resume the suppliers to be visited in the row denoted by Routes, the Quantities to collect from the corresponding suppliers of a route and the corresponding Periods. To illustrate these, we can consider the route in Fig. 3 obtained after the move. In this route, the vehicle visits the Suppliers $1, 2, \ldots, 6$. Then, it goes to the assembly plant before going back to the depot. At Supplier 1, the vehicle collects 2 unit of product corresponding to the Period 1 and 1 unit of product at the Period 3.

We can observe that this move modifies only the transportation cost of the current solution.

Insertion move between two routes from the same period $N_{1,2}$

The insertion move between two routes concerns the movement of a quantity to collect from one route to another route of the same period. Two cases can be considered:

- The supplier already exists in the route in which the insertion is done. In that case the quantity to be inserted has to be merged with the quantities already planned to be collected from this supplier. Figure 4 summarizes this case. In this example, the quantity of product collected from Supplier 3 originally in the route of Vehicle 2 is moved to the route of Vehicle 1. After the move the Customer 3 does not appear in the second route (i.e. $d_2^1(3) = 0$), whereas the quantity of product associated with Periods 1 and 2 is collected by Vehicle 1 (i.e. $d_1^1(3) = d_{3,1} + d_{3,2} = 6$).
- The supplier does not exist in the route of insertion which means that the insertion of a quantity is equivalent to the insertion of a new supplier in this route, as illustrated in Fig. 5. In this example, the Supplier 6 is moved from the second route to the first one.

Insertion move between two routes in different periods $N_{1,3}$

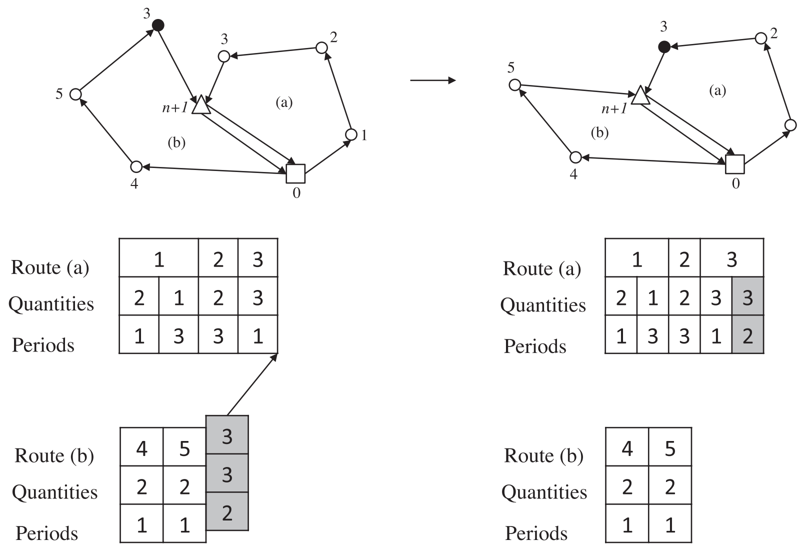

FIG. 4. Case 1 of insertion move $N_{1,2}$ (Supplier 3).

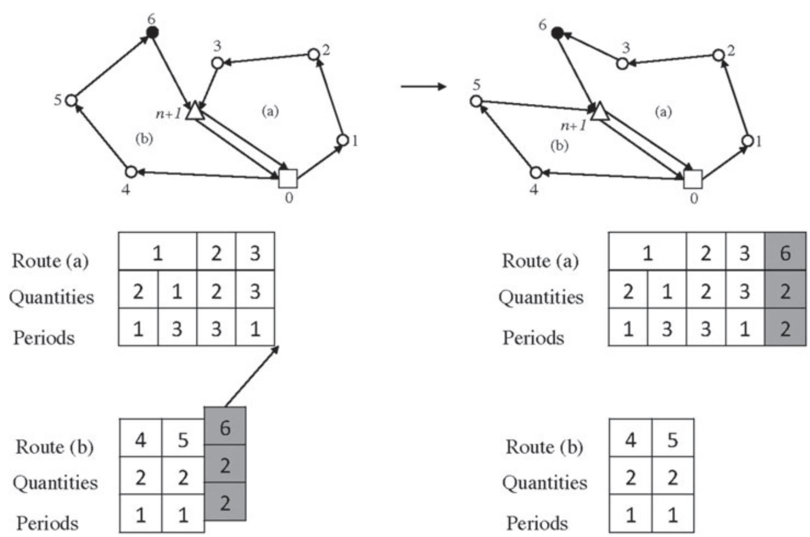

FIG. 5. Case 2 of insertion move $N_{1,2}$ (Supplier 6).

The third neighbourhood consists of inserting a quantity from a route to another route in a different period. As in $N_{1,2}$ if the supplier exists in the route, quantities will be jointed. Otherwise, the insertion of a quantity means the insertion of a new supplier in the route.

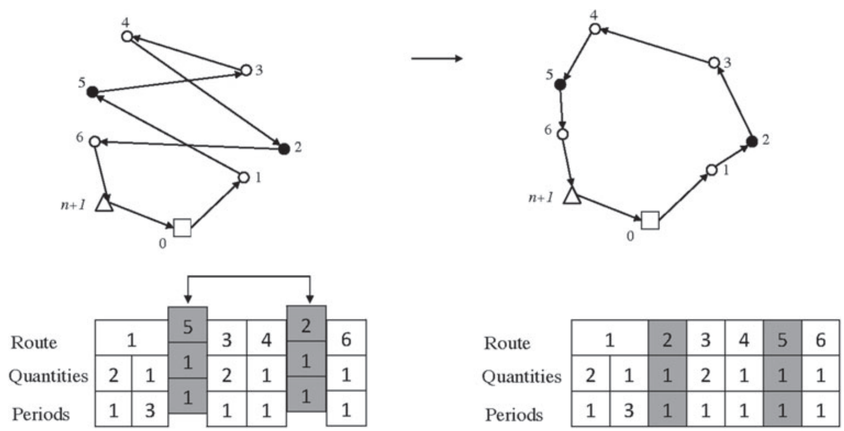

FIG. 6. Exchange move in one route $N_{2,1}$ (Suppliers 2 and 5).

3.3.2 *Exchange neighbourhood N_2* We define three neighbourhood structures based on the exchange of quantities on a given route, between two routes at the same period, or between two routes at different periods.

Exchange move within the same route $N_{2,1}$

This neighbourhood consists in exchanging the position of two quantities to collect from two different suppliers in the same route. Then, it is equivalent to exchange the order of visiting suppliers in one route. A 2-opt move can be performed for some special cases during the execution of this move. An example is given in Fig. 6.

Exchange move between two routes $N_{2,2}$

With this move we exchange the quantities to collect from two suppliers on two distinct routes at the same period. Like previously for the insertion move, two cases can be discussed:

- If the suppliers are only present in their initial route (and not in the route associated with the move), then the move is a classical exchange between two suppliers. This is a common move on routing problem and it is referred as 1-1 exchange move. Figure 7 provides an illustration with the exchange of Customers 3 and 5.
- If at least one of the two suppliers is already present in the route associated with the move, then the quantities to move will be jointed together. Figure 8 illustrates this situation for Suppliers 3 and 5. After the move, Supplier 3 is only in the second route and the quantity collected by the vehicle is $d_2^1(3) = d_{3,1} + d_{3,2}$ (assuming we are in Period 1).

Exchange move between two routes from different periods $N_{2,3}$

This neighbourhood structure is a generalization of the previous neighbourhood $N_{2,2}$, but it concerns routes from different periods.

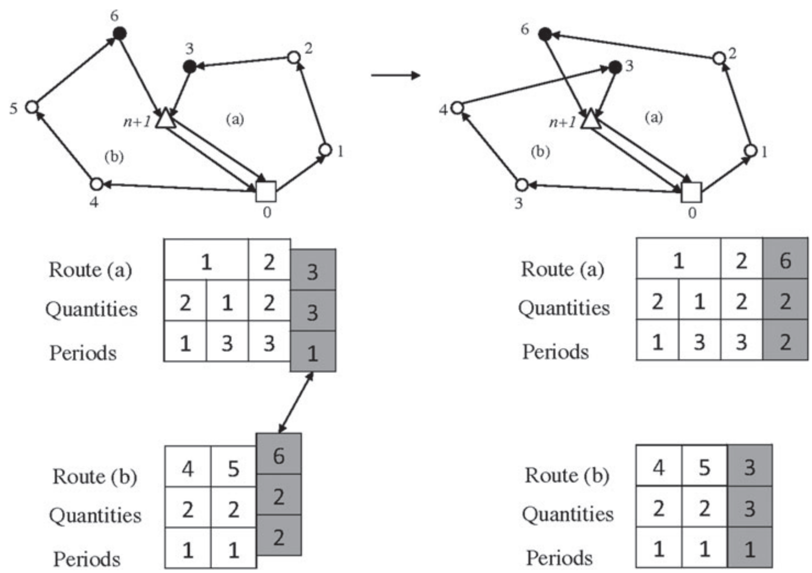

FIG. 7. Case 1 of exchange move $N_{2,2}$ (Suppliers 3 and 5).

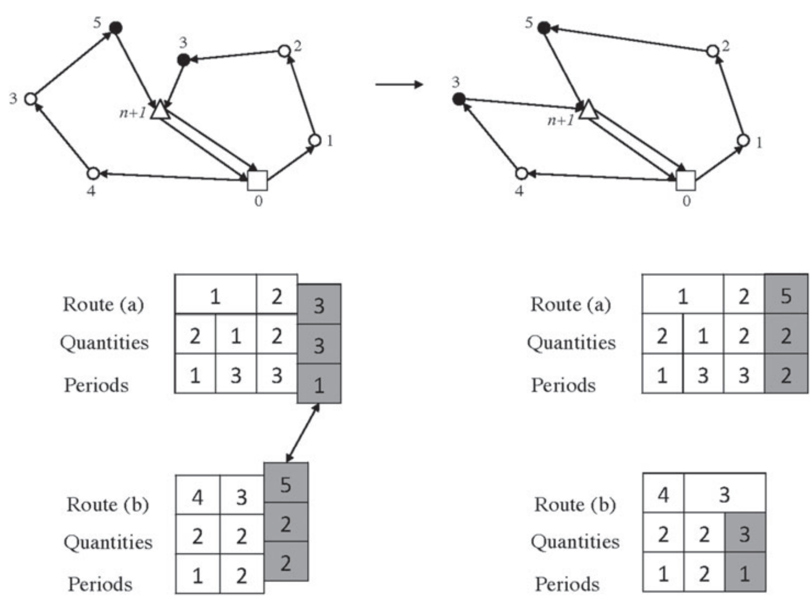

FIG. 8. Case 2 of exchange move $N_{2,2}$ (Suppliers 3 and 5).

4. A GVNS for the IRP

To solve the considered IRP, we apply a general VNS (GVNS) metaheuristic that uses a VND as a local search routine.

The VND algorithm is a deterministic variant of VNS. To describe this algorithm, we first define a local search algorithm denoted by $LSN(S, N_{i,1}, N_{i,2}, N_{i,3})$, where S is the initial solution, and $N_{i,1}$, $N_{i,2}$ and $N_{i,3}$ are three different neighbourhood structures for each $i \in \{1, 2\}$. This algorithm is a descent method and it is used in our VND. In the *LSN* algorithm, we explore consecutively the neighbourhoods $N_{i,1}$, $N_{i,2}$ and $N_{i,3}$ around the current solution until no improvement is possible, for a given $i \in \{1, 2\}$.

Algorithm 1: $LSN(S, N_{i,1}, N_{i,2}, N_{i,3})$

repeat
  $S_1 \leftarrow$ Local Search$(S, N_{i,1})$;
  $S_2 \leftarrow$ Local Search$(S_1, N_{i,2})$;
  $S_3 \leftarrow$ Local Search$(S_2, N_{i,3})$;
  $\Delta = f(S_3) - f(S)$;
  if $\Delta < 0$ **then**
    $S \leftarrow S_3$;
  end
until $\Delta \geqslant 0$
return *S;*

Next, we define the VND algorithm denoted by VND$(S, \mathcal{N}_{\text{set1}}, \mathcal{N}_{\text{set2}})$, where S is the initial solution and $\mathcal{N}_{\text{set1}} = \{N_{1,1}, N_{1,2}, N_{1,3}\}$ is the set of neighbourhood structures based on insertion moves and $\mathcal{N}_{\text{set2}} = \{N_{2,1}, N_{2,2}, N_{2,3}\}$ is the set of neighbourhood structures using exchange moves. This algorithm can be considered as a variant of the classical VND. More precisely, *LSN* is a deterministic descent method. It is used as a subroutine within VND to explore both the insertion and the exchange neighbourhood structures. The VND stops when no improvement is possible. The different steps of our VND algorithm are described in Algorithm 2.

Algorithm 2: VND$(S, \mathcal{N}_{set1}, \mathcal{N}_{set2})$

repeat
  $S_1 \leftarrow$ LSN$(S, N_{1,1}, N_{1,2}, N_{1,3})$;
  $S_2 \leftarrow$ LSN$(S_1, N_{2,1}, N_{2,2}, N_{2,3})$;
  $\Delta = f(S_2) - f(S)$;
  if $\Delta < 0$ **then**
    $S \leftarrow S_2$;
  end
until $\Delta \geqslant 0$
return *S;*

Our GVNS uses the previous VND as a local search subroutine which is the main difference with the basic VNS. An important issue related to our GVNS is the shaking phase. It is based on the insertion neighbourhood structures $N_{1,q}$, for $q \in \{1, 2, 3\}$. More precisely, at each iteration of the search we choose the neighbourhood to shake randomly between $N_{1,1}$, $N_{1,2}$ and $N_{1,3}$. Then, for a given neighbourhood $N_{1,q}$, we apply a shaking phase k times $(1 \leqslant k \leqslant k_{\max})$. We introduce the notation $N^{[k]}$ to denote k consecutive uses of the neighbourhood N. The multiple use of a given neighbourhood in the shaking phase can be considered as a difference with the basic VNS. When the shaking phase is done, we launch the VND algorithm to look for a better solution in the neighbourhood structures. The principle of our algorithm is summarized in Algorithm 3. Parameters of the algorithm are: S, an initial solution; $t_{\max}$ that corresponds to the maximum running time of the algorithm and $k_{\max}$ that is the maximum consecutive number of applications for a given neighbourhood. Notation $r[1,3]$ denotes the generation of a random integer number in $[1,3]$.

Algorithm 3: GVNS(S, k_{max}, t_{max})

```
Set 𝒩_set1 = {N_{1,1}, N_{1,2}, N_{1,3}};
Set 𝒩_set2 = {N_{2,1}, N_{2,2}, N_{2,3}};
repeat
    k ← 1;
    while k ⩽ k_max do
        i ← r[1,3];
        S_1 ← (N_{1,i})^[k](S);
        S_2 ← VND(S_1, 𝒩_set1, 𝒩_set2) ;
        k ← k + 1;
        if f(S_2) < f(S) then
            S ← S_2;
            k ← 1;
        end
    end
until t < t_max
return S;
```

5. Computational results

The proposed GVNS-based heuristic was implemented in C++ language. Tests were performed on a Pentium IV processor with 3.4 GHz of frequency and 4 GB RAM. The computational analysis is based on a set of 14 instances provided in Moin *et al.* (2010) and available in the Center for Logistics and Heuristic Optimisation (CLHO) website.[1] They are characterized by the number of suppliers n and the number of periods τ. For the proposed GVNS algorithm, we set the time limit $t_{\max} = 1,000$ s as a stopping criterion. We performed 10 runs for each instance.

We report in Table 1 the best objective value (denoted by f^*) and the number of vehicles (#v) obtained by the GA proposed in Moin *et al.* (2010), the 2p-VND and the 2p-VNS presented in Mjirda *et al.* (2013) and the proposed GVNS. Bold values in Table 1 refer to the best known solution values for the considered instance. Then, the CPU times actually used to get the best reported solutions (within 1,000 s) are presented in columns ‘Time’. We denote that GA were run on 2.8 GHZ processor with 4 GB RAM whether 2p-VND and 2p-VNS were run on a Pentium IV processor with 3.4 GHz of frequency and 4 GB RAM.

From Table 1, we outline the following conclusions:

- The new proposed GVNS provides better results for 12, 13 and 14 instances (out of 14) when compared with the 2p-VNS, the 2p-VND and the GA, respectively.

[1] http://www.kent.ac.uk/kbs/research/research-centres/clho.

TABLE 1 *Comparison between the GA in Moin et al. (2010), the two phase methods in Mjirda et al. (2013) and the proposed GVNS approach*

		GA			2p-VND			2p-VNS			GVNS		
Inst	(n, τ)	f^*	$\#v$	Time	f^*	$\#v$	Time	f^*	$\#v$	Time	f^*	$\#v$	Time
S12T5	(12, 5)	2,096.75	14	58.48	**1,961.71**	14	0.08	**1,961.71**	14	0.10	**1,961.71**	14	0.10
S12T10	(12, 10)	4,333.27	29	111.23	**4,002.85**	29	0.21	**4,002.85**	29	0.20	**4,002.85**	29	0.20
S12T14	(12, 14)	6,115.19	41	120.31	**5,635.77**	41	0.34	**5,635.77**	41	0.30	**5,635.77**	41	0.30
S20T5	(20, 5)	3,143.39	21	31.81	3,045.08	23	0.34	**2,861.26**	21	25.30	2,880.65	21	11.33
S20T10	(20, 10)	6,499.40	43	126.03	6,098.15	46	3.12	5,944.72	43	102.30	**5,919.38**	46	81.54
S20T14	(20, 14)	9,208.43	61	360.33	8,589.36	65	9.63	8,422.02	63	89.50	**8,349.86**	61	75.92
S20T21	(20, 21)	13,948.41	92	255.83	12,944.08	98	20.85	12,700.27	94	222.70	**12,627.87**	95	197.28
S50T5	(50, 5)	5,618.09	45	133.40	5,144.23	46	4.97	5,084.00	45	112.40	**5,026.55**	45	48.49
S50T10	(50, 10)	11,642.00	95	226.01	10,787.88	99	34.38	10,694.60	96	75.30	**10,544.43**	97	97.11
S50T14	(50, 14)	16,987.00	135	328.07	15,399.82	139	64.90	15,384.96	137	289.92	**15,116.20**	136	249.06
S50T21	(50, 21)	26,448.77	209	496.72	23,500.94	214	196.31	23,497.40	211	373.20	**23,331.83**	210	347.13
S98T5	(98, 5)	561,168.21	57	476.77	575,790.06	59	100.14	**557,287.97**	57	1768.30	557,901.25	57	998.20
S98T10	(98, 10)	1,124,797.57	113	1,307.26	1,160,921.87	119	467.61	1,119,767.72	114	1263.80	**1,114,725.28**	114	985.67
S98T14	(98, 14)	1,571,652.32	159	1,589.71	1,567,904.75	160	5793.01	1,577,951.04	161	1210.30	**1,562,861.43**	158	986.83
Average	—	240,261.34	79.57	401.57	242,980.47	82.29	478.28	239,371.16	80.43	395.26	237,920.36	80.29	291.37

TABLE 2 *Standard deviation*

Inst	$\Delta_{2p-\mathrm{VNS}}$ (%)			Δ_{GVNS} (%)		
	min	avg	max	min	avg	max
S12T5	0	0	0	0	0	0
S12T10	0	0	0	0	0	0
S12T14	0	0	0	0	0	0
S20T5	0	1.45	3.28	0.67	0.92	1.18
S12T10	0.43	0.87	1.29	0	0.29	0.61
S20T14	0.86	1.28	1.77	0	0.29	0.74
S20T21	0.57	1.28	1.75	0	0.24	0.66
S50T5	1.14	1.90	2.63	0	0.35	1.03
S50T10	1.42	2.26	2.82	0	0.69	1.22
S50T14	1.78	2.25	2.78	0	0.57	1.29
S50T21	0.71	1.16	1.66	0	0.61	1.49
S98T5	0	1.09	1.83	0.11	0.31	0.60
S98T10	0.45	1.71	4.09	0	0.27	0.51
S98T14	0.97	2.46	4.24	0.00	0.33	0.69
Average	0.60	1.26	2.01	0.06	0.35	0.72

- When considering the minimum number of vehicles used we can observe that the GA provides the minimum average value. However, the averages differences with the three VNS-based heuristics are not significant. Even if the use of more vehicles in the solutions leading to the minimum objective value may surprise, this situation can be explained by the fact that in all three VNS-based methods, we do not minimize only the number of vehicles.
- The average computational time required for GVNS is less than the average times required for 2p-VNS, 2p-VND and GA (see, in last row of Table 1 the values 291.37, 395.26, 478.28 and 401.57 for GVNS, 2p-VNS, 2p-VND and GA, respectively).

To complete the analysis of the results, we provide in Table 2 information about the deviation between the solution found by an algorithm and the best-known solution. We report the minimum deviation corresponding to the deviation with the best solution found over the 10 runs in column (*min*). Then, we put in column (*avg*) the average deviation observed and in column (*max*) the maximum deviation. The standard deviation is calculated as

$$\Delta(\%) = ((f - f_{\mathrm{best}})/f_{\mathrm{best}}) \times 100,$$

where f (respectively, f_{best}) denotes the value (respectively, the best-known value) for the considered instance. The results in Table 2 concern the 2p-VNS (column $\Delta_{2p-\mathrm{VNS}}$ (%)) and the GVNS proposed in this paper (column Δ_{GVNS} (%)).

From Table 2, it appears that the average deviation of GVNS is within the interval [0.06%, 0.72%] which is less than the average deviation of 2p-VNS. Thus, the new GVNS-based method may be considered as a new state-of-the-art for this problem, since it outperforms 2p-VNS, the previous best heuristic, in all three relevant criteria: the precision, the speed and the average deviation from the best-known objective value.

6. Conclusions

In this paper, we suggest a new GVNS-based heuristic for solving an IRP. In particular, we propose a new solution representation allowing the definition of movements between different time periods. The corresponding neighbourhood structures are used within a VND. Computational results on standard test instances indicate that our new GVNS-based heuristic may be seen as a new state-of-the-art heuristic for the IRP variant considered in this work. Future work may include the implementation of our heuristic to similar variants of IRP.

References

Andersson, H., Hoff, A., Christiansen, M., Hasle, G. & Løkketangen, A. (2010) Industrial aspects and literature survey: combined inventory management and routing. *Comput. Oper. Res.*, **37**, 1515–1536.

Bard, J. F., Huang, J. F., Jaillet, P. & Dror, M. (1998) A decomposition approach to the inventory routing problem with satellite facilities. *Transp. Sci.*, **32**, 189–203.

Bell, W., Dalberto, L., Fisher, M., Greenfield, A., Jaikumar, R., Kedia, P., Mack, R. & Prutzman, P. (1983) Improving the distribution of industrial gases with online computerized routing and scheduling optimizer. *Interfaces*, **13**, 4–23.

Bertazzi, L., Paletta, G. & Speranza, M. G. (2002) Deterministic order-up-to level policies in an inventory routing problem. *Transp. Sci.*, **36**, 119–132.

Campbell, A. M. & Savelsbergh, M. W. P. (2004) A decomposition approach for the inventory routing problem. *Transp. Sci.*, **38**, 488–502.

Christiansen, M., Fagerholt, K., Nygreen, B. & Ronen, D. (2007) Maritime transportation. *Transportation, Volume 14 of Handbooks in Operations Research and Management Science*. Amsterdam: North-Holland, pp. 189–284.

Christiansen, M., Fagerholt, K., Nygreen, B. & Ronen, D. (2013) Ship routing and scheduling in the new millennium. *European J. Oper. Res.*, **228**, 467–483.

Christiansen, M. & Nygreen, B. (2005) Robust inventory ship routing by column generation. *Column Generation*. New York: Springer, pp. 197–224.

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

Coelho, L. C., Cordeau, J. F. & Laporte, G. (2012) Thirty years of inventory-routing. *Technical Report, 52*. Montreal: CIRRELT.

Golden, B., Assad, A. & Dahl, R. (1984) Analysis of large-scale vehicle routing problem with inventory component. *Large Scale Syst. Inf. Decis. Technol.*, **7**, 181–190.

Hansen, P. & Mladenović, N. (2010) Variable neighbourhood search: algorithms and applications. *Ann. Oper. Res.*, **175**, 367–407.

Hemmelmayr, V., Doerner, K. F., Hartl, R. F. & Savelsbergh, M. W. P. (2009) Delivery strategies for blood products supplies. *OR Spektrum*, **31**, 707–725.

Hewitt, M., Nemhauser, G., Savelsbergh, M. & Song, J. H. (2013) A branch-and-price guided search approach to maritime inventory routing. *Comput. Oper. Res.*, **40**, 1410–1419.

Jarboui, B., Derbel, H., Hanafi, S. & Mladenović, N. (2013) Variable neighborhood search for location routing. *Comput. Oper. Res.*, **40**, 47–57.

Lee, C-G., Bozer, Y. A. & White III, C. C. (2003) A heuristic approach and properties of optimal solutions to the dynamic inventory routing problem. *Working Paper*. Canada: University of Toronto, Toronto.

Michel, S. & Vanderbeck, F. (2012) Column Generation based solution for a tactical inventory routing problem. *Oper. Res.*, **60**, 382–397.

Mjirda, A., Jarboui, B., Macedo, R., Hanafi, S. & Mladenović, N. (2013) A two phase variable neighborhood search for the multi-product inventory routing problem. *Comput. Oper. Res.* doi:10.1016/j.cor.2013.06.006.

Mladenović, N. & Hansen, P. (1997) Variable neighborhood search. *Comput. Oper. Res.*, **24**, 1097–1100.

Moin, N. H. & Salhi, S. (2007) Inventory routing problem: a logistical overview. *J. Oper. Res.*, **58**, 1185–1194.

Moin, N. H., Salhi, S. & Aziz, N. A. B. (2010) An efficient hybrid genetic algorithm for the multi-product multi-period inventory routing problem. *Internat. J. Prod. Econom.*, **133**, 334–343.

Popović, D., Vidović, M. & Radivojević, G. (2012) Variable neighborhood search heuristic for the inventory routing problem in fuel delivery. *Expert Syst. Appl.*, **39**, 13390–13398.

Rocha de Paula, M., Robson Mateus, G. & Pardalos, P. M. (2007) Solving parallel machines scheduling problems with sequence-dependent setup times using variable neighbourhood search. *IMA J. Manag. Math.*, **18**, 101–115.

Savelsvergh, M. & Song, J. (2007) Inventory Routing with continuous moves. *Comput. Oper. Res.*, **34**, 1744–1763.

Savelsvergh, M. & Song, J. (2008) An optimization algorithm for the inventory routing problem with continuous moves. *Oper. Res.*, **35**, 2266–2282.

Song, J. H. & Furman, K. C. (2013) A maritime inventory routing problem: practical approach. *Comput. Oper. Res.*, **40**, 657–665.

Stalhane, M., Rakke, J. G., Moe, C. R., Andersson, H., Christiansen, M. & Fagerholt, K. (2012) A construction and improvement heuristic for a liquefied natural gas inventory routing problem. *Comput. Ind. Eng.*, **62**, 245–255.

Vidovic, M., Popovic, D. & Ratkovic, B. (2014) Mixed integer and heuristics model for the inventory routing problem in fuel delivery. *Internat. J. Prod. Econ.*, **147**, 593–604.