Hybrid genetic algorithm for a problem lot Sizing with transport

Assia LAGGOUN [1], Kinza Nadia MOUSS [2], Imen.DRISS [3]

1 Département de Génie Industriel, Université de BATNA .Algérie, a.laggoun12@yahoo.fr

2 Département de Génie Industriel, Université de BATNA .Algérie, kinz.mouss@yahoo.fr

3 Département de Génie Industriel, Université de BATNA .Algérie , idrissamina@hotmail.fr

Astract— Planning production problems have been the subject of many authors. We consider a lot of problem Sizing whith two levels. This paper presents a methodology to solve a problem of supply chain of type OWMR (One whare-house Multi Retailer) whith direct delivery. We assume that the demand is deterministic. The work presented is about a lot sizing problem. The objective is to optimize the total cost of the supply chain, consisting of a production cost, storage cost and transport cost. First the problem was solved by an exact method, which has shown its limits if the number of clients and periods increase. Also we proposed the use of genetic algorithms as a heuristic to solve this problem. The results are satisfactory.

Key words — Lot Sizing, Supply chain, Genetic Algorithm..

1. Introduction

Strong competition in the market requires a focus on the supply chain. Its computerization for optimization of the system has been developed with the advent of new technologies. Optimizing the supply chain is therefore aimed at minimizing the delays and costs incurred between the supplier and the customer. An optimized supply chain reduces number of operations, costs and improves productivity while ensuring optimum quality of service for the end customer.

Our study is in the field of operational research. In this paper we will solve a lot sizing problem with transport, our objective is to optimize the total cost of the chain consisting of a cost of production, storage cost and transport cost.

Lot sizing problems differ and multiply according to assumptions and constraints. Our study is done on a problem of type a production unit, a central repository and multi-retailers with finite capacity named one warehouse multi retailers (OWMR).

[1] Demonstrate that lot sizing problems are NP hard . in [2] authors have classified this problem to many types. According to [3] formulated the problem with a general structure whose storage cost is stationary and fixed for customers, the solution is an algorithm of 1.8 times the optimal value of the objective function. [4] author has studied several structures under different hypothesies and constraints, proposed solutions for each one of them. We are only interested in models similar to ours with transport parameters. The author proposes dynamic programming algorithms.

[5] In his doctoral thesis, studied a model that includes simple production plants, a client set with varying demands over time, on a finite planning horizon and a heterogeneous vehicle fleet. The demand can be met either from the stock held to the customer or from the daily production. He proposed a solution using taboo search to solve the whole problem. [6] The authors dealt with a supply chain consisting of multi-vendor and mono repository and multi-retailer with backlogging, transport using a homogeneous fleet of vehicles with finite capacity. They proposed a solution with genetic algorithms.

2. Problème Description

We assume a structure consisting of a production unit (PU) which produces a single type of product with a constant and finite production capacity in all periods (t) of the planning horizon. This production unit is characterized by a setup cost of production and a unit cost of production which are fixed because it produces only one type. The product is shipped to a central depot (CD), which itself has a setup cost and a storage unit cost.

On the other hand, there are (ni) customers who have stock of limited capacity with a setup cost and a storage unit cost. These clients have deterministic and variable demands on the time horizon. The transport is done without time constraints, a fleet of heterogeneous vehicles is available, each with a finite capacity, a setup cost and a unit transport cost. Following this description, the structure considered is a single-plant mono-deposit (FIG. 1).

We have also defined a set of assumptions. Commencing with the quantity to be produced xt which must not exceed the production capacity in this period. In each period we have (di) demands from (ni) clients that can happen at the same time. The number of vehicles is far greater than the number of customers nv >> ni. The DC has unlimited capacity. We deliver only if there is sufficient space in its stock. Finally, each vehicle must visit only one customer in one period. Initially it is assumed that customer inventory and CD are empty.

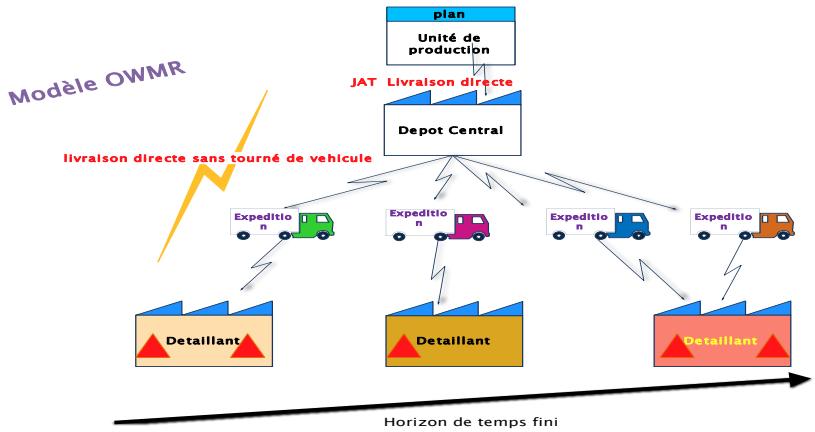


Fig.1 chart of Mono factory mono depot and multi-client and stock.

3. Mathematical Formulation

3.1 Notations.

In this part, we present notations used in mathematical modelling in linear integer programming (LEP).

Production unit :

Production costs are constant.

P : Production capacity of the production unit.

Pf : Setup cost of production.

Pu : Unitary production cost of the production unit.

Central depot :

Similarly for parameters of the CD, the costs do not depend of time.

In addition, the capacity of CD is assumed to be unlimited.

Df : Fixed holding cost .

Du : unitary holding cost .

Transport

Tu_v : Unit cost of transport for one unit of product transported by vehicle v

W_v : The capacity of the vehicle v.

Client storage :

Sf_i : fixed holing cost for client i

Su_i : unitary holding cost for client i.

Cap_i : holding capacity for client i.

Indices :

In the mathematical model we manipulate the variables according to three indices the first (t) to indicate time , the second (i) indicates customers and the third (v) to indicate vehicles.

Sets :

The indices in the previous paragraph (t, i, v) Take their values from the three sets nt, ni, nv respectively.

nt : All periods of the planning horizon

ni : Ensemble des clients

nv : All vehicles.

Decision variables:

Our model has eight decision variables, which are:

$\mathbf{x_t}$: The quantity to be produced in the period.

y : Binary variable which means whether there is production at period t or not

$\mathrm{II_t}$: The quantity stored on the CD at the beginning of the period t.

$\mathbf{z_t}$: Binary variable representing the state (empty or not) of the CD at the period t.

$\mathrm{III_{it}}$: The quantity stored in the customer's stock i at the beginning of the period t.

$\mathrm{zz_{it}}$: Binary variable representing the state (empty or not) of the stock of customer i at period t.

$\mathrm{qt_{vit}}$: The quantity transported to the customer's stock i by the vehicle v at period t.

vv_{tv} : Binary variable representing the state of the vehicle v (occupied or not) at period t.

ql_{it} : The quantity delivered to customer i at period t (equal to the sum of quantities transported in the same period).

3.2. Mathématical Modèle in LEP

Before giving the overall formulation, the three formulas were independently given. That of the cost of production, then the formula of the cost of storage and finally the cost of transport. After we assembled the whole in a global formula.

The formulation below contains the constraints. It is based on balancing stocks and respecting capacities.

$\text{Min} \sum_{t=1}^{nt}(((Pf\ .y_t + Pu\ .x_t) + \sum_{i=1}^{ni} Sf_i.ZZ_{it}\ Su_i\ .III_{it})) + (\sum_{v=1}^{NV}(Tf_v.VV_{vt}\ + \sum_{i=1}^{ni} Tu_t\ .qt_{vit})))$
…….. (1)

y_t: $\begin{cases} 1 : x_t>0 \\ 0 : \text{else} \end{cases}$ ……(2)

z_t: $\begin{cases} 1 : II_t>0 \\ 0 : \text{else} \end{cases}$ ……(3)

zz_t: $\begin{cases} 1 : III_{it}>0 \\ 0 : \text{else} \end{cases}$ ……(4)

vv: $\begin{cases} 1 : qt_{vit}>0 \\ 0 : \text{else} \end{cases}$ ……(5)

$III_{it} \leq Cap_i$ ……..(6)

$x_t \leq P_t$ ……(7)

$qt_{vit} \leq W_v$ ……..(8)

$\sum_{v=1}^{nv} qt_{vti} = ql_{it}$ ……(9)

$\forall\ v, i, t \geq 0$ ……..(10)

The objective function (1) minimizes the total cost of production, transport and storage. The constraints (2), (3), (4) and (5) define the domains of the decision variables. They impose that yt, zt, zzit and vvit are binary variables. The factory keeps its inventory balanced with the demand that is explained by constraint (6)

represents the stock balance of the central deposit, the stock at a given period t equals the stock up to t-1 added to the quantity produced at that time minus the cumulative quantity to be delivered to all Customers in the same period. For Constraint (7), the inventory balance of customers in period t is equal to the inventory up to t-1 added to the quantity to be delivered to the customer in period t minus the demand for the same period. The constraints (8), (9) and (10) represent the capacity constraints, the constraint (8) checks the stock level of the customer i at the period t which must not exceed the storage capacity of This customer stock. In constraint (9), it is found that the quantity to be produced must not exceed the production capacity. In (10) the quantity to be transported does not exceed the capacity of the vehicle. (10) shows that the quantity delivered to the customer i is equal to the sum of the quantities transported by the vehicles v.

Once our objective function was defined, we opted for an exact method using the CPLEX solver of the GAMS language.

4. solving problem with GAMS-CPLEX

We used General Algebraic Modeling System (GAMS) to solve the mathematical model of the problem with the CLPEX solver (referenced to the C language and the SIMPLEX algorithm),

4.1. Instances Description

We propose the generation of instances by the combination of the following three parameters:

• (v, i, t): Number of vehicles v, from customer i to period t.

• P: Production capacity.

• C = (PU, SU (i), TU (v)): The unit costs of production, customer storage and delivery

For each parameter we take 3 values, which gives us 3*3*3 = 27 possibilities (Tables 1 and 2). Each instance will be generated 10 times to calculate the average execution time. However, the remaining parameters of the model must remain invariant. d(i,t)= U(5,25).

PF= 10.

DU=0.5.

DF=1.5 .

W(v)=0.1*U(1,10)*dmax. ; such as dmax is maximum of all demands .

b=dmax*ni.

TF(v)=0.1*U(1,10).

CAP(i)=U(2,3)*dmaxcli(i) ; such as dmaxcli(i) maximum of client I demands.

SF(i)= U(1,5).

U (a,b): The discrete uniform law on the interval [a,b].

	value1	Value2	Value3

(V, I, T)	(10,20,10)	(20,40,20)	(40,60,30)
P	$1 * b$	$2 * b$	$3 * b$
Cost	Coût1	Coût2	Coût3

Table 1. Parameters configuration (V, I, T, P, Pu, Tu, Su).

	Pu	Tu(v)	Su(i)
Cost1	$10 * U(1,5)$	$1 * U(2,5)$	$1 * U(1,5)$
Cost2	$1 * U(1,5)$	$10 * U(2,5)$	$2 * U(1,5)$
Cot3	$1 * U(1,5)$	$1 * U(2,5)$	$10 * U(1,5)$

Table.2. different costs.

- Cost1: the cost of production is high.

- Cost2: the cost of delivery is high.

- Cost3: the cost of storage is high.

4.2. Résuls of exacte solution

Table 3 presents a simulation of the system under study with randomly generated parameters. All the possibilities of (Table 2.2) are formed and then the cost and the calculation time.

Iteration	(nv, ni, nt)	P	Costs			Average run time (s)
			Pu	Tu(v)	Su(i)	
1	(10,20,10)	1xb	10xU(1,5)	1xU(2,5)	1xU(1,5)	0.169
2	(10,20,10)	1xb	1x U(1,5)	10xU(2,5)	2xU(1,5)	0.189
3	(10,20,10)	1xb	1x U(1,5)	1xU(2,5)	10xU(1,5)	0.176
4	(10,20,10)	2xb	10xU(1,5)	1xU(2,5)	1xU(1,5)	0.189
5	(10,20,10)	2xb	1xU(1,5)	10xU(2,5)	2xU(1,5)	0.183
6	(10,20,10)	2xb	1x U(1,5)	1xU(2,5)	10xU(1,5)	0.198
7	(10,20,10)	3xb	10xU(1,5)	1xU(2,5)	1xU(1,5)	0.186
8	(10,20,10)	3xb	1x U(1,5)	10xU(2,5)	2xU(1,5)	0.215
9	(10,20,10)	3xb	1* U(1,5)	1xU(2,5)	10xU(1,5)	0.192
10	(20,40,20)	1xb	10xU(1,5)	1xU(2,5)	1xU(1,5)	3.353
11	(20,40,20)	1xb	1xU(1,5)	10xU(2,5)	2xU(1,5)	3.640
12	(20,40,20)	1xb	1x U(1,5)	1xU(2,5)	10xU(1,5)	3.073
13	(20,40,20)	2xb	10xU(1,5)	1xU(2,5)	1xU(1,5)	4.184
14	(20,40,20)	2*b	1xU(1,5)	10xU(2,5)	2xU(1,5)	5.513

15	(20,40,20)	2xb	1x U(1,5)	1xU(2,5)	10xU(1,5)	4.527
16	(20,40,20)	3xb	10xU(1,5)	1xU(2,5)	1xU(1,5)	4.649
17	(20,40,20)	3xb	1xU(1,5)	10xU(2,5)	2xU(1,5)	4.370
18	(20,40,20)	3xb	1x U(1,5)	1xU(2,5)	10xU(1,5)	5.167
19	(40,60,30)	1xb	10xU(1,5)	1xU(2,5)	1xU(1,5)	18.782
20	(40,60,30)	1xb	1xU(1,5)	10xU(2,5)	2xU(1,5)	18.843
21	(40,60,30)	1xb	1xU(1,5)	1xU(2,5)	10xU(1,5)	18.647
22	(40,60,30)	2xb	10xU(1,5)	1xU(2,5)	1xU(1,5)	19.191
23	(40,60,30)	2xb	1xU(1,5)	10xU(2,5)	2xU(1,5)	19.681
24	(40,60,30)	2xb	1xU(1,5)	1xU(2,5)	10xU(1,5)	20.230
25	(40,60,30)	3xb	10xU(1,5)	1xU(2,5)	1xU(1,5)	22.939
26	(40,60,30)	3xb	1xU(1,5)	10xU(2,5)	2xU(1,5)	25.906
27	(40,60,30)	3xb	1xU(1,5)	1xU(2,5)	10xU(1,5)	30.802
28	(50,70,40)	1xb	10xU(1,5)	1xU(2,5)	1xU(1,5)	50.874
29	(60,80,50)	1xb	10xU(1,5)	1xU(2,5)	1xU(1,5)	104.450
30	(100,120,70)	1xb	10*U(1,5)	1xU(2,5)	1xU(1,5)	650.590

Table 3. instances and the results of the CPLEX solver.

4.2. résults Interprétation

Figure 2 shows the mean execution time as a function of (nv, ni, nt), we notice that the time increases with the increase of (nv, ni, nt), this means that the CPLEX solver consumes more time To find the solution to the problem. The curve is very uniform and increasing, indicating that these three values have a great influence on the computation time. With large iterations, the calculation becomes very slow to see impossible to run the program on an ordinary machine.

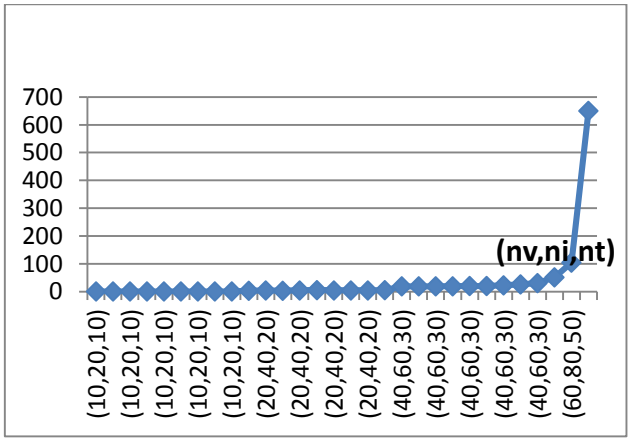


Fig .2. chart of processing time according to number of vehicle, customer and period.

We were able to do the mathematical modeling of the batch problem sizing with transport. We used the CPLEX solver to solve this problem, we obtained satisfactory results but with a certain limit of number of variables (Table 3). But in the case where the number of variables increases it will be necessary to seek a heuristic which saves more time. In the following we will see how to solve this problem using genetic algorithms.

5. Solving Problem with Genetic Algorithm

In 1859 the foundations of evolution were laid by C. Darwin with his idea of natural selection ("in every species the best are selected"). In 1901 they were the basis of genetics that were laid down by De-Vries following his theory of mutationism. In 1975 Jhon Holland proposed the Genetic Algorithm. In 1989 Goldberg exposed the mathematical foundations of genetic algorithms. [7]

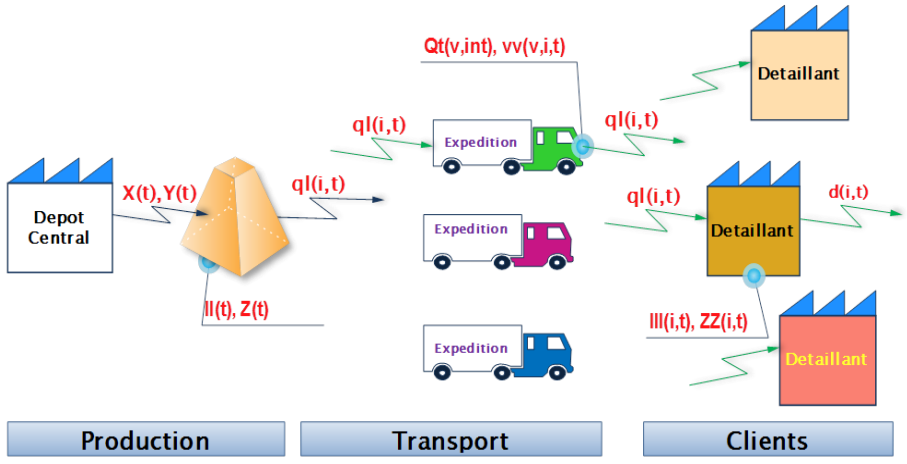
The application of GAs to solve the sizing problem considered in the model as a whole is very difficult, so we opted to decompose the model into three sub-structures (Figure 3).

• Production: which presents the production unit and the stock of the central deposit with the decision variables x (t), y (t), II (t), z (t).

• Transport: which presents the vehicles with the decision variables qt (v, i, t) vv (v, t).

• The client: which is presented by the client stock with the decision variables ql (i, t) III (i, t) zz (i, t).

Thus, our objective function consists of 8 variables of decisions which are x (t), y (t), z (t), II (t), qt (v, i, t), vv , III (i, t) and zz (i, t). We note that the variable ql (i, t) is a variable that is not part of the objective function but that was used as an intermediate variable to calculate other variables (see algorithm 5). The application of the genetic algorithm on the model in its raw state is difficult if not impossible. We will use the binary coding and we will deduce the rest of the decision variables with the repair algorithms. The procedure to follow is in the



flowchart of figure (4).

Fig.3. 3 sub structures of supply chain studied.

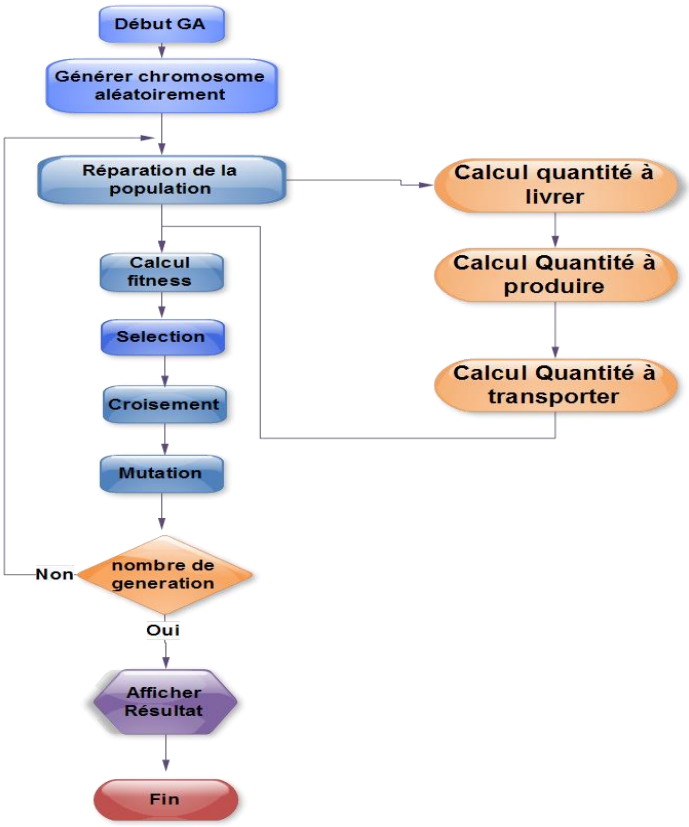


Fig.3. flow char of genetic algorithm

6.1. Chromosome Coding:

In the framework of our application, we have chosen a binary coding which limits the decision variables to two values. Thus, the multi chromosome with the following structure:

$$\begin{bmatrix} [y(1) & y(2) & \dots\dots\dots\dots\dots y(nt)] \\ \begin{bmatrix} zzz(1,1) & zzz(1,2) & \dots\dots\dots\dots\dots zzz(1.nt) \\ zzz(2,1) & zzz(2,2) & \dots\dots\dots \quad \dots zzz(2,nt) \\ \\ zzz(ni,1) & zzz(ni,2) & \qquad\qquad zzz(ni,nt) \end{bmatrix} \end{bmatrix}$$

Fig.4. multi chromosome structure.

The multi-chromosome is a binary matrix of ni + 1 rows and nt columns forming two parts, the first part occupies the first row present by the variable y (t) signifying whether there is pro-duction or not. The second part is a matrix of n rows and nt columns represented by the matrix zzz (i, t) (FIG. 4), indicating whether there is delivery to customer i at period t or not. Since the multi chromosome is binary and we have the integer and real decision variables, we applied the zero-switch principle to deduce the rest of the decision variables from the multi chromosomes. These variables are x (t), and ql (i, t).

6.2. zero Switch Principle

The method of the "zero switch "[8] and [7] gives an optimal solution to a lot sizing problem without capacity.

The principle is as follows:

X (t): quantity produced at time t.

Y (t): binary variable 0/1 which indicates whether there is production or not.

I (t): the stock level at time t.

$$x(t).I(t) = 0 \quad \rightarrow \quad \begin{cases} x(t) = 0 \; and \; I(t) = 0 \\ or \\ x(t) = 0 \; and \; I(t) \neq 0 \\ or \\ x(t) \neq 0 \; et \; I(t) = 0 \end{cases}$$

This means that ensuring the zero stock or zero quantity produced. If y is zero then the quantity to be produced is also zero. But if this is not the case, it is necessary to produce the quantity demanded in this period and the following periods (the sum of the demands for y (t) = 0 to y (t ') = 1 knowing that t'> t) 'To make it zero again..

5.1. vehicles Assignment

After calculating the quantities to be produced and delivered, the calculation of the quantity to be carried for each vehicle remains.

This problem is close to the generalized backpack problem (Muliknapsak) where the bag is the quantity to be delivered and the objects are the vehicles. Several heuristics have been used to solve such an NP-difficult problem, such as ant colonies, genetic algorithms, glutton algorithms, etc.

As part of this work we have opted for 'greedy algorithm.

The principle of greedy algorithms consists in sorting the objects (which are in our case the vehicles) in descending order of the weighted capacity Cp (v) which is equal to the sum of the fixed cost TF (v), and unit cost Tu) Weighted by the capacity of the vehicle W (v).$CP(v) = \frac{TF(v)+TU(v)}{w(v)}$

Tel que :

- CP : Weighted capacity.
- TF : Fixed transport cost.
- TU : Unit transport cost.
- W : vehicle capacity.
- v: véhicule v.
- ql : quantité à livrer.

5.2. Implémentation and numérical résults of intégral génétic algorithme

In order to implement this algorithm, we used the MATLAB language version 9 (2011), on a 2.3 GH processor and 4 GB RAM. The parameters are generated randomly, the results of a Simulation will be presented in the following.

Our approach converges towards the optimum, this can be seen in the y-curve of the cost which regresses as a function of the generation number (fig. 5).

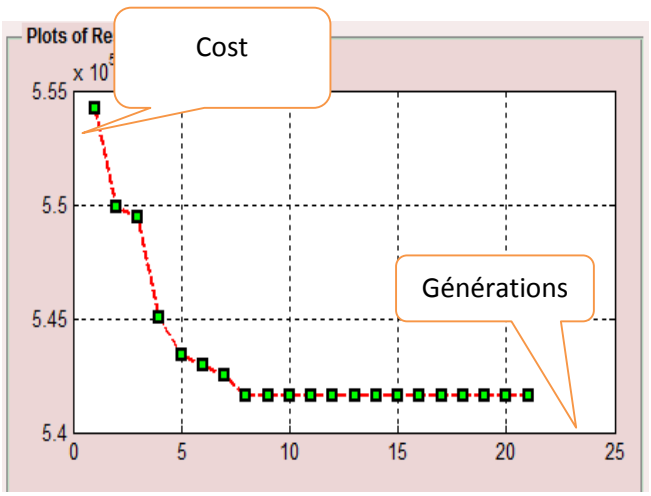


Fig.5. plot of cost according to generations.

6. Comparison between GA(meta- heuristic) and GAMS (exact Method)

Our research is a simulation that allows us to make the production plan of a system similar to ours. In order to verify the degree of validity of the evolutionary algorithm it was necessary to tilt the same parameters towards GAMS in order to have the same input data for the two solutions. The same parameters were used for the exact method.

Using the parameters of the genetic algorithm presented in Table 4, the simulation was performed.

GA parameters	values
Population	200
Generations	20
Mutation probability	0.05
Crossover probability	0.7
Number of mutation	10

Table.4. genetic algorithm properties.

Iter	i	t	MATLAB		GAMS		Error %
			Cost	Tmp	Cost	Tmp	
1	2	3	36 787.00	2 .20	35 125.00	0.25	4.73
2	4	4	475 321.62	6.50	464 892.00	0.26	2.24
3	8	5	924 298.64	6.81	890 483.00	0.26	3.80
4	10	6	756 863.03	7.12	726 382.00	0.20	4.20
5	12	7	1 809 077.23	7.60	1 741 985.00	0.30	3.85
6	14	8	3 015 339.59	9.09	2 889 644.00	0.50	4.35

7	16	9	3 874 523.31	11.00	3 795 595.93	0.70	2.08
8	18	10	2 332 398.57	15.32	2 296 532.71	0.89	1.56
9	20	11	3 885 962.24	16.65	3 803 995.41	1.23	2.15
10	22	12	7 526 902.34	18.21	7 436 158.91	2.10	1.22
11	24	13	5 488 154.36	19.83	5 313 347.75	3.10	3.29
12	26	14	7 922 471.64	25.21	7 896 458.78	3.00	0.33
13	28	15	9 302 135.66	30.10	9 141 618.41	4.05	1.76
14	30	16	8 489 162.32	34.36	8 142 114.32	4.50	4.26
15	32	17	9 578 227.58	36.00	9 127 856.37	5.97	4.93
16	34	18	1 737 943.52	38.25	1 709 840.37	9.28	1.64
17	36	19	1 557 960.00	40.58	1 530 737.36	11.00	1.78
18	38	20	1 722 353.21	55.28	1 668 987.96	13.04	3.20
19	40	21	1 882 930.00	60.52	1 817 528.00	15.38	3.60
20	50	50	6 876 659.12	80.58	6 759 755.02	94.00	1.73
21	60	60	6 525 364.14	100.00	6 230 809.52	124.00	4.73
22	70	70	113460100.25	480.00	108 284 790.68	870.00	4.88

Table.5. Comparison between the heuristic method and the exact method.

In Table 5 we find the simulation and the comparison between our approach and the exact method.

1. Interpretation of Results

In order to interpret the results, we have based on two axes: the first is the computation time, and the second is the margin of error between the two approaches (Fig. 6.7). We have seen that:

The error between our approach and the exact method is in the interval [0%, 5%]. If we calculate the average error we find: 3.01%.

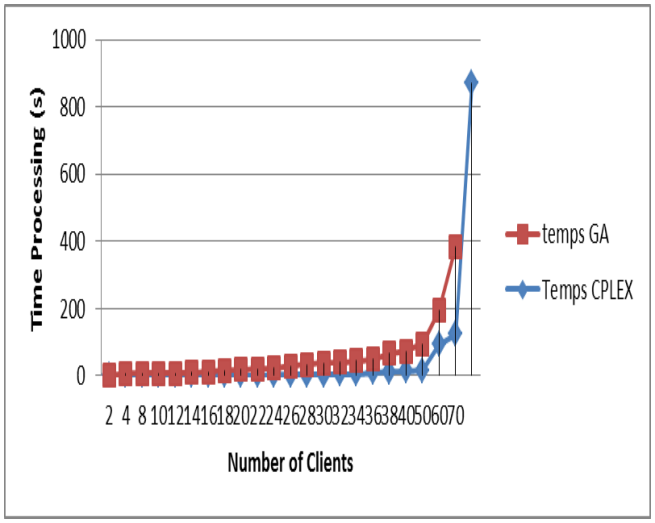


Fig6. Plot representing processing time of two methods according to the number of clients.

The computation time is small for the exact and high method for the approximate method, up to the 20th iteration (ni, nt) = (60,60). Beyond this we find the inverse (the calculation time of the ex-act method becomes more important than that of the genetic algorithm). During the simulation we observe the slowness or the blocking of the execution of the exact method (need of more memories (too many intermediate variables to solve the problem).

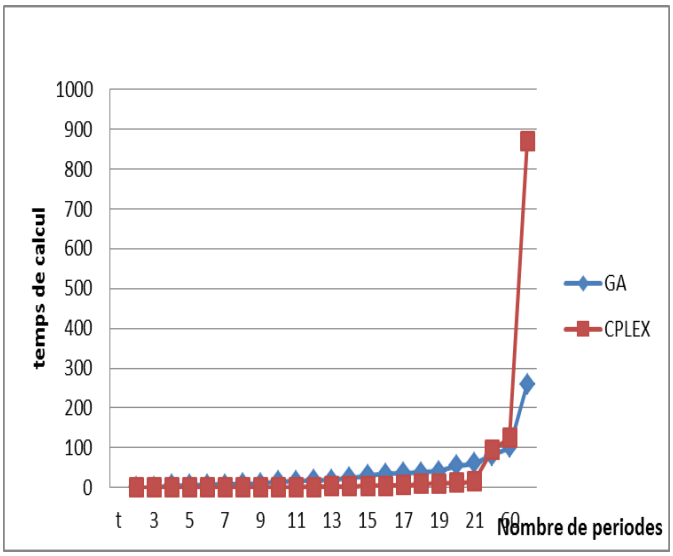


Fig6. Graphe représentant le temps de calcul des deux méthodes en fonction du nombre de périodes

According to the graphs in Fig. 6.7, the calculation time of the exact method increases by more than 200% with respect to the approximate method, which makes our approach valid and applicable in firms of such a structure (OWMR).

7. Conclusion And Perspective

The work presented in this paper concerns the development of an approach using genetic algorithms to solve a sizing batch problem with transport with finite capacity.

Our study consists of two parts. The first part describes the particular structure of the logistics chain, which consists of a production unit that deposits its product in a central repository with infinite capacity. This product is then shipped to geographically dispersed customers using a heterogeneous vehicle fleet and on the other hand its mathematical modeling and the use of an exact solution using the CPLEX solver under GAMS. With the exact method, it can be seen that the calculation time becomes too great if the number of clients and periods increases.

In the second part, a binary-coded genetic algorithm was presented as a heuristic to the solution of this problem. Given the complexity of the structure used, modular decomposition and use of repair algorithms was necessary to facilitate the use of genetic algorithms.

We simulated GA on a population of 200 individuals and 20 generations with a wheel roulette selection, a one-point cross with a probability of 0.7 and a multipoint mutation (10 mutation points) with a proba Of 0.05. The parameters of the objective function are randomly generated which makes our approach applicable to structures of this kind. The results obtained have shown that the calculation time depends on the number of customers, periods and vehicles. In addition, we compared our approach with the exact method. It has been observed that at the first iterations the computation time is considerably large for the GA. Once (nv, ni, nt) =

(120,60,60) the processing for CPELX becomes time consuming and memory-intensive, compared to the AG which gave acceptable results arriving at an error of 0.3% and A very short execution time compared to that of the exact method.

Our approach can be improved by adding other heuristics to reduce the error and approach the optimum as the local search. In addition, the model can be enhanced by adding assumptions such as central deposit capacity, backlogging, time window, producing multiple products instead of a single product, varying setup costs and unit production costs, Storage and transport, in each period. In addition, use a homogeneous vehicle fleet and delivery with vehicle turnover instead of a direct delivery, and integrate the VRP problem with the sizing lot problem. Also combining with environmental parameters and human resources.

References

E.Arkin, D.Joneja, R.Roundy, “Computational complexity of uncapacitated multi-echelon production planning problems”. Operations Research Letters, 1989.

Oğuz Solyalı, Haldun Süral, “The one-warehouse multi-retailer problem: reformulation, classification, and computational results”, Annals of Operations Research Springer Science Business Media, LLC 2011.

Retsef Levi, Robin Roundy, David Shmoys, Maxim Sviridenko, “A Constant Approximation Algorithm for the One-Warehouse Multi retailer Problem, Management “, Science April 2008 vol. 54 no. 4 763-776.

Ayse Akbalik , « Optimisation de la gestion intégrée des flux physiques dans une chaine logistique ; extension du problème de dimensionnement de lot » , thèse de doctorat. institut national polytechnique de Grenoble 2006.

Narameth Nananukul, “Lot-sizing and inventory routing for a production-distribution supply chain”, These de doctorat. the university of texas at austin may 2008.

W. Yang , Felix T.S. Chan,V. Kumar, “Optimizing replenishment polices using Genetic Algorithm for single-warehouse multi-retailer system, Expert systems with Applications”, An International Journal Volume 39 Issue 3, February, 2012.

Harvey M.Wagner, Thomson M. Within, “Dynamic version of the Economic Lot size Model”, Management Science, 1958.

T. Baeck, D. B. Fogel, Z. Michalewicz, “Evolutionary computation”», Institute of Physics Publishing, 2000.