Location Inventory Routing Problem

Guillaume Massonnet*[1] and Olivier Péton†[1]

[1]*IMT Atlantique, LS2N, 4 rue Alfred Kastler, 44300 Nantes, France*

March 19, 2018

Abstract

The abstract here.

1 Introduction

Location Inventory Routing Problem (LIRP) aims at integrating in a unified model the three level of decisions involved in the optimization of the supply chain. The strategical aspect relates to the designe of the network, by positioning and opening distribution centers on a geographical area. At the tactical, production and inventory management decisions are made while the operational part of the problem focuses on logistics and transportation solutions.

1.1 Literature review

Fill the .bib file and reference the papers, explain what the authors are doing.

2 The model

2.1 Description and notations

In this work, we present a unified model that covers a broad spectrum of systems in which one supplier (or central plant) produces semi-finished or end products. There is a set of candidates depots, that is location onto which *distribution centers* (DC), or *depots* may be opened to operate as intermediate between the central supplier and a (often larger) set of retailers, also called customers. The decision to open or not a DC incurs a fixed opening cost for each depot at the beginning of the planning horizon and cannot be changed during the exploitation of the network. Units stored at a DC or at a customer incur a holding cost that represents the financial immobilisation of the good as well as maintenance costs. Finally deliveries are made through predefined routes through the network. A route starts either from the supplier and visits a set of depots before returning to the supplier or starts from a depot and visits a set of customers before returning to the same depot. Using a route to deliver units during a time period also incurs a transportation cost that consists of a (possibly zero) fixed ordering cost along with a routing cost that depends on the length of the route and its number of stops.

Throughout the paper, we use the following notations:

*guillaume.massonnet@imt-atlantique.fr

†olivier.peton@imt-atlantique.fr

Set	Definition
T	Length of the planning horizon
L	Number of levels in the distribution network
$\mathcal{L}_k$	Set of distribution centers j of level $k = 1, \ldots, L$
$\mathcal{L} = \cup_{k=1}^{L} \mathcal{L}_k$	Set of all locations in the distribution network
$\mathcal{C} = \mathcal{L}_L$	Set of customers
$\mathcal{R}_k$	Set of routes starting from a DC at level $k-1$ and visiting and a subset of locations at level k, $k = 1, \ldots,$
$\mathcal{R} = \cup_{k=1}^{L} \mathcal{R}_k$	Set of all the routes in the distribution network
Data	Definition
f_j	Fixed cost of opening distribution center $j \in \mathcal{L}_k$, $k = 1, \ldots, L-1$
κ_k	Capacity of vehicles delivering locations at level k
ν_k	fleet size for vehicles delivering locations of level k
d_i^t	Demand faced by customer $i \in \mathcal{C}$ in period $t = 1, \ldots, T$
h_i^t	Per-unit, per-period holding cost of location $i \in \mathcal{L}$ in period $t = 1, \ldots, T$
I_{i0}	Initial inventory of location $i \in \mathcal{L}$ at the beginning of the planning horizon
c_r	Cost of route $r \in \mathcal{R}$
α_{ir}	indicator that route r visits location i
β_{jr}	indicator that route r starts from DC j
$I_i^{\max}$	Maximum inventory at location $i \in \mathcal{L}$
Variables	
Binary Variables	
y_j	$\rightarrow$ 1 if distribution center j is selected
z_r^t	$\rightarrow$ 1 if route r from level is selected in period t
Continuous variables	
u_{ir}^t	$\rightarrow$ quantity delivered by route r to location i in period $t = 1, \ldots, T$
I_i^t	$\rightarrow$ Inventory at location i in period $t = 1, \ldots, T$

2.2 MIP formulation

$$\text{minimize} \quad \sum_{j\in\mathcal{L}\setminus\mathcal{C}} f_j y_j + \sum_{t=1}^{T}\left(\sum_{r\in\mathcal{R}} c_r z_r^t + \sum_{i\in\mathcal{L}} h_i^t I_i^t\right) \tag{1}$$

$$\text{s.t.} \quad \sum_{r\in\mathcal{R}_L} \alpha_{ir} z_r^t \leq 1 \qquad \forall i\in\mathcal{C}, \forall t=1,\ldots,T \tag{2}$$

$$\sum_{r\in\mathcal{R}_k} \alpha_{jr} z_r^t \leq y_j \qquad \forall k=1,\ldots,L-1, \forall j\in\mathcal{L}_k, \forall t=1,\ldots,T \tag{3}$$

$$z_r^t \leq \sum_{j\in\mathcal{L}_k} \beta_{jr} y_j \qquad \forall k=1,\ldots,L-1, r\in\mathcal{R}_{k+1}, \forall t=1,\ldots,T \tag{4}$$

$$\sum_{r\in\mathcal{R}_k} z_r^t \leq \nu_k \qquad \forall k=1,\ldots,L, \forall t=1,\ldots,T \tag{5}$$

$$\sum_{i\in\mathcal{L}_k} u_{ir}^t \leq \kappa_k z_r^t \qquad \forall k=1,\ldots,L, \forall r\in\mathcal{R}_k, t=1,\ldots,T \tag{6}$$

$$I_j^{t-1} + \sum_{r\in\mathcal{R}_k} u_{jr}^t = I_j^t + \sum_{r'\in\mathcal{R}_{k+1}}\left(\beta_{jr'} \sum_{i\in\mathcal{L}_{k+1}} u_{ir'}^t\right) \qquad \forall k=1,\ldots,L-1, \forall j\in\mathcal{L}_k, \forall t=1,\ldots,T \tag{7}$$

$$I_i^{t-1} + \sum_{r\in\mathcal{R}_L} u_{ir}^t = I_i^t + d_i^t \qquad \forall i\in\mathcal{C}, \forall t=1,\ldots,T \tag{8}$$

$$I_j^t \leq I_j^{\max} y_j \qquad \forall k=1,\ldots,L-1, \forall j\in\mathcal{L}_k, \forall t=1,\ldots,T \tag{9}$$

$$I_i^t \leq \min\left(I_i^{\max}, \sum_{t'\geq t} d_i^{t'}\right) \qquad \forall i\in\mathcal{C}, \forall t=1,\ldots,T \tag{10}$$

$$u_{ir}^t \geq 0 \qquad \forall k=1,\ldots,L, \forall i\in\mathcal{L}_k, \forall r\in\mathcal{R}_k, \forall t=1,\ldots,T \tag{11}$$

$$I_i^t \geq 0 \qquad \forall i\in\mathcal{L}, \forall t=1,\ldots,T \tag{12}$$

$$y_j \in \{0,1\} \qquad \forall j\in\mathcal{L}\setminus\mathcal{C} \tag{13}$$

$$z_r^t \in \{0,1\} \qquad \forall r\in\mathcal{R}, \forall t=1,\ldots,T \tag{14}$$

The objective function (1) aims at minimizing the total cost incurred by the system. Constraints (2) and (3) state that every (open) location is served by at most one route $r \in \mathcal{R}$ serving the corresponding level in every period, respectively. Constraint (4) ensures that routes start only from opened depots. Constraints (5) states that the number of routes used to serve the locations of a given level k in period t does not exceed the fleet size allocated to the routes $r \in \mathcal{R}_k$. Constraint (6) ensures that the sum of the quantities deliverd through a route $r \in \mathcal{R}_k$ in period t is lower than the capacity of a vehicle. Constraints (7) and (8) define the units flow through the depots and customers, respectively. Finally, constraints (9) and (10) ensure that the capacity constraint on the inventory at the depots and the customers are satisfied in any period.

2.3 Restrictions

Although the MIP formulation from § 2.2 deals with a rather general version of the problem, we focus on two particular cases:

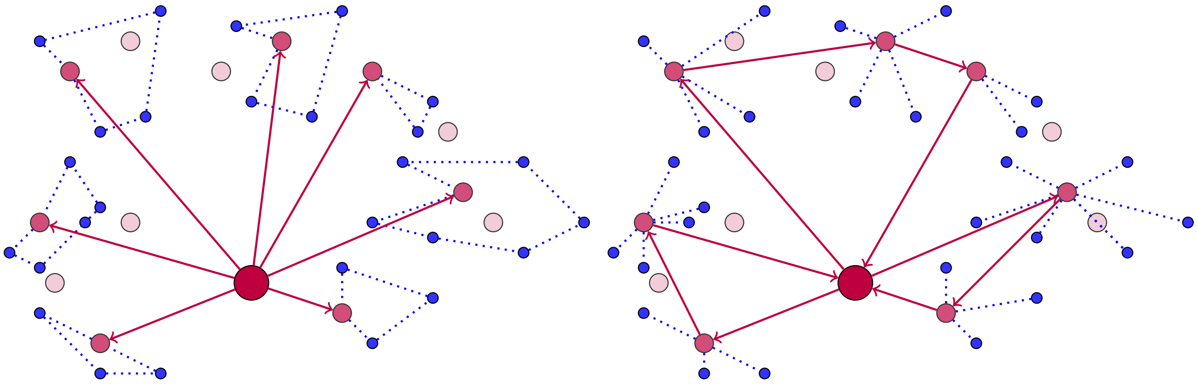
1. The first one considers a model "Direct+Loop" in which the set $\mathcal{R}_k$ consists only of direct routes between the supplier and the depots. That is, a route $r \in \mathcal{R}_k$ can be described by two arcs (s, j) and (j, s), where $j \in \mathcal{R}_k$ and s is the supplier.

2. The second one, called "Loop+Direct" considers a set $\mathcal{R}_L$ containing only direct routes between depots and customers. In other words, a route $r \in \mathcal{R}_L$ can be described by two arcs (j, i) and (i, j) with $j \in \mathcal{L}_k$ and $i \in \mathcal{C}$. Figure 1b.

Figure 1a and 1b illustrates the two types of structures related to these models.

3 Matheuristic method

The matheuristic method we propose for the LIRP is based on a sampling of the routes. After being shuffled, the original set of routes is splitted into α independent subsets. Each subset is solved with Cplex, leading to α



(a) Model “Direct-Loop” (b) Model “Loop-Direct”

Figure 1: The two types of routing structures considered

suboptimal LIRP solutions. The routes contained in each suboptimal solution are stored in a pool of *promising routes*. Finally, the pool of promising routes is used to create a new (hopefully tractable) instance which solved with Cplex. Algorithm 1 describes the procedure.

Algorithm 1 The proposed sampling matheuristic

Require: The set R of all routes generated
Require: A parameter α (number of subsets)
Require: $t1$ and $t2$: CPU allocated to Cplex in the inner and outer loop of the algorithm respectively
Require: J : parameter

$z^* = +\infty$
$collectedRoutes \leftarrow \emptyset$
repeat
  $amelioration = FALSE$
  $R = R \backslash collectedRoutes$
  Shuffle all routes in R
  $LocationSampling(R, \alpha)$ {partitions R into at most α subset of routes, regrouped by starting depots}
  $RouteSampling(R, \alpha)$ {partition R into α independent subsets of routes}
  for $s = 1$ to α **do**
    R_s = routes in subset s
    $bestRoutes_s = \emptyset$
    for $j = 1$ to J **do**
      $R_s \leftarrow R_s \backslash bestRoutes_s$ {remove the route just found, active when $J > 1$}
      $(z, bestRoutes_s) = SolveLIRP(R_s, t1)$
      $collectedRoutes \leftarrow bestRoutes_s$
    end for
  end for
  $(z, bestRoutes) = solveLIRP(collectedRoute, t2)$
  if $z < z^*$ **then**
    $z^* = z$
    $amelioration = TRUE$
  end if
until $amelioration = FALSE$
return $bestRoutes$

3.1 Location based sampling

Location Sampling s’effectue en deux temps. Tout d’abord, les clients sont pré-affectés à un certain nombre de dépôts, comme décrit dans l’algo 2. Ensuite, cette pré-affectation est utilisée pour constituer des sous-ensembles de routes, comme décrit dans l’algo 3

Algorithm 2 Pre-allocations of clients to depots

Require: D: set of depots
Require: C: set of clients
Require: $N_{close}, \mu_1, \mu_2, \mu_3$: parameters

Initialization: A: $|C| \times |D|$ allocation matrix filled with zeros
Initialization: $dist$: $|C| \times |D| + |C|$ distance matrix between clients and other clients + depot

for all clients $c \in C$ **do**
  for all depots $d \in D$ **do**
    calculate the distance $dist(c, d)$ between c and d
  end for
end for

for all clients $c \in C$ **do**
  $ClosestDepots(c)$ = List of all depots $d \in D$ ranked in non-decreasing order of the distances $dist(c, d)$.
  $A(c, ClosestDepots(c)[1]) = 1$ {pre-allocate client c to its closest depot}
  for $n = 2$ to N_{close} **do**
    if $dist(c, ClosestDepots(c)[n]) < \mu_1$ **then**
      $A(c, ClosestDepots(c)[n]) = 1$ {pre-allocate client c to depot ranked n}
    end if
  end for
end for

for all clients $c \in C$ **do**
  Rank all clients $c' \in C$, $c' \neq c$ in non-decreasing order of the distances $dist(c, c')$
  for all clients c' **do**
    if $dist(c, c') < \mu_2$ **then**
      for all $d \in ClosestDepots(c')$ such that $A(c', d) = 1$ **do**
        if $A(c, d) = 0$ and $dist(d, c') + dist(c', c) + dist(c, d) < \mu_3$ **then**
          $A(c, d) = 1$
        end if
      end for
    end if
  end for
end for
return A

Algorithm 3 Heuristic selection of depots

Require: D: set of depots
Require: C: set of clients
Require: $PreA$: pre-allocation matrix
Require: p: number of depots to be selected
Require: β: parameter strictly greater than 1.
Initialize: A allocation matrix to zero
$s \leftarrow 0$: number of depots selected
$nbClient \leftarrow 0$ number of clients assigned to a depot
$S \leftarrow \emptyset$: list of selected depots
while $s < p$ and $nbClient < |C|$ **do**
  for each depot $d \in D$ **do**
    $score(d)$ = number of clients pre-allocated to d
  end for
  Rank all depots by non-increasing scores
  Choose a random number y from the interval $[0, 1)]$
  $S = S \cup \{D[\ \lceil y^{\beta}|D|\rceil]\ \}$.
  $s \leftarrow s + 1$
  for all clients $c \in C$ **do**
    if $PreA[c][d] = 1$ **then**
      $A[c][d] = 1$
      $nbClient \leftarrow nbClient + 1$
      for all depots $d \in D$ **do**
        $PreA[c][d] = 0$
      end for
    end if
  end for
end while
if $nbClient < |C|$ **then**
  Add a dummy depot δ
  for all $c \in C$ such that $\sum_{d \in D} PreA[c][d] > 0$ **do**
    $A[c][\delta] = 1$
  end for
end if
return S
return A

3.2 Filtering SD routes after depot selection

Algorithm 4 SD Routes filtering

Require: $SDloop$: set of SD routes
Require: D set of depots
Require: C: set of clients
Require: A: allocation matrix

$filteredSD \leftarrow \emptyset$
for all $d \in D$ **do**
  for all $c \in C$ **do**
    if $A[c][d] = 1$ **then**
      $filteredSD = filteredSD \cup \{s, d\}$
      $c = nbclient$
    end if
  end for
end for
return $filteredSD$

3.3 Filtering DC routes after depot selection

Algorithm 5 DC Routes filtering

Require: $DCloop$: set of DC routes
Require: A: allocation matrix
Require: C: set of clients
Require: D set of depots

$filteredDC \leftarrow \emptyset$
for each route $r \in DCloop$ **do**
  $keep = 1$
  for all $c \in r$ **do**
    if $A[c][depot(r)] = 0$ **then**
      $keep = 0$
      $c \leftarrow$ end of the route
    end if
  end for
  if $keep = 1$ **then**
    $filteredDC = filteredDC \cup \{r\}$
  end if
end for
return $filteredDC$

References