International Journal of Mechanical, Industrial and Aerospace Sciences
ISSN: 2517-9950
Vol:7, No:2, 2013

Applying Genetic Algorithms for Inventory Lot-Sizing Problem with Supplier Selection under Storage Space

Vichai Rungreunganaun and Chirawat Woarawichai

Abstract**—**The objective of this research is to calculate the optimal inventory lot-sizing for each supplier and minimize the total inventory cost which includes joint purchase cost of the products, transaction cost for the suppliers, and holding cost for remaining inventory. Genetic algorithms (GAs) are applied to the multi-product and multi-period inventory lot-sizing problems with supplier selection under storage space. Also a maximum storage space for the decision maker in each period is considered. The decision maker needs to determine what products to order in what quantities with which suppliers in which periods. It is assumed that demand of multiple products is known over a planning horizon. The problem is formulated as a mixed integer programming and is solved with the GAs. The detailed computation results are presented.

Keywords**—**Genetic Algorithms, Inventory lot-sizing, Supplier selection, Storage space.

I. Introduction

LOT- sizing problems are production planning problems with the objective of determining the periods where production should take place and the quantities to be produced in order to satisfy demand while minimizing production and inventory costs [1]. Since lot-sizing decisions are critical to the efficiency of production and inventory systems, it is very important to determine the right lot-sizes in order to minimize the overall cost.

Lot-sizing problems have attracted the attention of researchers. The multi-period inventory lot-sizing scenario with a single product was introduced by Wagner and Whitin [2], where a dynamic programming solution algorithm was proposed to obtain feasible solutions to the problem. Soon afterwards, Basnet and Leung [3] developed the multi-period inventory lot-sizing scenario which involves multiple products and multiple suppliers. The model used in these former research works is formed by a single-level unconstrained resources indicating the type, amount, suppliers and purchasing time of the product.

Vichai Rungreunganaun is with the Department of Industrial Engineering, Faculty of Engineering, KMUTNB, Bangkok, Thailand 10800 (phone: 02913-2500; fax: 02912-2012; e-mail: r_vichai@yahoo.com).

Chirawat Woarawichai is with the Department of Industrial Engineering, Faculty of Engineering, Rajamangala University of Technology Lanna Tak, Thailand 63000 (phone: 055515-900; fax: 055515-900; e-mail: Chirawat-w@hotmail.com).

This model is not able to consider the capacity limitations. One of the important modifications we consider in this paper is that of introducing storage capacity. In this paper based on Basnet and Leung [3] genetic algorithms (GAs) are applied to the multi-product and multi-period inventory lot-sizing problem with supplier selection under storage space. Also a maximum storage space for the decision maker in each period is considered. The decision maker needs to determine what products to order in what quantities with which suppliers in which periods. The objective of this research is to calculate the optimal inventory lot-sizing for each supplier and minimize the total inventory cost.

II. Methodology

A. Genetic Algorithms Approach

The genetic algorithms (GAs) approach is developed to find optimal (or near – optimal) solution. Detailed discussion on GAs can be found in books by Holland [4], Michalewicz [5], Gen and Cheng [6] [7], Davis [8] and, Goldberg [9]. In this section, we explain GAs procedure is illustrated in Fig. 1. Topics covered include (1) Chromosome structure (2) Initial population (3) Evaluation (4) Selection (5) Crossover (6) Mutation, and (7) Termination rule.

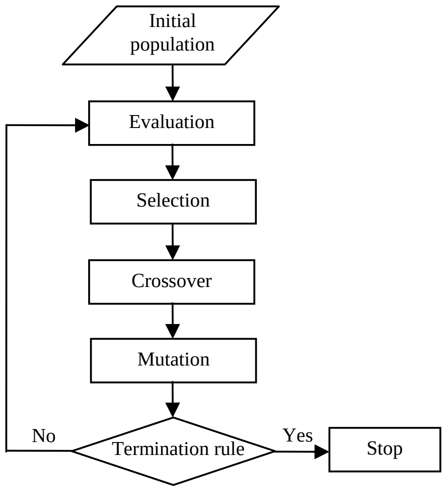


Fig. 1 The genetic algorithm procedure

1. Chromosome Structure

In this problem, we take each chromosome as a model solution, where *I, J* and *T* are the number of products, suppliers and periods, respectively, and each chromosome is a real values vector (we make it by *X*) by length of (*I* x *J* x *T*) and a binary values vector are 0 or 1 (we make it by *Y*) by length of (*J* x *T*), appropriate by each X_{ijt} and Y_{jt} (decision variables). For example, the representation of a chromosome is illustrated in Fig. 2.

Chromosome	X_{111}	X_{112}	$\dots$	X_{ijt}	$\dots$	X_{IJT-1}	X_{IJT}
	Y_{11}	Y_{12}	$\dots$	Y_{jt}	$\dots$	Y_{JT-1}	Y_{JT}

Fig. 2 Chromosome structure

2. Initial Population

The population initialization technique used in the GAs approach is a randomly generate solutions for the entire population. Population size depends only on the nature of problems and it must balance between time complexity and search space measure. More population size may increase the probability of finding optimal solution, but may induce a longer computer time. In this paper, we use a population size is set not less than twice the length of the vector of the chromosomes [10].

3. Evaluation or Fitness Function

It is evaluated by the chromosome structure which results in positive value in [11]. Fitness value defines the relative strength of a chromosome compared with the others, and the optimality of the solution to the problem. The fitness function of this model is an objective one (to minimize cost).

4. Selection

The selection of parents to produce successive generations plays an extremely important role in the GAs. The goal is to allow the fittest individuals to be selected more often to reproduce. However, all individuals in the population have a chance of being selected to reproduce the next generation. In this paper, the roulette wheel selection technique is used [12].

5. Crossover Operator

Crossover operators combine information from two parents in such a way that the two children (solutions for the next population) resemblance to each parent. There are several available methods to do so [13]. This paper adapts two point crossover operators to solve GAs [12].

6. Mutation Operator

Mutation operators alter or mutate one chromosome by changing one or more variables in some way or by some random amount to form one offspring. For mutation, we use a linear mutation by probability (1/*I* x *J* x *T*) for mutating *X* vector and bit-wise mutation by probability (1/*J* x *T*) for *Y* vector [14], [15].

7. Termination Rule

The GAs moves from generation to generation selecting and reproducing parents until a termination criterion is met. The most frequently used stopping criterion is a specified maximum number of generations. In this paper, there are two stop criteria. First, the process is stopped when the number of interations has reached the maximum generations. Second, the process is stopped when the maximum time exceeds (set at 120 minutes) [3].

B. Formulation

We also make the following assumptions and mathematical for the model:

<u>*Assumptions*</u>

- Demand of products in period is known over a planning horizon.
- All requirements must be fulfilled in the period in which they occur: shortage or backordering is not allowed.
- Transaction cost is supplier dependent, but does not depend on the variety and quantity of products involved.
- Holding cost of product per period is product-dependent.
- Product needs a storage space and available total storage space is limited. The storage space is for finished goods.

Base on the above assumption of model, Fig. 3 shows the behavior of the model considering the scenario of multi-period inventory lot-sizing problem with supplier selection under storage space. The characteristics of the model used to determine what products i, with which suppliers j, and in which periods t to order (X_{ijt}).

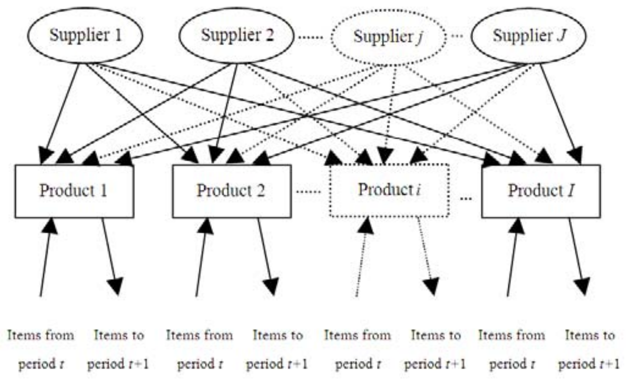


Fig. 3 Behavior of the model in period

<u>*Mathematical Modeling*</u>

This paper is built upon Basnet and Leung [3] model. We formulate the multi-product and multi-period inventory lot-sizing problem with supplier selection under storage space using the following notation:

Indices:

i	=	$1,\dots,I$	index of products
j	=	$1,\dots,J$	index of suppliers
t	=	$1,\dots,T$	index of time periods

Parameters:

D_{it} = demand of product i in period t
P_{ij} = purchase price of product i from supplier j
H_i = holding cost of product i per period
O_j = transaction cost for supplier j
w_i = storage space product i
S = total storage space

Decision Variables:

X_{ijt} = number of product i ordered from supplier j in period t
Y_{jt} = 1 if an order is placed on supplier j in time period t, 0 otherwise

Intermediate Variable:

R_{it} = Inventory of product i, carried over from period t to period $t+1$

Regarding the above notation, the mixed integer programming is formulated as follows:

$$\text{Minimize (TC)} \sum_i \sum_j \sum_t P_{ij} X_{ijt} + \sum_j \sum_t O_j Y_{jt} + \sum_i \sum_t H_i \left(\sum_{k=1}^{t} \sum_j X_{ijk} - \sum_{k=1}^{t} D_{ik} \right) \quad (1)$$

Subject to:

$$R_{it} = \sum_{k=1}^{t} \sum_j X_{ijk} - \sum_{k=1}^{t} D_{ik} \geq 0 \quad \forall i, t \quad (2)$$

$$\left(\sum_{k=t}^{T} D_{ik} \right) Y_{jt} - X_{ijt} \geq 0 \quad \forall i, j, t \quad (3)$$

$$\sum_i w_i \left(\sum_{k=1}^{t} \sum_j X_{ijk} - \sum_{k=1}^{t} D_{ik} \right) \leq S \quad \forall t \quad (4)$$

$$Y_{jt} = 0 \text{ or } 1 \quad \forall j, t \quad (5)$$

$$X_{ijt} \geq 0 \quad \forall i, j, t \quad (6)$$

The objective function as shown in (1) consists of three parts: the total cost (TC) of 1) purchase cost of the products, 2) transaction cost for the suppliers, and 3) holding cost for remaining inventory in each period in $t+1$.

Constraint in (2) all requirements must be filled in the period in which they occur: shortage or backordering is not allowed. Constraint in (3) there is not an order without charging an appropriate transaction cost. Constraint in (4) each products have limited capacity. Constraint in (5) is binary variable 0 or 1 and Constraint in (6) is non-negativity restrictions on the decision variable. According to a large optimal problem, a GAs approach is applied to solve this problem.

C. A Numerical Example

In this section we solved a numerical example of the model using real parameter genetic algorithms. We consider a scenario with three products over a planning horizon of five periods whose requirements are as follows: demands of three products over a planning horizon of five periods are given in Table I. There are three suppliers and their prices and transaction cost, holding cost and storage space are show in Table II and Table III, respectively.

TABLE I
DEMANDS OF THREE PRODUCTS OVER A PLANNING HORIZON OF FIVE PERIODS (D_{it})

Products	Planning Horizon (Five Periods)				
	1	2	3	4	5
A	12	15	17	20	13
B	20	21	22	23	24
C	20	19	18	17	16

TABLE II
PRICE OF THREE PRODUCT BY EACH OF THREE SUPPLIERS X, Y, Z (P_{ij}) AND TRANSACTION COST OF THEM (O_j).

Products	Price		
	1	2	3
A	30	33	32
B	32	35	30
C	45	43	45
Transaction Cost	110	80	102

TABLE III
HOLDING COST OF THREE PRODUCTS A, B, C (H_i) AND STORAGE SPACE OF THEM (w_i)

Products	Products		
	A	B	C
Holding Cost	1	2	3
Storage Space	10	40	50

The total storage space (S) is equal to 200.

The results of applying the proposed method are shown in Table IV. The solution of this problem (I = 3, J = 3, and T = 5) is to place the following orders. All other $X_{ijt} = 0$:

TABLE IV
ORDER OF THREE PRODUCTS OVER A PLANNING HORIZON OF FIVE PERIODS (X_{ijt})

Products	Planning Horizon (Five Periods)				
	1	2	3	4	5
A	X_{131} = 12	X_{132} = 15	X_{113} = 37	-	X_{135} = 13
B	X_{231} = 20	X_{231} = 21	X_{213} = 22	X_{234} = 23	X_{235} = 24
C	X_{131} = 12	X_{132} = 15	X_{113} = 37	-	X_{135} = 13

Cost calculation for this solution:
Purchase cost for product 1 from supplier 1, 3
= (37×30) + (12+15+13) × 32 = 2,390.
Purchase cost for product 2 from supplier 1, 3
= (22×32) + (20+21+23+24) × 30 = 3,344.
Purchase cost for product 3 from supplier 1, 3
= (18×45) + (20+19+17+16) × 45 = 4,050.
Transaction cost from supplier 1, 3
= (1×110) + (4×102) = 518.
Holding cost for product 1
$R_{13} = X_{113} - D_{13}$ = 37 − 17 = 20.
$= H_1 \sum R_{1t}$ = 1× (0 + 0 + 20 + 0 + 0) = 20.
Thus, the total cost for this solution
= 2,390 + 3,344 + 4,050 + 518 + 20
= 10,322.

III. RESULTS

Computation Results

In this section the comparison of the two methods solved problem size is using a commercially available optimization package like LINGO12 and GAs code is developed in MATLAB7. Experiments are conducted on a personal computer equipped with an Intel Core 2 duo 2.00 GHz, CPU speeds, and 1 GB of RAM. The transaction costs are generated from *int* [50; 200], a uniform integer distribution including 50 and 200. The prices are from *int* [20; 50], the holding costs from *int* [1; 5], the storage space from *int* [10; 50], and the demands are from *int* [10; 200].

The result in Table V shows the GAs comparing with LINGO12 for the nine problem sizes. A problem size of *I; J; T* indicates number of suppliers = *I*, number of products = *J*, and number of periods = *T*. Computation time limit is set at 120 minutes. For comparison, the percentage error is calculated by (7) and (8).

Percentage error of LINGO12

$$= \left[\frac{\text{Upper bound - Lower bound}}{\text{Upper bound}}\right] \times 100 \qquad (7)$$

Percentage error of GAs

$$= \left[\frac{\text{Upper bound LINGO - GAs}}{\text{Upper bound LINGO}}\right] \times 100 \qquad (8)$$

The solution time of LINGO12 to optimal is a short time as the small problem size (with the problem sizes 3 x 3 x 5; 3 x 3 x 10; 3 x 3 x 15; and 4 x 4 x 10). For large problems sizes LINGO12 cannot obtain optimal solutions within limit time due to as the larger problem size (with the problem sizes 4 x 4 x 15; 5 x 5 x 20; 10 x 10 x 50; 10 x 10 x 80; and 15 x 15 x 50). The GAs can optimally solve when the problem size is small (with the problem sizes 3 x 3 x 5; 3 x 3 x 10; 3 x 3 x 15; 4 x 4 x 10; 4 x 4 x 15; 5 x 5 x 20; and 10 x 10 x 50). There are two problems which GAs cannot obtain optimal solutions (with the problem sizes 10 x 10 x 80; and 15 x 15 x 50).

Next, we study differences in the problem sizes between solutions from the optimization with LINGO12 and the GAs. The results are show in Fig. 4, a plot of the problem size versus solution time. LINGO12 uses longer computation time more than GAs with seven problem sizes, but uses equal time with two problem sizes. As show in Fig. 5 a plot of the problem size versus % error when the problem size is very large, LINGO12 used a maximum % error from the optimal solutions is found to be 4.41% (at the problem size 10 x 10 x

TABLE V
COMPARATIVE RESULTS OF THE TWO METHOD

Problem size	Optimization approach with LINGO12			Genetic Algorithms (GAs)		
	Total cost	Solution time (minute)	% Error	Total cost	Solution time (minute)	% Error
3 x 3 x 5	10,322	0.01	0	10,322	0.02	0
3 x 3 x 10	20,644	0.14	0	20,644	0.21	0
3 x 3 x 15	30,966	14.35	0	30,966	1.45	0
4 x 4 x 10	25,436	6.34	0	25,436	0.51	0
4 x 4 x 15	38,154[a], 37,828[b]	120	0.85	38,154	2.47	0
5 x 5 x 20	60,218[a], 59,527[b]	120	1.14	60,200	3.36	0.03
10 x 10 x 50	285,344[a], 274,758[b]	120	3.70	284,940	108.50	0.14
10 x 10 x 80	456,494[a], 436,317[b]	120	4.41	455,904	120	0.12
15 x 15 x 50	417,800[a], 405,155[b]	120	2.66	416,473	120	0.31

[a]LINGO12 = Upper bound, [b]LINGO12 = Lower bound.

80) which has more % error than GAs. The GAs can solve small % error of two problem sizes (at the problem size 10 x 10 x 80; and 15 x 15 x 50). Fig. 6 and Fig. 7 show compares result between LINGO12 and GAs in problem size 3 x 3 x 5.

Thus, it is evident that GAs is an effective means for solving the problem. GAs solution is optimal when the problem size is small. For larger problems GAs can find feasible solution within time limit for which LINGO12 fails to find the optimum. However, the GAs provides superior solutions to those from LINGO12 that are close to optimum in a very short time, and thus appears quite suitable for realistically sized problems.

Additionally, the computation time when using GAs is also short, making it a very practical means for solving the multiple products and multi-period inventory lot-sizing problem with supplier selection under storage space.

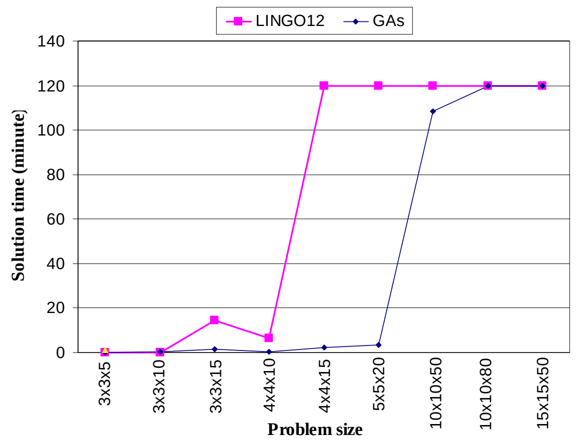


Fig. 4 Plot of the problem size vs. solution time (minute)

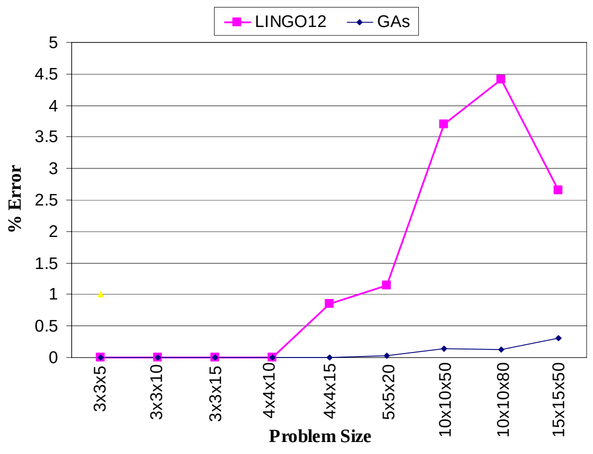
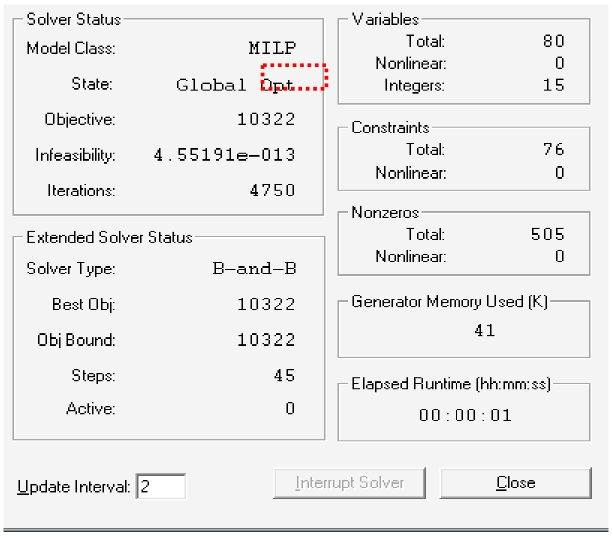


Fig. 5 Plot of the problem size vs. % error



Solver Status	
Model Class:	MILP
State:	Global Opt
Objective:	10322
Infeasibility:	4.55191e-013
Iterations:	4750

Extended Solver Status	
Solver Type:	B-and-B
Best Obj:	10322
Obj Bound:	10322
Steps:	45
Active:	0

Variables	
Total:	80
Nonlinear:	0
Integers:	15

Constraints	
Total:	76
Nonlinear:	0

Nonzeros	
Total:	505
Nonlinear:	0

Generator Memory Used (K): 41

Elapsed Runtime (hh:mm:ss): 00:00:01

Update Interval: 2

Fig. 6 The best objective of LINGO 12

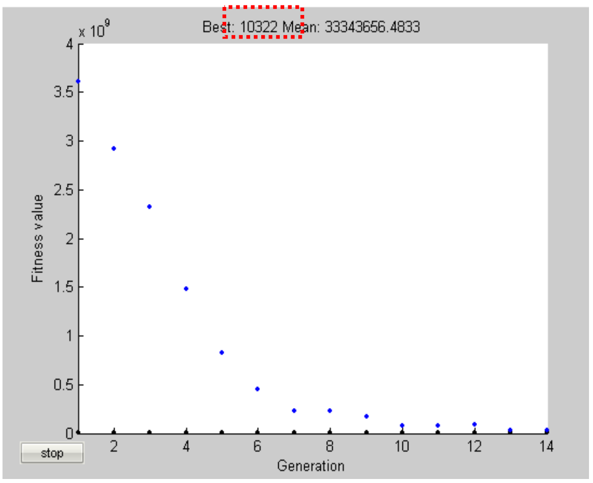


Fig. 7 The fitness value of GAs

IV. Discussion

In this paper, we present genetic algorithms (GAs) applied to the multi-product and multi-period inventory lot-sizing problem with supplier selection under storage space. Also a maximum storage space for the decision maker in each period is considered. The decision maker needs to determine what products to order in what quantities with which suppliers in which periods. The mathematical model is give and the use of the model is illustrated though a numerical example. The problem is formulated as a mixed integer programming and is solved with LINGO12 and the GAs. As compared to the solution of optimization package like LINGO12, the GAs solutions are superior.

References

[1] G.H. Goren, S. Tunali, and R. Jans, “A review of applications of genetic algorithms in lot sizing”, *Journal of Intelligent Manufacturing*. Springer Netherlands. 2008.

[2] H. M. Wagner, and T. M. Whitin, “Dynamic version of the economic lot-size model”, *Management Science*. 5, pp, 89–96, 1958.
[3] C. Basnet and J. M. Y. Leung, “Inventory lot-sizing with supplier selection”, *Computers and Operations Research*. 32, pp, 1–14, 2005.
[4] J. H. Holland, Adaptation in Natural and Artificial Systems. The University of Michigan Press, Ann Arbor. 1975.
[5] Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs. *AI Series*. Springer-Verlag, New York. 1944.
[6] M. Gen, and R. Cheng, Genetic Algorithms and Engineering Design. Wiley, New York. 1977.
[7] M. Gen, and R. Cheng, Genetic Algorithms and Engineering Optimization. Wiley, New York. 2000.
[8] L. Davis, The handbook of genetic algorithms, Van Nostrand Reinhold, New York. 1991.
[9] D. E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, Reading, MA. 1989.
[10] S. R. M. Mogador, A. Afsar, and B. Sohrabi, Inventory lot-sizing with supplier selection using hybrid intelligent algorithm. *Applied Soft Computing*. vol, 8, pp, 1523–1529, 2008.
[11] S. H. Chan, W. Chung, and S. Wadhwa, A hybrid genetic algorithm for production and distribution. *Omega*. 33, pp, 345–355, 2005.
[12] R. Sarker, and C. Newton, A genetic algorithm for solving economic lot size scheduling problem. *Computers and Industrial Engineering*. 42, 2002.
[13] K. Deb, Multi-Objective Optimization using Evolutionary Algorithms Wiley, Chichester, 2001.
[14] J. Rezaei, and M. Davoodi, Genetic algorithm for inventory lot-sizing with supplier selection under fuzzy demand and costs. *Advances in Applied Artificial Intelligence*. Springer-Verlag Berlin Heidelberg. 4031, 2006.
[15] J. Rezaei, and M. Davoodi, A deterministic, multi-item inventory model with supplier selection and imperfect quality. *Applied Mathematical Modelling*. 32, pp, 2106–2116, 2008.