# General variable neighborhood search for the multi-product dynamic lot sizing problem in closed-loop supply chain

Angelo Sifaleras [a,1] Ioannis Konstantaras [b,2]

[a] *Department of Applied Informatics, School of Information Sciences, University of Macedonia, 156 Egnatia Str., Thessaloniki 54636, Greece*

[b] *Department of Business Administration, School of Business Administration, University of Macedonia, 156 Egnatia Str., Thessaloniki 54636, Greece*

**Abstract**

The multi-product dynamic lot sizing problem with product returns and recovery is an important problem that appears in reverse logistics and is known to be NP-hard. In this paper we suggest a General Variable Neighborhood Search (GVNS) metaheuristic algorithm for solving this problem. It is the first time that such an approach is used for this problem in the literature. Furthermore, we present some encouraging computational results obtained on a new set of very large benchmark instances, compared with Gurobi optimizer.

*Keywords:* Variable Neighborhood Search, Mathematical Programming, Dynamic Lot Sizing, Remanufacturing

[1] Email: sifalera@uom.gr
[2] Email: ikonst@uom.gr

Please cite this paper as:

Sifaleras A. and Konstantaras I., "General variable neighborhood search for the multi-product dynamic lot sizing problem in closed-loop supply chain", Electronic Notes in Discrete Mathematics, Elsevier B.V., Vol. 47, pp. 69-76, 2015.

The final publication is available at Elsevier via http://dx.doi.org/10.1016/j.endm.2014.11.010

# 1 Introduction

Inventory control and dynamic lot sizing problems have been most extensively researched topics in production and inventory management. A very detailed review paper on inventory lot sizing problems is given by Ullah and Parveen [6]. One of the most important issues that should be tackled by a company is the determination of the replenishment policy of the different goods (spare parts, raw material, components or finished goods) involved in the supply chain. This problem becomes more complex when the demand for each of the $K$ products varies with time through a finite planning horizon with $T$ periods.

Due to the increasing attention on the sustainability in industry, inventory control problems have been studied in the field of closed-loop systems. Manufacturers have started to integrate remanufacturing facilities into the regular production environment. The Multi-product Dynamic Lot Sizing Problem with Remanufacturing activities (MDLSRP) is an important problem that appears in closed-loop supply chain in the last years. Recently, Variable Neighborhood Search (VNS) has been succesfully applied for the economic lot sizing problem with product returns and recovery [5]. In this paper we extend that work and present some first results on the application of VNS to the MDLSRP. We propose a General Variable Neighborhood Search (GVNS) metaheuristic algorithm [2] for solving the multi-product uncapacitated dynamic lot-sizing problem with remanufacturing. Although, VNS has also been applied to other inventory problems, this is the first time that it is used for this particular problem in the literature. The rest of the paper is organized as follows: Section 2 provides the basic formulation of the problem. In Section 3, the GVNS approach and the neighborhood structures that are used, are briefly described. Section 4 exposes the obtained results, and the paper concludes with Section 5.

# 2 Model formulation

The problem discussed in this paper is to satisfy the demand of items in each period at the lowest total cost possible. The demand is given for a finite planning horizon and is assumed to be not stationary. It can be satisfied by both manufactured new items and remanufactured returned ones (both known as "serviceables"). Also the number of returns is known for all periods and assumed to be not stationary. The returns can be completely remanufactured and sold as new one. The problem studied in our paper consists of the multi-product dynamic lot sizing model with both remanufacturing and manufac-

turing setup costs; a modified version of the model introduced by Sahling [4], without capacities. The lot sizing problem under separate manufacturing and remanufacturing set up costs is suitable for situations where there are separate production lines, one for manufacturing and one for remanufacturing. The aim is to determine the number of remanufactured and manufactured items per period, in order to minimize the sum of set up costs of the manufacturing and remanufacturing processes and holding costs for returns and serviceables under various operational constraints. Before presenting the formulation of the MDLSRP, we first introduce some notations:

$k$: product, $k = 1, 2, \ldots, K$.
$t$: time period, $t = 1, 2, \ldots, T$.
$D(k,t)$: demand for product $k$ in time period $t$.
$R(k,t)$: number of returned items of product $k$ in period $t$ that can be completely remanufactured and sold as new.
$h_M(k)$: holding cost for the serviceable items of product $k$ per unit time.
$h_R(k)$: holding cost for the recoverable items of product $k$ per unit time.
$z_M(k,t)$: binary decision variable denoting the initiation of a manufacturing lot of product $k$ in period $t$.
$z_R(k,t)$: binary decision variable denoting the initiation of a remanufacturing lot of product $k$ in period $t$.
$x_M(k,t)$: number of manufactured items of product $k$ in period $t$.
$x_R(k,t)$: number of items that are eventually remanufactured of product $k$ in period $t$.
$k_M(k)$: manufacturing setup cost of product $k$.
$k_R(k)$: remanufacturing setup cost of product $k$.
$p_M(k)$: production cost of product $k$ per unit.
$p_R(k)$: remanufacturing cost of product $k$ per unit.
$y_M(k,t)$: inventory level of serviceable items of product $k$ in period $t$.
$y_R(k,t)$: inventory level of remanufacturable items of product $k$ in period $t$.
$M$: A sufficiently large number.

The MDLSRP can be modeled as a Mixed Integer Linear Programming (MILP) problem as follows [4]:

$$\begin{aligned} \min z = & \sum_{k=1}^{K}\sum_{t=1}^{T}\left(h_R(k)y_R(k,t) + h_M(k)y_M(k,t)\right) \\ & + \sum_{k=1}^{K}\sum_{t=1}^{T}\left(p_R(k)x_R(k,t) + p_M(k)x_M(k,t)\right) \\ & + \sum_{k=1}^{K}\sum_{t=1}^{T}\left(k_R(k)z_R(k,t) + k_M(k)z_M(k,t)\right) \end{aligned} \tag{1}$$

where:

$$z_R(k,t) = \begin{cases} 1, \text{ if } x_R(k,t) > 0, \\ 0, \text{ otherwise,} \end{cases} \qquad z_M(k,t) = \begin{cases} 1, \text{ if } x_M(k,t) > 0, \\ 0, \text{ otherwise,} \end{cases} \tag{2}$$

are binary decision variables denoting the initiation of a remanufacturing or manufacturing lot, respectively.

$$\begin{aligned} y_R(k,t) &= y_R(k,t-1) + R(k,t) - x_R(k,t), \\ y_M(k,t) &= y_M(k,t-1) + x_R(k,t) + x_M(k,t) - D(k,t), \\ &\forall t = 1,2,\ldots,T, \quad \forall k = 1,2,\ldots,K. \end{aligned} \tag{3}$$

$$\begin{aligned} y_M(k,t) &\geq \sum_{s=t+1}^{t+p} D(k,s) - \sum_{s=t+1}^{t+p} M(k,s)\,(z_M(k,s) + z_R(k,s)), \\ &\forall k, \quad \forall t = 1,\ldots,T-1, \quad \forall p = 1,\ldots T-t, \\ y_R(k,t) &\geq \sum_{s=t-p}^{t} R(k,s) - \sum_{s=t-p}^{t} M(k,s) z_R(k,s), \\ &\forall k, \quad \forall t = 2,\ldots,T, \quad \forall p = 1,\ldots t-1, \end{aligned} \tag{4}$$

$$\begin{aligned} x_R(k,t) \leq M z_R(k,t), \qquad x_M(k,t) \leq M z_M(k,t), \\ \forall t = 1,2,\ldots,T, \quad \forall k = 1,2,\ldots,K. \end{aligned} \tag{5}$$

$$\begin{aligned} &y_R(k,t), y_M(k,t), x_R(k,t), x_M(k,t) \geq 0, \quad z_R(k,t), z_M(k,t) \in \{0,1\}, \\ &y_R(k,0) = y_M(k,0) = 0, \quad \forall t = 1,2,\ldots,T, \quad \forall k = 1,2,\ldots,K. \end{aligned} \tag{6}$$

The constraints defined in Equations (3) are the inventory balance equations which compute the inventory of returns and serviceables, respectively. Equations (2) ensure that a fixed setup cost is incurred when remanufacturing or manufacturing takes place, respectively. Equations (6) ensure that the inventories are initially empty, set the indicator variables and prevent negative (re)manufacturing or inventory. Finally, Equations 4 are the adapted valid inequalities for the MDLSRP proposed in [3].

## 3 Neighborhood structures & General VNS (GVNS)

A neighborhood $N(\sigma)$ is a set of neighboring solutions in the solution space that is defined as $N(\sigma) = \{$solution $\sigma'$ obtained by applying one or more changes to $\sigma\}$. We have used four different neighborhoods (i.e., $k_{max} = |N| = 4$), based on shifting the values of each binary variable.

$N_1(\sigma) = \{\sigma' \mid \sigma'$ is obtained by shifting the set-up for remanufacturing $z_R(k,t)$ in one period from 1 to 0 (or reduce only that $x_R(k,t)$ variable), by strictly increasing some $y_R(k,t)$ variables and reducing some $y_M(k,t)$ variables, for some $k = 1,2,\ldots,K$, $t = 1,2,\ldots,T\}$.

$N_2(\sigma) = \{\sigma' \mid \sigma'$ is obtained by shifting the set-up for remanufacturing $z_R(k,t)$ in one period from 0 to 1, by strictly reducing some $y_R(k,t)$, $y_M(k,t)$, and $x_M(k,t)$ variables, for some $k = 1, 2, \ldots, K$, $t = 1, 2, \ldots, T\}$.

$N_3(\sigma) = \{\sigma' \mid \sigma'$ is obtained by shifting the set-up for manufacturing $z_M(k,t)$ for one product $k$ in one period from 1 to 0, by increasing only some $y_M(k,t)$ variables and a previous $x_M(k,t)$ variable, for some $k = 1, 2, \ldots, K$, $t = 1, 2, \ldots, T$, and $i < t\}$.

$N_4(\sigma) = \{\sigma' \mid \sigma'$ is obtained by shifting the set-up for manufacturing $z_M(k,t)$ for one product $k$ in one period $t$ from 0 to 1, by reducing only some $y_M(k,t)$ variables and a $x_M(k,t)$ variable, for some $k = 1, 2, \ldots, K$, $t = 1, 2, \ldots, T$, and $i < t\}$.

The pseudocode of the proposed GVND algorithm (`GVND`) is as follows.

**Algorithm 1** GVND

```
1: procedure GVND(K, T, R, D, h_R, h_M, k_R, k_M, p_R, p_M, σ, k_max)
2:     repeat
3:         improvement ← 0
4:         for k ← 1, k_max do
5:             for i ← 1, K do
6:                 for t ← 1, T do
7:                     Find the best neighbor σ' of σ (σ' ∈ N_k(σ))
8:                     if the obtained solution σ' is better than σ then
9:                         Set σ ← σ'
10:                        Set improvement ← 1
11:                    end if
12:                end for
13:            end for
14:        end for
15:    until improvement = 0
16: end procedure
```

The pseudo-code of the proposed GVNS scheme (`GVNS`) is as follows:

**Algorithm 2** GVNS

```
1: procedure GVNS(K, T, R, D, h_R, h_M, k_R, k_M, σ, k_max)
2:     Apply the Heuristic_Start method
3:     while time < time limit do
4:         Apply the GVND method
5:         Shake N_k (k ∈ 1 ... k_max)
6:     end while
7: end procedure
```

A simple constructive heuristic has been implemented, where the total demand is fulfilled by a single lot (without remanufacturing units) for each

product in the first period, in order to find an initial solution to the MDLSRP.

**Algorithm 3** Heuristic initialization method

**procedure** HEURISTIC_START($K, T, R, D, x_R, x_M, y_R, y_M, z_R, z_M$)
  $x_R(:,:),\ z_R(:,:),\ x_M(:,:),\ z_M(:,:),\ y_R(:,:),\ y_M(:,:) \leftarrow 0$
  $y_R(:,1) \leftarrow R(:,1)$
  **for** $i \leftarrow 1, K$ **do**
    **for** $j = 2, T$ **do**
      $y_R(i,j) \leftarrow y_R(i,j-1) + R(i,j)$
      $y_M(i, T+1-j) \leftarrow y_M(i, T+2-j) + D(i, T+2-j)$
    **end for**
  **end for**
  $x_M(:,1) \leftarrow y_M(:,1) + D(:,1)$
  $z_M(:,1) \leftarrow 1$
**end procedure**

## 4 Experimental results

Our GVNS metaheuristic algorithm was applied on a new set of benchmark problems with 300 products, 52 periods, and varying parameters such as $h_R(k) = \{0.2, 0.5, 0.8\}$, $k_R(k) = \{200, 500, 2000\}$, and $k_M(k) = \{200, 500, 2000\}$; thus, totally 27 different instances. It is noteworthy, that this new benchmark set is the larger that have ever been used in the literature, for the MDLSRP problem. Each instance consists of 93600 variables (31200 binary and 62400 linear variables), 858000 linear constraints, and 1 linear objective.

We ran the experiments on a computer running Ubuntu Linux 14.04 64Bit with an Intel Core i7 4770K CPU at 3.5 GHz with 8 MB L3 cache and 32 GB DDR3 1600MHz main memory. The GVNS implementation was implemented in Fortran and compiled using the Intel Fortran 64 compiler XE v.14.0.1.106. Efforts were made to solve each one of them using the latest version of the state-of-the-art Gurobi optimizer v5.6.3 within a reasonable amount of time set to 1 hour. The time limit (stopping condition) of our GVNS approach was 30 seconds. However, due to the increased computational difficulty, Gurobi was not able to solve any instance to optimality. Thus, Gurobi has only computed an upper bound of the optimal objective value. Therefore, Table 1 reports the comparative numerical results regarding the incumbent solutions found by Gurobi (time limit = 3600 secs) and the proposed GVNS implementation (time limit = 30 secs). In the last row of Table 1, we provide the number of times (wins) that each of the methods managed to obtain the best objective value (displayed in bold fonts) against the other.

Table 1
Numerical results

| No | Gurobi 5.6.3 | GVNS | No | Gurobi 5.6.3 | GVNS |
|---|---|---|---|---|---|
| 1 | 3.821.398,80 | 4.418.976,00 | 15 | 9.641.900,20 | 12.585.880,00 |
| 2 | 4.058.656,00 | 7.075.118,50 | 16 | 9.653.241,40 | 10.536.628,00 |
| 3 | 4.801.976,80 | 9.731.397,00 | 17 | 14.665.659,50 | **13.362.837,00** |
| 4 | 4.540.967,60 | 5.983.343,50 | 18 | 18.526.225,20 | **16.220.067,00** |
| 5 | 4.462.903,50 | 8.639.642,00 | 19 | 21.685.001,60 | **9.205.697,00** |
| 6 | 5.102.224,40 | 11.295.788,00 | 20 | 24.298.360,00 | **11.860.608,00** |
| 7 | 5.149.999,20 | 9.235.279,00 | 21 | 26.472.176,80 | **14.518.819,00** |
| 8 | 8.518.942,00 | 12.065.381,00 | 22 | 27.680.000,20 | **10.767.096,00** |
| 9 | 11.707.019,60 | 14.920.090,00 | 23 | 28.077.188,50 | **13.424.938,00** |
| 10 | 7.345.698,80 | **5.709.826,50** | 24 | 26.968.400,20 | **16.083.138,00** |
| 11 | 8.016.488,50 | 8.364.853,50 | 25 | 37.691.000,20 | **14.051.599,00** |
| 12 | 7.737.800,20 | 11.021.917,00 | 26 | 36.364.056,00 | **16.869.502,00** |
| 13 | 9.621.500,20 | **7.273.325,00** | 27 | 38.485.376,80 | **19.733.662,00** |
| 14 | 9.455.556,00 | 9.930.458,00 | Wins | 14 | 13 |

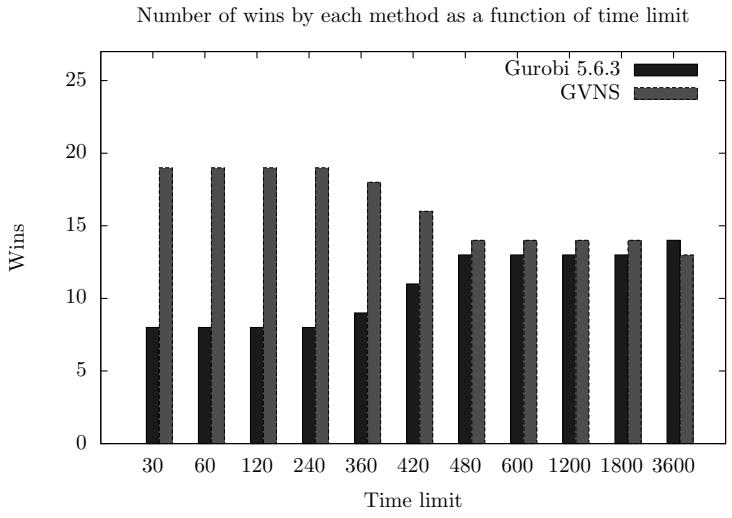


Fig. 1. Number of wins by each method as a function of time limit

As it is shown in Table 1 the proposed GVNS metaheuristic approach found better solutions than Gurobi in the 13 out of the 27 instances. However, the GVNS method required approximately 120 times less computational time for reaching these solutions. Furthermore, the mean absolute percentage error of Gurobi, regarding the 13 instances where GVNS performed better, was 86.18%; whereas the corresponding error of GVNS, for the remaining 14 instances where Gurobi performed better, was only 48.52%. Also, Figure 1 shows how the imposed time limit influences the number of best objective values computed by each method. It appears that, the GVNS approach outperforms Gurobi if the time limit is less than 1h. Thus, one can conclude that the GVNS approach is the best method for computing good quality solutions for large MDLSRP instances in a small amount of time.

## 5 Conclusions and Future work

In this paper we have addressed the multi-product dynamic lot sizing problem with product returns and recovery in reverse logistics. We have proposed a new GVNS metaheuristic algorithm for exploring four neighborhoods for this hard combinatorial optimization problem. Based on preliminary numerical testing using a new set of benchmark problems with 27 instances, our approach was able to achieve very encouraging results on large problems with 300 products and 52 periods. Moreover, it would be interesting to develop a parallel implementation of our proposed GVNS method using MPI [1].

## References


[1] Davidović, T. and T. G. Crainic, *MPI parallelization of variable neighborhood search*, Electronic Notes in Discrete Mathematics **39** (2012), pp. 241–248.

[2] Hansen, P., N. Mladenović and J. A. Moreno Pérez, *Variable neighbourhood search: methods andapplications*, Annals of Operations Research **175** (2010), pp. 367–407.

[3] Retel Helmrich, M. J., R. Jans, W. van den Heuvel and A. P. Wagelmans, *Economic lot-sizing with remanufacturing: complexity and efficient formulations*, IIE Transactions **46** (2014), pp. 67–86.

[4] Sahling, F., *A column-generation approach for a short-term production planning problem in closed-loop supply chains*, Business Research **6** (2013), pp. 55–75.

[5] Sifaleras, A., I. Konstantaras and N. Mladenović, *Variable neighborhood search for the economic lot sizing problem with product returns and recovery*, to appear in International Journal of Production Economics (2015).

[6] Ullah, H. and S. Parveen, *A literature review on inventory lot sizing problems*, Global Journal of Researches In Engineering **10** (2010), pp. 21–36.