

# Multi-start Iterated Local Search for Two-echelon Distribution Network for Perishable Products

S. Kande<sup>1,2</sup>, C. Prins<sup>1</sup>, L. Belgacem<sup>2</sup> and B. Redon<sup>2</sup>

<sup>1</sup>ICD-LOSI, University of Technology of Troyes, 12 rue Marie Curie, CS 42060, 10004 Troyes Cedex, France

<sup>2</sup>FuturMaster, 1 cours de l'île seguin, 92100 Boulogne-Billancourt, France

**Keywords:** Distribution network, Multi-Start Iterated Local Search, Local Search, Lot-sizing, Metaheuristic, Multi-echelon inventory, Multi-sourcing, Perishable product, Supply planning, Transport capacity, Variable neighborhood descent.

**Abstract:** This article presents a planning problem in a distribution network incorporating two levels inventory management of perishable products, lot-sizing, multi-sourcing and transport capacity with a homogeneous fleet of vehicles. A mixed integer linear programming (MILP) and a greedy heuristic were developed to solve this real planning problem. There are some instances for which the solver cannot give a good lower bound within the limited time and for other instances it takes a lot of time to solve MILP. The greedy heuristic is an alternative to the mixed integer linear program to quickly solve some large instances taking into account original and difficult constraints. For some instances the gap between the solutions of the solver (MILP) and the heuristic becomes quite significant. A multi-start iterated local search (MS-ILS), using the variable neighborhood descent (VND) and a greedy randomized heuristic, has been implemented. It has been included in an APS (Advanced Planning System) and compared with an exact resolution of the MILP. The instances are derived from actual data or built using a random generator of instances to have wider diversity for computational evaluation. The MS-ILS significantly improves the quality of solutions.

## 1 INTRODUCTION

In the problem under study, warehouses provide finished perishable products to various distribution centers with a homogeneous fleet of vehicles. A distribution center may be supplied by several warehouses. The choice of sourcing (warehouse) is determined by the availability of products in warehouses inventory, fleet of vehicles, and transport costs on roads. The goal is to come up with a compromise between the transportation costs, the warehouses and distribution centers inventory costs, the loss due to products that are outdated. Furthermore, we comply with: flow conservation, inventory conservation at warehouses and distribution centers, capacity constraints (limited fleet of vehicles on each road) and supply constraints (lot-sizing, minimum order quantities and dates).

The remainder of this article is structured as follows. In Section 2, we present a literature review. The problem addressed in the paper is described in Section 3. We explain how the multi-start iterated local search has been implemented in Section 5. Section 6 is dedicated to computational experiments and comparison

with solver solution (MILP) and greedy heuristic. The last section (7) gives the next steps of the work.

## 2 LITERATURE REVIEW

Iterated local search (ILS) method is a popular heuristic in the literature. (Lourenço et al., 2003) present a tutorial about the ILS method and its application to some problems such as the travelling salesman problem and the scheduling, and compare the ILS with other metaheuristics; also with another recent paper (Lourenço et al., 2010).

(Cuervo et al., 2014) recently propose an ILS for the vehicle routing problem with backhauls (VRPB). It is an extension of the VRP that deals with two types of customers: the consumers (linehaul) that request goods from the depot and the suppliers (backhaul) that send goods to the depot. New best solutions have been found by the authors for two instances in one of the benchmark sets. A multi-start iterated local search (MS-ILS) is developed by (Nguyen et al., 2012) for

the two-echelon location-routing problem (LRP-2E). They use three greedy randomized heuristics (cyclically to get initial solutions), the variable neighborhood descent and a tabu list. MS-ILS is also used by (Michallet et al., 2014) for the periodic vehicle routing problems with time spread constraints on services. Their paper addresses a real-world problem. The time windows must be absolutely respected and the hours of any two visits to the same customer must differ by a given time constant. They make evaluations on instances derived from classical benchmarks for the vehicle routing problem with time windows, and on two practical instances. For a particular case with a single period (the vehicle routing problem with soft time windows), MS-ILS competes with two published algorithms and improves six best known solutions.

The greedy randomized adaptive search procedure (GRASP) (Feo and Bard, 1989) (Feo and Resende, 1995) is a memory-less multi-start method in which local search is applied to different initial solutions constructed with a greedy randomized heuristic. (Villegas et al., 2010) propose two metaheuristics based on greedy randomized adaptive search procedures (GRASP), variable neighborhood descent (VND) and evolutionary local search (ELS). The problem under study is the single truck and trailer routing problem with satellite depots (STTRPSD): a vehicle composed of a truck with a detachable trailer serves the demand of a set of customers reachable only by the truck without the trailer. The accessibility constraint implies the selection of locations to park the trailer before performing the trips to the customers. The computational experiment shows that a multi-start evolutionary local search (GRASP-ELS) outperforms a GRASP/VND. Moreover, it obtains competitive results when applied to the multi-depot vehicle routing problem, that can be seen as a special case of the STTRPSD. The GRASP-ELS is also used by (Duhamel et al., 2011) for an extension of the capacitated vehicle routing problem where customer demand is composed of two-dimensional weighted items (2L-CVRP). The results show that their method is highly efficient and outperforms the best previous published methods on the topic.

We have not found any paper that proposes multi-start iterated local search method for the problem under study.

### 3 PROBLEM DESCRIPTION

The distribution network considered includes two levels: warehouses that provide perishable products to several distribution centers (see Figure 1). A distri-

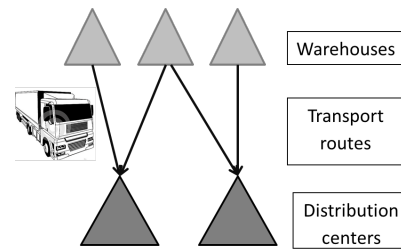


Figure 1: Two echelon distribution network with multi-sourcing.

bution center may be supplied by several warehouses. The choice of the sourcing (warehouse) is determined by the products availability in warehouses inventory, transport capacity and transport costs on the roads. The out of stock is permitted at the distribution centers; unmet demand may be postponed but is penalized by a cost. Stock levels are taken into account in the end of period. Every product shipped in a road has an associated cost of transport which determines the priority level of the warehouse. For each product-site (product-warehouse and product-distribution center) a stock policy is defined by three elements:

- a variable target stock in time: ideally reach target;
- a variable maximum stock in time: limit beyond which there is an overstock and product poses a risk of obsolescence;
- a variable minimum stock in time: reserve to cope with fluctuations in demand.

Products are made in batches to the warehouse. To differentiate with the concept of design in batches, “batch” stands for the set of product characterized by an amount of one type of product, the input date into the stock and expiry date. For each warehouse the batches of products are ordered in ascending order of their expiry dates and consumption occurs in this order. In addition, a freshness contract of product is made between the distribution center and their customers. It is a remaining life at least at the delivery time to the customer (at the period of distribution center requirement). A distribution center requirement can be satisfied by several batches of the same product. When a batch is not consumed before its expiry date then there is a loss of the remaining amount. It is penalized by a cost called waste cost.

Constraints of lot size and a minimum supply quantity must be respected. The minimum quantity expresses an economic quantity necessary for profitable order launch and reception operations. The lot size constraint corresponds sending by pallets for the transport. The combination of these two constraints (lot size and minimum quantity) makes this much

more complex than classical multi-echelon inventory management problems. A shipping calendar (opening or closing depending on the period) is also defined on each transport road.

The supply planning shall establish the quantities for each product and period to deliver to warehouses and distribution centers to satisfy at best the requirements (distribution centers) and reach the target stock (warehouses and distribution centers). The objective is to minimize the costs of transport, warehouses storage and distribution centers storage, the loss due to product obsolescence, out of stock penalties and tardiness penalties. We must respect the flow's coherence, the capacity constraints, the supply constraints (lot size, minimum quantity and dates) and seek to optimize the balancing of multi-echelon stock.

For reasons of flexibility, the user has the possibility of imposing quantity to supply inventory warehouses, distribution centers, also for transport. It is used to circumvent the various constraints in a critical situation. However, it can lead to inconsistencies in the calculation. During the execution of these heuristics, the inconsistencies are detected and reported without interrupting the supply planning calculation; which cannot be mixed with the MILP solver.

## 4 MATHEMATICAL MODEL

The problem can be formulated as a mixed integer linear programming. There are  $K$  warehouses (indexed by  $k$ ),  $L$  distribution centers (indexed by  $l$ ),  $P$  products (indexed by  $p$ ),  $T$  number of periods of horizon of calculation (indexed by  $t$ ),  $U$  number of periods of horizon of expiry dates (indexed by  $u$ ) as  $u \in 1, \dots, T + \max(DLU_{kp})$ . All indexes start at 1 except the stock levels  $sd_{lpt}$ ,  $sw_{kpt}$  for which the index starts at 0 for the initial stock. Constraints use a large positive constant  $M$ .

### 4.1 Notations

The names of the data is in upper-case and those variables in lower-case.

$D_{lpt} \in \mathbb{R}^+$ ,  $l \in L, p \in P, t \in T$  : customer requirement of the product  $p$  at the distribution center  $l$  at the period  $t$  ;

#### Data for the Supply and the Shipment:

$QT_{ipt}^{min} \in \mathbb{N}$ ,  $i \in K \cup L, p \in P, t \in T$  : minimum quantity of the product  $p$  to supply at the warehouse/distribution center  $i$  at the period  $t$  ;

$CE_{kpt} \in \{0, 1\}$ ,  $k \in K, p \in P, t \in T$  : shipment calendar; if  $CE_{kpt} = 1$  the product  $p$  can be shipped from the warehouse  $k$  at the period  $t$  ;

$CR_{lpt} \in \{0, 1\}$ ,  $l \in L, p \in P, t \in T$  : reception calendar; if  $CR_{lpt} = 1$  the product  $p$  can be receipt at the distribution center  $l$  at the period  $t$  ;

$DLU_{kp} \in \mathbb{N}$  : number of periods the quantity of the product  $p$ , supplied at the warehouse  $k$ , can be used : after this lead time the product is expired ;

$MCL_{lpt} \in \mathbb{N}$  : minimum customer life is a remaining lifetime, at least, when the product  $p$  is delivered to the customer from the distribution center  $l$  at the period  $t$  ;

$FPO_{ipt} \in \mathbb{R}^+$  : firm planned order is a quantity of the product  $p$  imposed by the user at the warehouse/distribution center  $i$  at the period  $t$  ;

$IsFPO_{ipt} \in \{0, 1\}$  :  $IsFPO_{ipt} = 1$  if there is a quantity of the product  $p$  imposed by the user at the warehouse/distribution center  $i$  at the period  $t$ , so we can not supply a different quantity for the same product at this time (the value zero is considered);

$QEID_{lpt} \in \mathbb{R}^+$  : quantity of the product  $p$  receipt at the distribution center  $l$  at the period  $t$  from a external sourcing.

#### Data for the Inventory:

$S_{ipt}^{mit} \in \mathbb{R}^+$ ,  $i \in K \cup L, p \in P$  : initial stock level of the product  $p$  at the warehouse/distribution center  $i$  ;

$S_{ipt}^{obj} \in \mathbb{R}^+$ ,  $i \in K \cup L, p \in P, t \in T$  : target stock level of the product  $p$  at the warehouse/distribution center  $i$  at the period  $t$  ;

$S_{ipt}^{max} \in \mathbb{R}^+$ ,  $i \in K \cup L, p \in P, t \in T$  : maximum stock level of the product  $p$  at the warehouse/distribution center  $i$  at the period  $t$  ;

$S_{ipt}^{min} \in \mathbb{R}^+$ ,  $i \in K \cup L, p \in P, t \in T$  : minimum stock level of the product  $p$  at the warehouse/distribution center  $i$  at the period  $t$  ;

$P_{ipt}^{>max} \in \mathbb{R}^+$ ,  $i \in K \cup L, p \in P, t \in T$  : unit penalty of overstock if we exceed the maximum stock level of the product  $p$  at the warehouse/distribution center  $i$  at the period  $t$  (per unit of surplus stock);

$P_{ipt}^{>obj} \in \mathbb{R}^+$ ,  $i \in K \cup L, p \in P, t \in T$  : unit earliness penalty when the stock level, of the product  $p$  at the warehouse/distribution center  $i$  at the period  $t$ , is between the target stock and the maximum stock;

$P_{ipt}^{<obj} \in \mathbb{R}^+$ ,  $i \in K \cup L, p \in P, t \in T$  : unit tardiness penalty when the stock level, of the product  $p$  at the warehouse/distribution center  $i$  at the period  $t$ , is between the minimum stock and the target stock (by missing unit to reach the target stock);

$P_{ipt}^{<min} \in \mathbb{R}^+$ ,  $i \in K \cup L, p \in P, t \in T$  : unit penalty applied when the stock level, of the product  $p$  at the warehouse/distribution center  $i$  at the period  $t$ , is between 0 and the minimum stock (by missing unit to reach the minimum stock);

$P_{lpt}^{up} \in \mathbb{R}^+$ ,  $l \in L, p \in P, t \in T$  : unit penalty of out of stock applied when the stock level, of the product  $p$  at the distribution center  $l$  at the period  $t$ , is under 0 (the out of stock is only permitted at the distribution centers);

$P_{kpt}^{obs} \in \mathbb{R}^+$ ,  $k \in K, p \in P, t \in T$ : unit penalty of obsolescence of the product  $p$  at the warehouse  $k$  at the period  $t$  (the lifetime is exceeded).

#### Data for the Transport:

$QTA_{klpt}^{min} \in \mathbb{N}$ : minimum quantity of the product  $p$  shipped on edge  $(k, l)$  at the period  $t$ ;

$TL_{klp} \in \mathbb{N}$ : lot size for the shipment of the product  $p$  on the edge  $(k, l)$  in units of product;

$CT_{klpt} \in \{0, 1\}$ :  $CT_{klpt} = 1$  if the transport of the product  $p$  is permitted on the edge  $(k, l)$  at the period  $t$ ;

$CA_{klt} \in \mathbb{N}$ ,  $k \in K, l \in L, t \in T$ : available capacity on the edge  $(k, l)$  at the period  $t$ , expressed as the number of pallets;

$CoPa_p \in \mathbb{R}^+$ ,  $p \in P$ : conversion factor quantity  $\rightarrow$  pallets of the product  $p$ ;

$HV_{kl} \in \mathbb{R}^+$ ,  $k \in K, l \in L$ : cost of operating a vehicle on the edge  $(k, l)$  (counted only once even if the vehicle travelled several days);

$HT_{klpt} \in \mathbb{R}^+$ ,  $k \in K, l \in L, p \in P, t \in T$ : unit cost of the shipment of the product  $p$  on the edge  $(k, l)$  at the period  $t$ ;

$LT_{klt} \in \mathbb{N}$ ,  $k \in K, l \in L, t \in T$ : lead time of transport on the edge  $(k, l)$  at the period  $t$ ;

$NP^{max} \in \mathbb{N}$ : maximum number of pallets a vehicle can be loaded (same for all vehicles);

$PV^{max} \in \mathbb{R}^+$ : maximum loading weight of a vehicle (same for all vehicles);

$PU_p \in \mathbb{R}^+$ ,  $p \in P$ : unit weight of the product  $p$ ;

$FPOA_{klpt} \in \mathbb{R}^+$ : quantity imposed to ship the product  $p$  on the edge  $(k, l)$  at the period  $t$  (useful for imposing an origin and a destination);

$IsFPOA_{klpt} \in \{0, 1\}$ :  $IsFPOA_{klpt} = 1$  if there is a quantity imposed to ship the product  $p$  on the edge  $(k, l)$  at the period  $t$  ( $FPOA_{klpt}$ );

$v_{klt}^{max}$ ,  $k \in K, l \in L, t \in T$ : number of vehicles used on the edge  $(k, l)$  at the period  $t$ .

#### Variables for the Supply and the Shipment:

$qi_{ipt}$ ,  $i \in K \cup L, p \in P, t \in T$ : quantity of the product  $p$  supplied at warehouse/distribution center  $i$  at the period  $t$ ;

$isqi_{ipt} \in \{0, 1\}$ ,  $i \in K \cup L, p \in P, t \in T$ :  $isqi_{ipt} = 1$  if a non-zero quantity of the product  $p$  is supplied at warehouse/distribution center  $i$  at the period  $t$ ;

$qo_{kpt}$ ,  $k \in K, p \in P, t \in T$ : quantity of the product  $p$  shipped by warehouse  $k$  at the period  $t$  (output quantity);

$qp_{kpt}$ ,  $k \in K, p \in P, t \in T$ : quantity of the product  $p$  lost due to obsolescence at the warehouse  $k$  at the period  $t$ ;

$n_{klpt} \in \mathbb{N}$ ,  $k \in K, l \in L, p \in P, t \in T$ : number of lots that form the quantity shipped on the edge  $(k, l)$  at the period  $t$ ;

$qe_{klpt}$ ,  $k \in K, l \in L, p \in P, t \in T$ : quantity of the product  $p$  shipped on the edge  $(k, l)$  at the period  $t$ ;

$isqe_{klpt} \in \{0, 1\}$ ,  $k \in K, l \in L, p \in P, t \in T$ :  $isqe_{klpt} = 1$  if a non-zero quantity of the product  $p$  is shipped on the edge  $(k, l)$  at the period  $t$ ;

$qed_{klpu}$ ,  $k \in K, l \in L, p \in P, u \in U$ : total quantity of the product  $p$  shipped on the edge  $(k, l)$  and whose expiration date is  $u$ ;

$qed_{pklptu}$ ,  $k \in K, l \in L, p \in P, t \in T, u \in U$ : quantity of the product  $p$  shipped on the edge  $(k, l)$  at the period  $t$  and whose expiration date is  $u$ ;

$qoi_{klpt}$ ,  $k \in K, l \in L, p \in P, t \in T$ : quantity of the product  $p$  shipped by the warehouse  $k$  and receipt at the distribution center  $l$  at the period  $t$ .

#### Variables for the Inventory:

$sdlu_{kpt}$ : quantity of the product  $p$  and whose expiration date is  $u$  at the warehouse  $k$ ;

$s_{ipt}$ ,  $i \in K \cup L, p \in P, t \in T$ : stock level of the product  $p$  at the warehouse  $k$  at the period  $t$ ;

$s_{lpt}^{rupt}$ ,  $l \in L, p \in P, t \in T$ : missing quantity to avoid the out of stock when the stock level is negative;

$s_{ipt}^{<min}$ ,  $i \in K \cup L, p \in P, t \in T$ : missing quantity to reach the minimum stock, from 0 until the minimum stock;

$s_{ipt}^{<obj}$ ,  $i \in K \cup L, p \in P, t \in T$ : missing quantity to reach the target stock, from the minimum stock until the target stock;

$s_{ipt}^{>obj}$ ,  $i \in K \cup L, p \in P, t \in T$ : quantity between the stock level and the target stock, from the target stock until the maximum stock;

$s_{ipt}^{>max}$ ,  $i \in K \cup L, p \in P, t \in T$ : quantity between the stock level and the maximum stock, when the stock level is over the maximum stock.

## 4.2 Stock Cost Formulation

Two cost functions, whose shape is shown in the figure 2, are defined to evaluate the inventory of the warehouses and the distribution centers. The function  $f(s_{ipt})$  means the stock cost of the product  $p$  in the warehouse/distribution center  $i$  at time  $t$ . This is a piecewise linear convex function. It has a zero value when the stock level is equal to the target stock. This corresponds to a cost of storage and optionally of overstock (stock level greater than the maximum stock) when the stock level is above the target stock. When the stock level is lower than the target stock, it is the cost of delay and possible risk of out of stock (stock level below the minimum stock). For the distribution centers, a penalty for negative stock (cost of out of stock) is added.

Illustrated in the figure 3, the variables of gap are defined to linearise the piecewise linear functions. These are the quantities of gap for each piece. Cost functions of stock at the warehouses and the

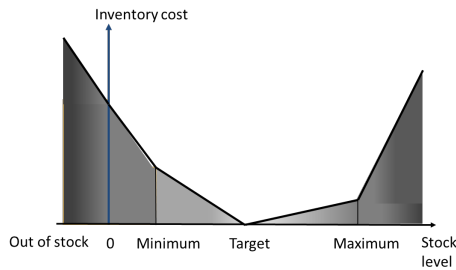


Figure 2: Stock policy and the penalties.

distribution centers may be formulated as follows:

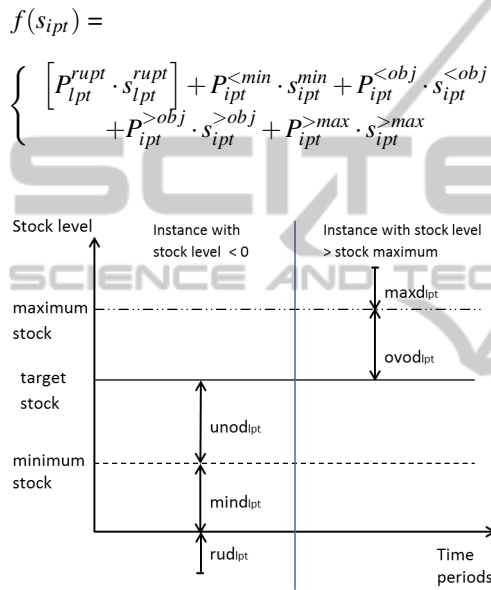


Figure 3: Variables for linearisation of the piecewise linear functions.

### 4.3 MILP Formulation

Objective function:

$$\begin{aligned} \text{Min } Z = & \quad (1) \\ & \underbrace{\sum_{k=1}^K \sum_{l=1}^L \sum_{t=1}^T \left[ (HV_{kl} \cdot v_{klt}^{max}) + \sum_{p=1}^P (qe_{klpt} \cdot HT_{klpt}) \right]}_{\text{transport cost}} \\ & + \underbrace{\sum_{i=1}^{KUL} \sum_{p=1}^P \sum_{t=1}^T f(s_{ipt})}_{\text{warehouses and distribution centers stock cost}} \\ & + \underbrace{\sum_{k=1}^K \sum_{p=1}^P \sum_{t=1}^T P_{kpt}^{obs} \cdot qP_{kpt}}_{\text{cost of loss due to obsolescence}} \end{aligned}$$

subject to:

$$\begin{aligned} qe_{klpt} &= FPOA_{klpt} \text{ if } IsFPOA_{klpt} = 1 & \forall k \ \forall l \ \forall p \ \forall t & \quad (2) \\ qe_{klpt} &= n_{klpt} \cdot TL_{klp} \text{ if } IsFPOA_{klpt} = 0, TL_{klp} \neq 0 & \forall k \ \forall l \ \forall p \ \forall t & \quad (3) \\ qe_{klpt} &\geq isqe_{klpt} \cdot (1 - IsFPOA_{klpt}) \cdot QTA_{klpt}^{min} & \forall k \ \forall l \ \forall p \ \forall t & \quad (4) \\ qe_{klpt} &\leq isqe_{klpt} \cdot M & \forall k \ \forall l \ \forall p \ \forall t & \quad (5) \\ qe_{klpt} &= qe_{klpt} \cdot CT_{klpt} \text{ if } IsFPOA_{klpt} = 0 & \forall k \ \forall l \ \forall p \ \forall t & \quad (6) \\ qoi_{klpt+LT_{klt}} &= qe_{klpt} & \forall k \ \forall l \ \forall p \ \forall t & \quad (7) \\ qi_{ipt} &= FPO_{ipt} \text{ if } IsFPO_{ipt} = 1 & \forall i \ \forall p \ \forall t & \quad (8) \\ qi_{ipt} &\geq isqi_{ipt} \cdot (1 - IsFPO_{ipt}) \cdot QT_{ipt}^{min} & \forall i \ \forall p \ \forall t & \quad (9) \\ qi_{ipt} &\leq isqi_{ipt} \cdot M & \forall i \ \forall p \ \forall t & \quad (10) \\ qi_{ipt} &= CRD_{ipt} \cdot \sum_{k=1}^K qoi_{klpt} \text{ if } IsFPO_{ipt} = 0 & \forall l \ \forall p \ \forall t > LT_{klt} & \quad (11) \\ qo_{kpt} &= CE_{kpt} \cdot \sum_{l=1}^L qe_{klpt} & \forall k \ \forall p \ \forall t & \quad (12) \\ s_{ip0} &= S_{ip}^{init} & \forall i \ \forall p & \quad (13) \\ s_{kpt} &= sw_{kpt-1} + qi_{kpt} - qo_{kpt} & \forall k \ \forall p \ \forall t > 0 & \quad (14) \\ s_{ipt} &= sd_{ipt-1} + qi_{ipt} + QEID_{ipt} - D_{ipt} & \forall l \ \forall p \ \forall t > 0 & \quad (15) \\ swdlu_{kpu} &= 0 & \forall k \ \forall p \ \forall u < DLU_{kp} & \quad (16) \\ swdlu_{kpu} &= S_{kp}^{init} & \forall k \ \forall p \ \forall u = DLU_{kp} & \quad (17) \\ swdlu_{kpu} &= q_{kpt} & \forall k \ \forall p \ \forall t & \quad (18) \\ u &= t + DLU_{kp} \\ qe_{klpt} &= \sum_{u=1}^U qed_{klptu} & \forall k \ \forall l \ \forall p & \quad (19) \\ \forall t & \geq u - DLU_{kp} \\ qed_{klpu} &= \sum_{t=1}^u qed_{klptu} & \forall k \ \forall l \ \forall p \ \forall u & \quad (20) \\ \sum_{l=1}^L qed_{klpu} &\leq swdlu_{kpu} & \forall k \ \forall p \ \forall u & \quad (21) \\ \sum_{u=1}^U qed_{klpu} &\leq \sum_{t=1}^T D_{ipt} & \forall k \ \forall l \ \forall p \ \forall u \leq T & \quad (22) \\ \forall t &\leq u - MCL_{ipt} \\ qP_{kpt} &= swdlu_{kpu} - \sum_{l=1}^L qed_{klpu} & \forall k \ \forall p \ \forall t = u & \quad (23) \\ CA_{klt} &\geq v_{klt}^{max} \cdot NP_{max} & \forall k \ \forall l \ \forall t & \quad (24) \\ v_{klt}^{max} &\geq \frac{\sum_{p=1}^P qe_{klpt} \cdot COPA_p}{NP_{max}} & \forall k \ \forall l \ \forall t & \quad (25) \\ v_{klt}^{max} &\geq \frac{\sum_{p=1}^P (qe_{klpt} \cdot PU_p)}{PV_{max}} & \forall k \ \forall l \ \forall t & \quad (26) \\ s_{ipt}^{rupt} &\geq 0 - s_{ipt} & \forall l \ \forall p \ \forall t & \quad (27) \\ s_{ipt}^{<min} + s_{ipt}^{rupt} &\geq s_{ipt}^{min} - s_{ipt} & \forall i \ \forall p \ \forall t & \quad (28) \\ s_{ipt}^{<obj} + s_{ipt}^{<min} + s_{ipt}^{rupt} &\geq s_{ipt}^{obj} - s_{ipt} & \forall i \ \forall p \ \forall t & \quad (29) \\ s_{ipt}^{>obj} + s_{ipt}^{>max} &\geq s_{ipt} - S_{ipt}^{obj} & \forall i \ \forall p \ \forall t & \quad (30) \\ s_{ipt}^{>max} &\geq s_{ipt} - S_{ipt}^{max} & \forall i \ \forall p \ \forall t & \quad (31) \\ s_{kpt}, swdlu_{kpu} &\geq 0 & \forall k \ \forall p \ \forall t \ \forall u & \quad (32) \\ s_{ipt} &\in IR & \forall l \ \forall p \ \forall t & \quad (33) \\ n_{klpt} &\in IN & \forall k \ \forall l \ \forall p \ \forall t & \quad (34) \\ qo_{kpt}, qP_{kpt}, qi_{ipt}, qe_{klpt}, qoi_{klpt}, qed_{klpu}, & & & \\ qe_{klptu} &\geq 0 & \forall i \ \forall k \ \forall l \ \forall p \ \forall t & \quad (35) \\ v_{klt}^{max} &\in IN & \forall k \ \forall l \ \forall t & \quad (36) \\ isqe_{klpt}, isqi_{ipt} &\in \{0, 1\} & \forall k \ \forall l \ \forall p \ \forall t & \quad (37) \\ s_{ipt}^{rupt} &\geq 0 & \forall l \ \forall p \ \forall t & \quad (38) \\ s_{ipt}^{<min}, s_{ipt}^{<obj}, s_{ipt}^{>obj}, s_{ipt}^{>max} &\geq 0 & \forall i \ \forall p \ \forall t & \quad (39) \end{aligned}$$

The objective function (1) includes the transport costs, the costs of stock for the warehouses and distribution centers, the cost of loss due to obsolescence. The constraints (2) and (8) ensure that the imposed

quantities are respected. The lot size of the shipped quantities is respected via the constraint (3). Constraints (4), (5), (9) and (10) concern the minimum quantities for the receipt quantities at the distribution centers and the warehouses, and the shipped quantities on the routes. If there is not a imposed quantity, we can ship if the shipping calendar is open (6). The shipping quantity arrives at the distribution center after the lead time on the route (7). The receipt quantity at the distribution center corresponds to the sum of all the shipped quantities and the reception calendar must be opened (11). The output quantity at the warehouse is equal to the sum of all the shipped quantities and the shipping calendar must be opened (12). Flow conservation and the levels of inventory are expressed via constraints (13)-(15). The inventory stock by date of expiry is expressed via constraints (16)-(18). Constraints (19)-(21) concern the use of batches of quantities taking into account their expiry date. Constraints (22) guarantee that the freshness contract (minimum customer life) is respected. The lost quantities in the warehouse (the batches that are not consumed before expiry date) are calculated using constraint (23). Constraints (24)-(26) check that the transport capacity is not exceeded by the number of pallets and the total weight of shipped products. The linearisation of the variables of gap are made with constraints (27)-(31) for the stock at the distribution centers and the stock at the warehouses. The variables are defined in the lines (32)-(39).

There are some instances for which the solver cannot give a good lower bound within the limited time and for other instances it takes a lot of time to solve MILP. A greedy heuristic has been implemented as an alternative to the mixed integer linear program to quickly solve some large instances. For some instances the gap between the solution provided by the solver (MILP) and the heuristic becomes quite significant. An multi-start iterated local search (MS-ILS) has been implemented to improve the quality of heuristic solutions.

## 5 MULTI-START ITERATED LOCAL SEARCH

The MS-ILS method uses several runs of ILS method, which is developed with a reactive randomized heuristic (RRH) and the variable neighborhood descent (VND) method.

```
MS-ILS algorithm
execute 100 times the reactive randomized
heuristic to get statistics and identify
the best settings
```

```
j:= 1
Repeat
execute the ILS method
Until {j > Number_{run} or Time limit reached}.
```

### 5.1 RRH-VND

We were inspired by the papers of (Prais and Ribeiro, 2000) and, (Prins and Calvo, 2005) for the GRASP. Prais and Ribeiro proposed reactive method GRASP for a problem in telecommunication. Prins and Wolfer Calvo also proposed a GRASP which is repeatedly running a greedy randomized heuristic that improves with a local search. In our method, a greedy heuristic is used to get initial solution and VND procedure is used to improve this solution. A greedy heuristic has been implemented as an alternative to the mixed integer linear program to quickly solve some large instances. It is divided into three main procedures to better balance stock levels. The requirement of the distribution centers is first processed regardless of target stock: this is to determine the amount necessary to avoid out of stock (minimum supply process). Then, we seek to achieve the target stock. Finally, the surplus in the warehouses is deployed to the distribution centers. This division into three phases allows better management of cases of shortage or very limited capacity, since it seeks to ensure that all the distribution centers are supplied to meet requirement, before trying to establish the target stock.

Two types of randomization have been incorporated in the heuristic. This is the sort of list of couples product-warehouse and the product-distribution center.

#### 5.1.1 Randomization of The List of Couples Product-Warehouse

In the process of minimum supply list couples product warehouse is sorted then traveled in order “natural” (each product-warehouse is indexed by its memory address). For each product-warehouse, the list of couples supplied product-distribution center is generated. The available quantity is shared in order to avoid out of stock (at couples product-distribution center). The purpose of this randomization is to test different courses from the list of couples product-warehouse. A parameter  $k$  is defined, it represents the number of elements of the set. A couple is randomly chosen among the  $k$  first items on the list of couples product-warehouse. The method is as follows:

- create a set with the first  $k$  elements;
- randomly select an item in the set;

- allocate quantities to avoid breakage distribution center product couples who purchase mainly from the product warehouse selected;
- integrate the next element located at position k+1 in the list as a whole.

### 5.1.2 Randomization of The List of Couples Product-Distribution Center

In the procurement process to achieve the target stock of couples product-distribution center, the list of couples is sorted in order of the memory address. For each product-distribution center, just in time supply is made from the main sourcing (warehouse): the unit cost of transportation and prioritizes the lowest cost associated with the arc carrying the main warehouse. If this treatment does not achieve the target stock torque then distribution center product supply options early, late and from other warehouses (secondary, tertiary, etc ...) are evaluated and applied. We seek to disrupt the criticality associated with couples distribution center product; which changes the sort order of the list of golf couples distribution center product. This randomization is performed as follows:

- calculate the criticality of each product-distribution center: in the case of sorting criteria default value of 1 is assigned to each couple;
- randomly increase the value for each criticality of 0 to k %;
- sort the list of couples distribution center product in descending order of criticality values obtained.

### 5.1.3 Variable Neighborhood Descent

(Hansen and Mladenović, 2001) proposed the variable neighborhood search (VNS) in which several neighborhoods are successively used. VNS does not follow a single trajectory but explores increasingly distant neighbors of the incumbent and jumps from this solution to a new one in case of improvement. Local search is used to get from these neighbors to local optima. VNS is based on the principle of systematically exploring several different neighborhoods, combined with a perturbation move (called shaking in the VNS literature) to escape from local optima. Variable neighborhood descent (VND) is essentially a simple variant of VNS, in which the shaking phase is omitted. Therefore, contrary to VNS, VND is usually completely deterministic.

The different movements used in our VND procedure are more or less complex. Simple movements are to increase or reduce the supplied amounts to the distribution centers and the warehouses. Complex movements can transfer amounts from a time period to an-

other, from a distribution center to another and from a warehouse to another. The decrease of the supplied quantities (movements 1 and 6) is designed to avoid overstock: when the stock level is above the target stock at product-site couples (product-warehouse and product-distribution center). When applied on a product-distribution center the quantity is spread across the flow; on the road and the sourcing (product-warehouse). The increase of supplied quantity (movements 3 and 7) can compensate for the gap when the stock level is below the target stock level at couples product-site. Amounts may be transferred between different periods of the horizon for all product-site couples (movements 2 and 9). The amount removed from the starting period is not necessarily equal to that added at the arrival time. An exchange can occur between two product-distribution center couples sharing the same transport road (movement 4). The interest of this movement is to find a balance between the stock levels of the different products in a distribution center using the same warehouse provider. Consumption of transport capacity is not optimized by the greedy heuristic. The quantity removed from the original product-distribution center may be different from the added amount. Different distribution centers supplying the same product from the same warehouse are also the subject of an exchange (movement 8). A particular movement (movement 5) is integrated to reduce the cost of loss due to product obsolescence in warehouses. When the removal is possible, the lot in question is exchanged with some or several consignments to meet the needs of product-distribution center couples. Constraints dates such as expiry date and contract freshness are always respected.

The VND stops if no movement improves solution cost or the improvement percentage (IP) of solution cost between two successive iterations is less than 0.01%. This gap is also a stop criterion for solver MILP resolution. The second criterion can reduce computational time without degrading solution quality. The formula of this gap is as follows:

$$IP = \frac{\text{last solution cost} - \text{solution cost}}{\text{solution cost}} \times 100 \quad (40)$$

```
VND algorithm
Cost(Sol):=reactive randomized heuristic solution cost
initialise Number_{Move}
Repeat
Cost(LastSol):= Cost(Sol)
i:= 1
While {i <= Number_{Move}}
For {$N(Sol,i)$}
Sol':= LocalSearch N(Sol,i)
If {Cost(Sol') < Cost(Sol)}
Sol:= Sol'
EndIf
EndFor
```

```

EndWhile
Until {Cost(Sol) >= Cost(LastSol) or IP <= 0.01
      or Time limit reached}.

```

## 5.2 Iterated Local Search

For (Lourenço et al., 2003), ILS has many of the desirable features of a metaheuristic: it is simple, easy to implement, robust, and highly effective. The essential idea of ILS lies in focusing the search not on the full space of solutions but on a smaller subspace defined by the solutions that are locally optimal for a given optimization engine. Each iteration, a copy of the best known solution is disturbed and the local search is generally used to improve it. The strength of the method is that the solutions generated are dependent and trace a path of local optima near each other in the space of solutions. ILS is also very fast: if the disturbance is slight enough, the new solution looks to the parent solution and the local search needs only few movements to re-optimize it.

In our method, a reactive randomized heuristic is used as initial solution and variable neighborhood descent (VND) is used to improve this solution. We get a local optimum at the end of VND. At each iteration, movements (perturbation) are executed to get a new solution from which VND is applied. After the perturbation the new solution is different and less good than the local optimum but VND can improve it and give a better solution. ILS stops if a fixed number of iterations is reached. The higher the number of iterations, the greater the computational time is.

```

ILS algorithm
execute VND (BestSol)
initialise shaking parameters k_{min}, k_{max}
initialise iterations number max_{iterator}
k:= k_{min}
i:= 1
Repeat
  Sol:= BestSol
  Shaking move (Sol)
  execute VND (Sol)
  If {Cost(Sol) < Cost(BestSol)}
    BestSol:= Sol
    k := k_{min}
  Else
    Sol:= BestSol
    k := k + 1
  If {k > k_{max}}
    k := k_{min}
  EndIf
EndIf
Until {i > max_{iterator} or Time limit reached}.

```

### 5.2.1 Perturbation

Perturbation is a mechanism to escape from local optima. Our perturbation is powered by four movements. The first is the movement of removing obsolete batch for product-warehouse couples. Deleting obsolete batches can degrade the stock of the warehouse over several periods; which cause an increase in the stock cost and cancel the decrease in the waste cost. When this perturbation movement is applied the change is accepted even if the cost of the solution increases. The movement of adding quantity for product-warehouse couples can correct this effect in the next VND application. The second and third movements are movements of decreasing quantity, respectively for the product-distribution center couples and the product-warehouse couples. These movements are applied in VND and may cause deterioration of the stock over future periods. The fourth movement is used to exchange amounts between the product-distribution center couples sharing the same source. In some cases the available warehouse resources is restricted to satisfy all requirements of distribution centers. Some exchanges of amounts are made to modify the distribution made by the greedy heuristic and let VND improve the solution. For each movement of perturbation, the list of product-site (product-distribution center couples or the product-warehouse couples) is randomly sorted. Each movement is applied on 10% of the associated list so that the VND easily repairs the perturbation. The ILS is implemented with a constant shaking rate of perturbation (10%).

## 6 COMPUTATIONAL EVALUATION

The multi-start iterated local search is implemented in C++ Visual Studio 2010 development environment while the MILP is solved with the solver CPLEX version 12.6. The instances (60) are tested on a server with an Intel Xeon 2.93 GHz processor and 48 GB of RAM.

### 6.1 Instances

The 20 first instances have been extracted from actual databases representing real distribution networks for customers of FuturMaster, a french software publisher. The 40 other instances have been randomly generated to have wider diversity for computational evaluation. First database has provided 10 small instances with the number of products varies from 2



to 10, 2 warehouses, 2 distribution centers and horizon times is from 10 to 20 periods. The 10 instances from second database have the number of products that varies from 5 to 20, 11 warehouses, 145 distribution centers and horizon times varying from 10 to 20 periods. We have tried to evaluate greater instances with this database but when the number of products exceeds 20 the solver is out of memory. The random generator of instances has provided 40 instances with the number of products varying from 5 to 50, warehouses from 5 to 10, distribution centers from 10 to 20 and horizon times from 10 to 30 periods. In all instances, each product is perishable, with a lifetime, and a freshness contract (minimum customer life) is defined for each couple product-distribution center. To fix ideas, the MILP of the smaller instance contains 1,760 constraints and 2569 variables and for the greater instance there are 921,712 constraints and 2,760,510 variables.

### 6.2 Results

The results of MILP solver and greedy heuristic (initial solution) for all instances are in Table 1. The default parameter of relative gap between lower bound (LB) and upper bound (UB), 0.01, is used as stop criterion. The computational time is in seconds. The gap between greedy heuristic solution cost (H) and lower bound of MILP (H/LB) and its computational times are given in Table 1:

$$\% \text{ Gap UB/LB} = \frac{\text{UB} - \text{LB}}{\text{LB}} \times 100 \quad (41)$$

$$\% \text{ Gap H/LB} = \frac{\text{H} - \text{LB}}{\text{LB}} \times 100 \quad (42)$$

We note that for the instances from the database 1 the MILP resolution finds a good solution with 0.001 as average gap between lower and upper bound for 29.51 seconds. For the instances extracted from the database 2 and from the random generator, the computational time of MILP resolution increases: in average 7427.13 seconds for database 2 and 1418.28 seconds for the instances generated randomly. The gap UB/LB decreases too, and for some instances of the random generator, the solver does not provide a good lower bound after 1 hour and half of calculation: 11 are not represented in the results and heuristic results are not compared with MILP solver. Greedy heuristic is very fast (in average less than 1 second) but for the instances of the database 2 (16.626%) and the random generator (31.829% ) the average gap, between the LB and the cost of heuristic solution, becomes quite significant. The worst cases are due to the structure of the greedy heuristic.

Table 1: Results MILP solver and greedy heuristic.

Instances from	MILP solver		Heuristic (H)	
	%Gap UB/LB	Time (s)	%Gap H/LB	Time (s)
Database 1	0.001	29.51	0.010	0.02
Database 2	0.073	7427.13	16.626	0.66
Random generator	0.519	1418.28	31.829	0.04
Average for all	0.322	2361.15	22.233	0.16

MS-ILS is compared with an exact resolution of the MILP and we show the improvement of the solution quality of the heuristic and the VND in Table 2. The average gap between the lower bound of MILP resolution and the MS-ILS is 3.445%. The average gap between the solver solution and greedy heuristic solution (initial solution) is 22.233% then this gap decreases by 18.674% when the MS-ILS is tested with three runs; the ILS is applied with 10 calls of VND. The average computational time of calculation of MS-ILS is 1 890.47 seconds. The next step is to reduce the computational time and test the ILS with more iterations to reduce the gap again.

Table 2: MS-ILS results.

Instances from	MS-ILS with 3 runs of 10 ILS iterations			
	%Gap	%Gap	%Gap	Time
	MS-ILS/LB	MS-ILS/H	MS-ILS/VND	(s)
Database 1	0.009	-0.001	0.000	160.00
Database 2	1.359	-14.467	0.292	2 590.60
Random generator	5.349	-25.067	-1.325	2 245.76
Average for all	3.445	-17.788	-0.741	1 890.47

## 7 CONCLUSION

A mutli-start iterated local search (MS-ILS) is implemented and applied to a problem of two-echelon distribution network with capacity constraints, multi-sourcing and lot-sizing for perishable products. We compare the solutions of MS-ILS with the resolution of the MILP. The problem under study is an actual industrial problem, the method is included in an APS (Advanced Planning System). Some customers reject the CPLEX solution because we can not explain with simple rules how it is built, and require heuristics for which they can understand the logic.

There are some instances for which the solver cannot give a good lower bound within the limited time. Our next step is to develop a Lagrangian relaxation to evaluate these instances with the other methods: greedy heuristic, VND and MS-ILS. Lagrangian relaxation solution could also be used to build a feasible solution with a repair heuristic.

## ACKNOWLEDGEMENTS

This work was supported by FuturMaster, a french software publisher. The reviewers of the paper are greatly acknowledged for their helpful comments.

## REFERENCES

- Cuervo, D., Goos, P., Sörensen, K., and Arráiz, E. (2014). An iterated local search algorithm for the vehicle routing problem with backhauls. *European Journal of Operational Research*, 237:454–464.
- Duhamel, C., Lacomme, P., Quilliot, A., and H. Toussaint (2011). A multi-start evolutionary local search for the two-dimensional loading capacitated vehicle routing problem. *Computers & Operations Research*, 38:617–640.
- Feo, T. and Bard, J. (1989). Flight scheduling and maintenance base planning. *Management Science*, 35(12):1415–1432.
- Feo, T. and Resende, M. (1995). Greedy randomized adaptive search procedures. *Journal of Global Optimization*, 6:109–133.
- Hansen, P. and Mladenović, N. (2001). Variable neighborhood search: Principles and applications. *European Journal of Operational Research*, 130(3):449–467.
- Lourenço, H., Martin, O., and Stützle, T. (2003). In: *Handbook of Metaheuristics*, chapter Iterated local search, pages 321–353. Kluwer and Dordrecht.
- Lourenço, H., Martin, O., and Stützle, T. (2010). In: *Handbook of Metaheuristics*, volume 146, chapter Iterated local search: Framework and applications, pages 363–397. Springer, New York.
- Michallet, J., Prins, C., Amodeo, L., Yalaoui, F., and Vitry, G. (2014). Multi-start iterated local search for the periodic vehicle routing problem with time windows and time spread constraints on services. *Computers & Operations Research*, (41):196–207.
- Nguyen, V., Prins, C., and Prodhon, C. (2012). A multi-start iterated local search with tabu list and path relinking for the two-echelon location-routing problem. *Engineering Applications of Artificial Intelligence*, 25:56–71.
- Prais, M. and Ribeiro, C. (2000). Reactive grasp: an application to a matrix decomposition problem in tdma assignment. *INFORMS Journal on Computing*, 12(3):164–176.
- Prins, C. and Calvo, R. W. (2005). A fast grasp with path relinking for the capacitated arc routing problem. In et C. Mouro, L. G., editor, *INOC 2005: 3rd International Network Optimization Conference, Lisbonne, Portugal, 20-23/03/2005*, pages 289–295. Université de Lisbonne.
- Villegas, J. G., Prins, C., Prodhon, C., Medaglia, A. L., and Velasco, N. (2010). Grasp/vnd and multi-start evolutionary local search for the single truck and trailer routing problem with satellite depots. *Engineering Applications of Artificial Intelligence*, 23:780–794.