A Particle Swarm Optimization method based on cost modification heuristic for the Multi Level Lot Sizing Problem

Arnaud Laurent, David Lemoine

Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnline CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 55-10 (2022) 1243–1248

A Particle Swarm Optimization method based on cost modification heuristic for the Multi Level Lot Sizing Problem

Arnaud. Laurent * **David. Lemoine** **

* *IMT Atlantique, LS2N (e-mail: arnaud.laurent@imt-atlantique.fr).*
** *IMT Atlantique, LS2N (e-mail:david.lemoine@imt-atlantique.fr)*

Abstract: In this paper we provide a metaheuristic method based on Particle Swarm Optimization to solve the uncapacitated Multi Level Lot Sizing Problem. The PSO is based on a cost modification heuristic. Eventually, the final solution is refined with a deterministic local search. The whole method is tested on a set of small and medium size instances of the literature. The global results are promising and we managed to improve the upper bound of one of the instance.

Keywords: Multi Level Lot Sizing Problem, Particle Swarm Optimization, Metaheuristic, Local Search

1. INTRODUCTION

Lot sizing problem aims to determine the optimal planning in a production system with the best trade-off between setup and holding costs. The uncapacitated mono-level problem has been defined and solve optimally in polynomial time by Wagner and Whitin (1958). This resolution method is usually called Wagner-Whithin (WW) algorithm. The Multi Level Lot-Sizing Problem (MLLS) consider the whole Material Request Planning (MRP) to determine the optimal planning for all the items and their components. Contrary to the one level lot sizing problem, the MLLS was proven strongly NP-hard by Arkin et al. (1989).

In order to solve this problem, Afentakis and Gavish (1986) propose a branch and bound algorithm using the echelon stock notion introduced by Clark and Scarf (1960). More works dealing with the MLLS developed approximate approaches such as heuristics or metaheuristics. Several heuristics solve the problem sequentially, following the bill of material. First historical approaches consist in applying mono-level heuristics or optimal algorithm iteratively. The heuristic start by fixing the top level items and then fix iteratively their components, level by level such as Yelle (1979) with mono-level heuristics and Veral and LaForge (1985) for the resolution with the Wagner-Whithin algorithm. Following the same iterative solving process, some works developed cost modification heuristics. The principle stay the same as Veral and LaForge (1985) but the difference is that the price considered to apply the Wagner-Whithin algorithm are modified to consider the sub-item cost. Several works propose different pricing strategy such as Blackburn and Millen (1982), Bookbinder and Koch (1990) and Dellaert and Jeunet (2000).

Finally, some resolution methods use metaheuristics to solve the MLLS problem. Dellaert and Jeunet (2000) designed a genetic algorithm, Pitakaso et al. (2007) developed a min-max ant colony resolution method, Tang (2004), Kuik and Salomon (1990) and Jeunet and Jonard (2005) proposed simulated annealing approaches to solve the MLLS. The Particle Swarm Optimization (PSO) method has already been used to solve the MLLS. Han et al. (2009) defined a discrete version of the PSO to solve the MLLS. Conversely, Deroussi and Lemoine (2013) use the continuous PSO to modify the holding and setup prices. The solution are obtain by solving each level with WW is then applied at each step with the prices set with the PSO. With this approach, they improved the results obtained with the DPSO on the same set of small instances used by Han et al. (2009). However the method still lack of efficiency for the larger instances. Xiao et al. (2014) proposed a Variable Neighborhood Search (VNS) using many good properties to reduce the solution space. They also provide a really efficient method for quickly calculate the cost variation between two solutions. This method is really efficient and improved many of the best known solutions from literature data-sets.

We propose in this paper a PSO method based on a cost-modification heuristic. We present in the Section 2 the problem and its mathematical model. In Section 3 we present the PSO algorithm, the solution representation and the evaluation algorithm. We also discuss the optimal accessibility to the optimal solution and propose a local search to address this issue. Eventually in Section 4 we present the results obtained with our method on two classical sets of instances.

2. PROBLEM

In this section we describe the problem, its model and the properties we use to solve it. We define in the Table 1 all the notations used in this paper.

The mathematical model corresponding to the MLLS problem is the following:

10.1016/j.ifacol.2022.09.560

Model	
N	Number of items
T	Number of periods
$D_{i,t}$	Demand for item i at period t
$a_{j',j}$	Number of item j required to produce one item j'
h_i	Holding cost for item i
s_i	Setup cost for item i
M	A very large number
$Q_{i,t}$	Production quantity of item i at period t
$I_{i,t}$	Inventory for item i at period t
$Y_{i,t}$	Binary variable indicating if a production is set in period t for item i
PSO	
P	Number of particles in the swarm
$pBest_p$	Best solution founded by particles p
$lBest_p$	Best solution founded by the neighbors of particles p
L	Number of neighbors considered for each particle
$MaxIte$	Maximum number of iterations
$v_{p,k}$	The speed of particle p at iteration k
$x^{p,k}$	The position/solution of particle p at iteration k
c_1	The coefficient of influence of the $pBest$ solution
c_2	The coefficient of influence of the $lBest$ solution
ω	The coefficient of inertia
r	A random value drawn uniformly in $[0, 1]$
Sub_i	Set of all direct and indirect sub-items of the item i

Table 1. Notations used in this paper

$$Minimize \sum_{t=1}^{T}\sum_{i=1}^{N}(h_i I_{i,t} + s_i Y_{i,t}) \quad (1)$$

$$\forall i, t \in [1, N] \times [1, T] :$$

$$I_{i,t} = I_{i,t-1} + Q_{i,t} - D_{i,t} - \sum_{j=1}^{N} a_{j,i} Q_{j,t} \quad (2)$$

$$Q_{i,t} \leq M Y_{i,t} \quad (3)$$

$$I_{i,t}, Q_{i,t} \in \mathbb{N}^2, Y_{i,t} \in \{0, 1\} \quad (4)$$

The objective function (2) represent the sum of the setup costs and holding costs of all the items. The constraint (2) represent the inventory balance. The constraint (3) insure that if a production is done for product i in period t, a setup cost is considered for this period.

3. RESOLUTION METHOD

3.1 The particle Swarm Optimization method

The Particle Swarm Optimization (PSO) is a metaheuristic aiming to solve continuous optimization problem proposed by Eberhart and Kennedy (1995). PSO has been widely adapted and used to solve combinatorial problems such as scheduling Farid et al. (2020) Xiong et al. (2019), bin packing Pandiselvi and Sivakumar (2021) Laurent and Klement (2019) or transportation problem Cansiz and Göçmen (2018) Wu et al. (2018).

The method consist in a set of P particles representing solutions, evolving through solution space by influencing each other. We distinguish two influences for a particle:

- The personal influence (pBest). This influence for a particle correspond to the best solution ever reached by the particle itself.
- The global or local influence (lBest). This influence correspond to the best solution founded by the other particles. The term topology is used to describe the graph of influence between the particles, El Dor et al. (2015) give an overview of different used topology. If all the particles are connected in the graph, the influence is called "global", otherwise if the influence only come from a given set of particle, the influence is called "local". The parameter L define the number of particle considered in the neighboring of each particle.

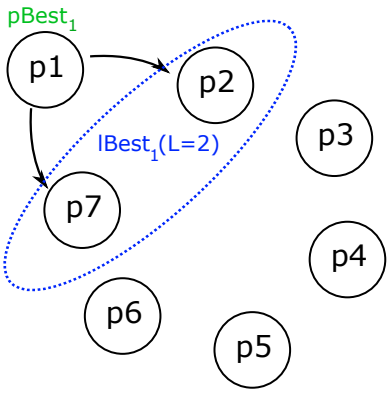

Fig. 1. The two influences of a particle

We give an illustration of both influences in Figure 1.

A particle is then defined by a solution (its position), a speed and its local neighbors. At each iteration k, all particles move in the search space depending on their personal and local influence and their inertia. The inertia correspond to the previous movement of the particle. The new position of particle p at iteration k is computed as equation (5) and the speed is computed as equation (6):

$$x^{p,k+1} = x^{p,k} + \overrightarrow{v_{p,k+1}} \quad (5)$$

$$\overrightarrow{v_{p,k+1}} = \omega\overrightarrow{v_{p,k}} + c_1 r_1 \overrightarrow{x^{p,k} pBest_p} + c_2 r_2 \overrightarrow{x^{p,k} lBest_p} \quad (6)$$

The three coefficients ω, $c1$ and $c2$ represent respectively the coefficient of inertia, the coefficient of personal influence and the coefficient of local influence. The value r_1 and r_2 are two random variables drawn uniformly in $[0, 1]$. The two values $\overrightarrow{x^{p,k} pBest_p}$ and $\overrightarrow{x^{p,k} lBest_p}$ represent respectively the distance vector from the current position of the particle p to the best past position (pBest) and the best past position of its neighbors (lBest).

The algorithm 1 describe the principle of PSO.

3.2 Solution encoding and decoding

A directed tree graph between items can be computed such that an arc from item i to item j exists only if item i requires at least one unit of item j ($a_{i,j} \geq 1$). An example of this graph is given in Figure 2.

We define the set A of all the items having no sub-item dependency, such that:

$$A = \{j \mid \forall j' \; \exists a_{j',j} \neq 0\} \quad (7)$$

The items in the set A are the leaf nodes (in dashed lines) of the graph of relation previously presented in Figure 2. The item i having no sub-items ($i \in A$), its optimal planning can be computed with Wagner Whithin algorithm

Algorithm 1. Algorithm of principle of the PSO

Require: $c_1, c_2, P, L, MaxIte$
  Initialize the swarm
  $ite \leftarrow 1$
  while $ite \leq MaxIte$ **do**
    for each particle p **do**
      Update the speed of p with equation (6)
      Update the solution of p with equation (5)
      Evaluate the new solution of particle p
      if The new solution is better than $pBest$ **then**
        $pBest_p \leftarrow$ The current solution
      end if
    end for
    for each particle p **do**
      for each particle p' in the L neighbors of p **do**
        if $pBest_{p'}$ is better than $lBest_p$ **then**
          $lBest_p \leftarrow pBest_{p'}$
        end if
      end for
    end for
  end while

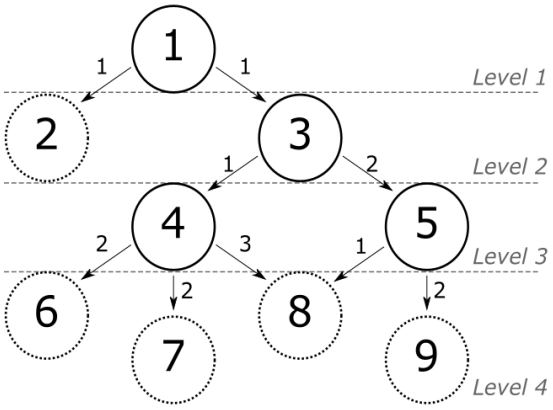

Fig. 2. Graph of items and their dependencies

(WW), with the demand for each period depending on the planning of the production of items requiring item i.

We represent a particle p at iteration k with a vector $x^{p,k}$ such that:

$$x^{p,k} = \{x_1^{p,k} x_i^{p,k} ... x_N^{p,k}\} \tag{8}$$

With $x_j^{i,k}$ representing a modified setup cost for all items $i \notin A$.

A solution for the MLLS is to give for each item i and for each period t, the quantity $Q_{i,t}$ produced. To obtain this solution corresponding to the particle i at step k the sequential approach heuristic described in Algorithm 2 is applied.

The evaluation algorithm first compute the optimal one level planning for each item with WW, from the top level item to the last level in the order of the bill of material (see Figure 2). The setup price considered at for each item is rather s_i if $i \in A$ or x_i as presented in Figure 3. Then, the total cost considering the real setup cost s_i for all items i is computed with the planning obtained from the sequential heuristic.

Algorithm 2. Algorithm of evaluation of a solution

Require: $h, s, x^{p,k}$
  for each item i ordered by increasing level **do**
    if item i requires at least one components **then**
      $Q_i \leftarrow WW(D_i, h_i, x_i^{p,k})$
      for each item j such that $a_{i,j} \geq 1$ **do**
        for each period t **do**
          $D_{j,t} \leftarrow D_{j,t} + Q_{i,t} \times a_{i,j}$
        end for
      end for
    else
      $Q_i \leftarrow WW(D_i, h_i, s_i)$
    end if
  end for
  Compute the cost H of the solution considering h, s and Q
  return H, Q

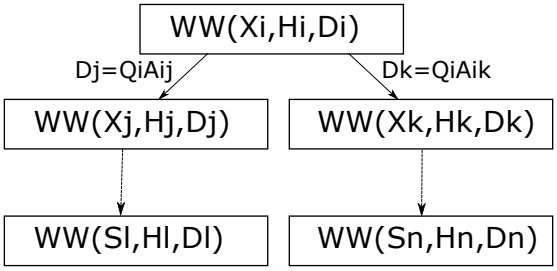

Fig. 3. Diagram of sequence to compute a solution from the setup prices X

3.3 Initialisation of the particles

The PSO method need an initialization of each particle. This initialization is important to help the method to reach a better solution, in a faster way. We fix the setup price $x_i^{p,0}$ of each item $i \notin A$ for each particle p as presented in equation 9.

$$x_i^{p,0} = (r_p + 0.5)(Sr_i) + s_i; \forall p = 1, .., P; \forall i = 1, .., N \tag{9}$$

$$Sr_i = \sum_{i' | a_{i,i'} \geq 1} x_{i'}^{p,0}; \quad \forall i = 1, .., N \tag{10}$$

With r_p a random value $\in [0, 1]$ to diversify the position of particles in the search space. For each item i, its setup is updated considering the setup cost of its sub-items.

3.4 Accessibility to optimal solution

The sequential approach heuristic used in this paper consider only solution for which the mono-level planning is reachable with WW algorithm. We present here a small example of an instance were the optimal solution cannot be reached with our PSO method.

Let define the following instance:

- T=3
- N=2
- Item 1
  - $D_1 = [20, 10, 9]$
  - $h_1 = 0.1$
  - $s_1 = 1$

  - $a_{1,2} = 1$
- Item 2
  - $D_2 = [1000, 0, 1000]$
  - $h_2 = 1000$
  - $s_2 = 1000$

The optimal solution here is driven by the high cost of item 2. The optimal production for item 1 is $[30, 0, 9]$ and $[1030, 0, 1009]$ for item 2. Because of the high costs on item 2, no production should be done on period 2 for item 1 (to avoid a setup cost or holding costs with item 2. However, the production $[30, 0, 9]$ is impossible to reach with a mono-level optimal algorithm, with any setup cost x_1 considered.

Indeed, if we consider the setup cost of item 1 to x_1 and its holding cost to h_1. The total cost of the planning $[30, 0, 9]$ is equal to $2x_1 + 10h_1$. However, the solution $[20, 19, 0]$ has an objective cost equals to $2x_1 + 9h_1$. For any holding cost $h_1 > 0$ and setup cost $x_1 > 0$, the second solution has always a lower cost from a mono-level perspective. Thus, the first solution cannot be reached with an optimal mono-level algorithm.

From this observation we can assert that it exists instances for which the optimal solution cannot be reached by our algorithm and by any cost modification heuristic with an optimal mono-level resolution.

To reduce the number of optimal solutions unreachable, we propose to apply a local search after the PSO method. We present this method in the next section.

3.5 Local search

When the stopping criterion of the PSO optimization is reached, we perform a Local Search (LS) to improve the best solution obtained. The principle of this LS is to test small modifications on the best solution find, in order to improve it. The algorithm 3 describe the behaviour of the LS, with Y^* the binary vector corresponding to the production periods and x^* the setup prices, both returned by the PSO. We denote in this algorithm the set Sub_i of all the direct or indirect sub-items of item i. An item j is included in the set Sub_i if and only if in the graph of relation (Figure 2) it exists a path from item i to item j.

The principle of this LS is simple, for each period of production of every items with components, we test to move this production to the previous period or to the next one. Then the mono-level optimal production planning is computed (with WW) for each sub-items concerned by the shift with the setup price x^* of the solution returned by the PSO.

We do not claim that this LS solve the accessibility issue pointed out in the previous section, however this mechanism improve the result obtained in a reasonable computation time.

4. RESULTS

We present in this section some results obtained with the proposed method. The method is implemented in Python and the experiment run using a processor 11th Gen Intel(R) Core(TM) i7-1185G7 @ 3.00GHz.

Algorithm 3. Algorithm of the local search performed after the PSO

Require: x^*, Y^*
  for each item i ordered by increasing level **do**
    if item i requires at least one components **then**
      for each period $t > 1$ such that $Y^*_{i,t} = 1$ **do**
                                ▷ *Move to the left*
        $Y' \leftarrow Y^*$
        $Y'_{i,t} \leftarrow 0$
        $Y'_{i,t-1} \leftarrow 1$
        for each item $j \in Sub_i$ **do**
          Compute D'_j considering Y'
          $Q'_j \leftarrow WW(D'_j, h_j, x^*_j)$
          Update Y'_j considering Q'_j
        end for
        Compute the cost of the new solution Y'
        if solution Y' has a lower cost **then**
          $Y^* \leftarrow Y'$
        else
                                ▷ *Move to the right*
          $Y' \leftarrow Y^*$
          $Y'_{i,t} \leftarrow 0$
          if $t < T$ **then**
            $Y'_{i,t+1} \leftarrow 1$
          end if
          for each item $j \in Sub_i$ **do**
            Compute D'_j considering Y'
            $Q'_j \leftarrow WW(D'_j, h_j, x^*_j)$
            Update Y'_j considering Q'_j
          end for
          Compute the cost of the new solution Y'
          if solution Y' has a lower cost **then**
            $Y^* \leftarrow Y'$
          end if
        end if
      end for
    end if
  end for
  return Y^*

4.1 Instances

We deal in this paper with two sets of instances used in the literature. The first one is a set of 96 small instances developed by Coleman and McKnew (1991) with 5 items and a horizon of 12 periods. The second set is composed of 40 medium size instances provided by Afentakis and Gavish (1986) and Dellaert and Jeunet (2000) structured with 40 to 50 items over 12 or 24 periods. For each instance considered, each method is run 10 times.

4.2 Parameters and stopping criterion

To test our method we test empirically different sets of parameters. We tested several topologies for the local influence on the particle and retained the circle topology defined by Mendes et al. (2004). We present here the parameters used to obtain the results presented in Section 4.3. We use the following parameters:

- Number of particle (P): 32
- Number of neighbors considered for each particle (L): 4

Method		Gap to Optimal (%)	Time (s)
Init	Min	1.390	0.02
	Max	1.908	
	Average	1.591	
init+PSO	Min	0.006	0.53
	Max	0.016	
	Average	0.007	
Init+PSO+LS	Min	0.000	0.55
	Max	0.014	
	Average	0.004	
	AvSTD	0.005	
Xiao et al. (2014)	1 run	0.000	0.1

Table 2. Results on the small size instances

Method		Gap to BKS (%)	Time (s)
Init	Min	1.388	0.18
	Max	1.964	
	Average	1.565	
init+PSO	Min	0.228	13.75
	Max	0.346	
	Average	0.272	
Init+PSO+LS	Min	0.099	14.30
	Max	0.281	
	Average	0.166	
	AvSTD	0.06	
Xiao et al. (2014)	1 run	0.05	30

Table 3. Results on the medium size instances

- The coefficient of influence of the $pBest$ solution (c_1): 1.0
- The coefficient of influence of the $lBest$ solution (c_2): 1.0
- The coefficient of inertia (ω): 0.8

We use two stopping criteria:

- The maximum number of iterations (MaxIte=1000) is reached
- No improvement on the best solution found is done during $noImp$ iterations. We use $noImp = 20$ for the small instances and $noImp = 40$ for the medium instances.

When a stopping criterion is reached, the LS presented in section 3.5 is executed. In order to reflect the performance of PSO alone and the relevance of the LS, we also provide the results obtained from the initialization method and with the PSO without the LS.

4.3 Performance of the different methods

In this part we report the performance of the different steps of the optimization method developed:

- The initialization method (see Section 3.3)
- The PSO (see Section 3.1)
- The LS executed when the PSO stop (see Section 3.5)

The results are presented in Table 2 for the small size instance and in Table 3 for the medium size instances. In this tables, we report the minimal, maximal and average value obtained for each step of the defined method. The gap to the Best Known Solution (BKS) is computed with equation 11, with res the value return by the method. For the small instances the BKS is optimal, but for the medium ones, most of the values are upper bounds. We consider the BKS reported by Xiao et al. (2014). We report the average computation time of one run over one instance in the column "Time". The $AvSTD$ value correspond to the average value of the standard deviation obtained over 10 runs for each instance. We also report the result obtained by Xiao et al. (2014) with the VNS giving the best results over the given set of instances.

$$\frac{BKS - res}{BKS} * 100 \tag{11}$$

We note that the result obtained with the developed method are quite encouraging. The method provide good quality solution in average and in a short amount of time. We improved the best known solution for the instance 21 in Xiao et al. (2014) with a value of 292,814 (BKS was previously 292,820).

5. CONCLUSION

We provide in this paper an efficient method to solve the Multi Level Lot Sizing problem. We used a PSO method based on a cost modification heuristic inspired from the work of Deroussi and Lemoine (2013). We improved the method by reducing the number of variables in order to converge toward good solutions in a short amount of time, even for large instances. After that, a local search is performed to improve locally the solution. We tested this method on small and medium size instances and the results were close to the best ones reported in the literature.

We do not provide in this study the results on large instances. This study is a first step in solving the MLLS, with the objective to deal with instances having a larger number of items. However, the first results obtained on the set of large instances proposed by Dellaert and Jeunet (2000) seems promising. Indeed, during the first test runs on this set, we managed to improve some of the best known solutions of the literature.

To improve the resolution method for the set of large instances we aims to strengthen the LS part of our method. Indeed the LS used in this study is quite simple and can be improved, based on the local search methods developed by Xiao et al. (2014).

REFERENCES

Afentakis, P. and Gavish, B. (1986). Optimal lot-sizing algorithms for complex product structures. *Operations research*, 34(2), 237–249.

Arkin, E., Joneja, D., and Roundy, R. (1989). Computational complexity of uncapacitated multi-echelon production planning problems. *Operations research letters*, 8(2), 61–66.

Blackburn, J.D. and Millen, R.A. (1982). Improved heuristics for multi-stage requirements planning systems. *Management Science*, 28(1), 44–56.

Bookbinder, J.H. and Koch, L.A. (1990). Production planning for mixed assembly/arborescent systems. *Journal of Operations Management*, 9(1), 7–23.

Cansiz, O.F. and Göçmen, S. (2018). Distance analysis of multimodal transportation based on traveling salesman problem with particle swarm optimization method. *International Journal of Advanced Engineering Research and Science*, 5(6).

Clark, A.J. and Scarf, H. (1960). Optimal policies for a multi-echelon inventory problem. *Management science*, 6(4), 475–490.

Coleman, B.J. and McKnew, M.A. (1991). An improved heuristic for multilevel lot sizing in material requirements planning. *Decision Sciences*, 22(1), 136–156.

Dellaert, N. and Jeunet, J. (2000). Solving large unconstrained multilevel lot-sizing problems using a hybrid genetic algorithm. *International Journal of Production Research*, 38(5), 1083–1099.

Deroussi, L. and Lemoine, D. (2013). Discrete particle swarm optimization for the multi-level lot-sizing problem. *Trends in Developing Metaheuristics, Algorithms, and Optimization Approaches*, 99–113.

Eberhart, R. and Kennedy, J. (1995). A new optimizer using particle swarm theory. In *MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, 39–43. Ieee.

El Dor, A., Lemoine, D., Clerc, M., Siarry, P., Deroussi, L., and Gourgand, M. (2015). Dynamic cluster in particle swarm optimization algorithm. *Natural Computing*, 14(4), 655–672.

Farid, M., Latip, R., Hussin, M., and Abdul Hamid, N.A.W. (2020). A survey on qos requirements based on particle swarm optimization scheduling techniques for workflow scheduling in cloud computing. *Symmetry*, 12(4), 551.

Han, Y., Tang, J., Kaku, I., and Mu, L. (2009). Solving uncapacitated multilevel lot-sizing problems using a particle swarm optimization with flexible inertial weight. *Computers & Mathematics with Applications*, 57(11-12), 1748–1755.

Jeunet, J. and Jonard, N. (2005). Single-point stochastic search algorithms for the multi-level lot-sizing problem. *Computers & operations research*, 32(4), 985–1006.

Kuik, R. and Salomon, M. (1990). Multi-level lot-sizing problem: Evaluation of a simulated-annealing heuristic. *European Journal of Operational Research*, 45(1), 25–37.

Laurent, A. and Klement, N. (2019). Bin packing problem with priorities and incompatibilities using pso: application in a health care community. *IFAC-PapersOnLine*, 52(13), 2596–2601.

Mendes, R., Kennedy, J., and Neves, J. (2004). The fully informed particle swarm: simpler, maybe better. *IEEE transactions on evolutionary computation*, 8(3), 204–210.

Pandiselvi, C. and Sivakumar, S. (2021). Performance of particle swarm optimization bin packing algorithm for dynamic virtual machine placement for the consolidation of cloud server. In *IOP Conference Series: Materials Science and Engineering*, volume 1110, 012007. IOP Publishing.

Pitakaso, R., Almeder, C., Doerner, K.F., and Hartl, R.F. (2007). A max-min ant system for unconstrained multi-level lot-sizing problems. *Computers & Operations Research*, 34(9), 2533–2552.

Tang, O. (2004). Simulated annealing in lot sizing problems. *International Journal of Production Economics*, 88(2), 173–181.

Veral, E.A. and LaForge, R.L. (1985). The performance of a simple incremental lot-sizing rule in a multilevel inventory environment. *Decision Sciences*, 16(1), 57–72.

Wagner, H.M. and Whitin, T.M. (1958). Dynamic version of the economic lot size model. *Management science*, 5(1), 89–96.

Wu, D., Huo, J., Zhang, G., and Zhang, W. (2018). Minimization of logistics cost and carbon emissions based on quantum particle swarm optimization. *Sustainability*, 10(10), 3791.

Xiao, Y., Zhang, R., Zhao, Q., Kaku, I., and Xu, Y. (2014). A variable neighborhood search with an effective local search for uncapacitated multilevel lot-sizing problems. *European Journal of Operational Research*, 235(1), 102–114.

Xiong, F., Gong, P., Jin, P., and Fan, J. (2019). Supply chain scheduling optimization based on genetic particle swarm optimization algorithm. *Cluster Computing*, 22(6), 14767–14775.

Yelle, L.E. (1979). Materials requirements lot sizing: A multi-level approach. *International Journal of Production Research*, 17(3), 223–232.