Proceedings of the 2014 International Conference on Industrial Engineering and Operations Management
Bali, Indonesia, January 7 – 9, 2014

Designing a New Particle Swarm Optimization for Make-with-Buy Inventory Model

Saeed Poormoaied
Georgia Institute of Technology
Atlanta, United States of America

Mahsa Yavari
Ferdowsi University of Mashhad
Mashhad, Iran

Abstract

In real world, some manufacturing companies may encounter the production restriction. For instance, if the number of products increases, given company may not be able to produce all products. Therefore, it may encounter the backlogging. On the other hand, if product demand increases, given company may encounter restricted capacity to respond to such demands; so, it will also encounter backlogging. In this paper, we consider such companies that will encounter the mentioned conditions. To respond these exceeded demands, companies will be enforced to buy some products from outside. Therefore, the objective of this paper is to determine the optimum quantities of make and buy for each product by minimizing total inventory cost. We refer to the proposed model as make-with-buy model. In this paper we formulate the make-with-buy model and solve it by two meta-heuristic algorithms namely genetic algorithm (GA) and particle swarm optimization (PSO).

Keywords
Inventory cost; Make-with-buy; Multi-item; Restricted-production rate.

1. Introduction

In recent years, researchers add some consideration in the EOQ and EPQ models to increase their applications. The most of this consideration are defective item and rework process (Ben-Daya, 2002, Jamal et al., 2004, Jeang, 2001), constrained space (pasandideh and Niaki 2008, Pasandideh et al., 2010), time-varying demand (Goyal and Giri, 2003, Teng et al., 2005), partial backordering (Pentico and Drake, 2011) and maintenance operations (Liao and Sheu, 2011, Chelbi, A., Ait-Kadi, 2004). Our contribution in this paper is to investigate deterministic multi-product EPQ model with restriction in production capacity. Many of researchers propose a production policy that incurs shortage when the production capacity is insufficient to satisfy the annual demand. Goyal and Gopalakrishnan (1996) addressed the EPQ model with insufficient capacity. Leopoldo et al. (2009) develop an EPQ inventory model for a single product which is made in a single-stage manufacturing process that generates imperfect quality products and all defective products must be reworked at the same cycle. Drake et al. (2011) use EPQ for coordinated planning of a product and its components where the final product is subject to partial backordering of unfilled demand. Taleizadeh et al. (2010a, 2010b, 2010c, 2012) study Multi-product production quantity model with random defective items, failure in repair and service level constraints. The aim of these researches is to determine the optimal cycle length, optimal production quantity and optimal backordered quantity of each product such that the expected total cost is minimized. A piece of researches closely related to this paper is found in Hariga (1998) and chiu and chiu (2006). Hargia (1998) proposes a production-ordering policy to handle the situation of limited production capacity. This model is constrained to single item and the inventory control policy is to receive the external order at the stoppage time of the production batch. In this paper, we expand EPQ model to multi-products with limited production capacity. In proposed model, the demands of each product may be responded simultaneously through production in the company and ordering from the outside and the external order can be received at any time of inventory cycle. This paper is organized as follows: After this introduction, we deal with the significance of the proposed model. Section 3 indicates problem assumptions. Section 4 is devoted to problem modelling. Section 5 gives a numerical example. Finally, section 6 presents the conclusions of this paper.

2. Motivation and Significance of Model

In this paper, we deal with make-with-buy model. This model is applicable when production rate is more than demand rate, and when we encounter multi-item condition along with production capacity restriction. Assume that we produce several products by single machine, and this machine can produce all products without backlogging. When demand or the number of products increases, the machine may not be able to produce all products, so we will encounter backlogging and its huge costs. In such condition, we need to buy some products and hold them; by using this method we prepare more time (opportunity) for the machine to produce all products and to prevent backlogging. We refer to this issue as make-with-buy model. In this paper, we formulate make-with-buy model to minimize total inventory cost including holding cost, ordering cost and purchasing and production costs. After solving this model we determine optimum quantities of make and buy for each product.

3. Assumptions

1- Demand rate is certain and deterministic.
2- Production rate is certain and deterministic.
3- Shortage isn't allowable.
4- All items are produced by single machine.
5- The machine can produce one item at a time.
6- All items have continuous consuming.

4. Problem Modeling

In this section, first we apply the make-with-buy model for a single item to explain the methodology of modelling, and then this model is extended for multi items.

4.1 make-with-buy modelling: single item ($P > D$)
Parameters applied in modelling are as follows:
Q_1: Buying quantity, Q_2: Making quantity, R: Net inventory at the beginning of make in period T, P: Production rate, h: Holding cost rate, D: Demand rate, TMC: Total purchasing and Production (Material) costs, THC: Total holding cost, TOC: Total ordering cost, TIC: Total inventory cost, NS(t): Net inventory at time t, A_1: Ordering cost for buy, A_2: Ordering cost for production, C_1: Price of buy (each unit), C_2: Price of production (each unit), T: Duration of period, t_p: Duration of production in each period, S: Set-up time for production of each product.
According to Fig 1, to prevent backlogging, we buy specific amount of product at the beginning of the period; this amount is consumed with specific rate (demand rate) till net inventory decreases to R, then machine starts production for period t_p. As production rate is more than demand rate, net inventory in period t_p will increase. At the end of period t_p production stops and consumption continues till net inventory decreases to zero, then the other period launches as well. Note that in each period T, we buy and make only one time.

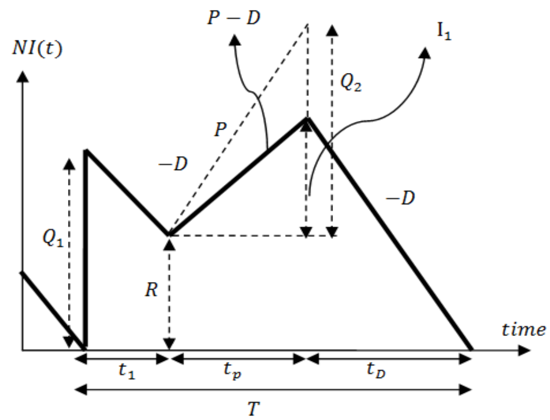

Figure 1. Net inventory diagram when P > D, single item

According to Fig.1, we can conclude that:

$$t_1 = \frac{Q_1 - R}{D} \quad , \quad t_p = \frac{Q_2}{P} \quad , \quad t_D = \frac{I_1 + R}{D} \quad ,$$

$$I_1 = t_p(P - D) \rightarrow \ I_1 = Q_2\left(1 - \frac{D}{P}\right)$$

$$t_1 + t_P + t_D = T \ \rightarrow T = \frac{Q_1 + Q_2}{D}$$

$$TIC = TMC + TOC + THC$$

$$TMC = (C_1Q_1 + C_2Q_2) \times \frac{1}{T} = (C_1Q_1 + C_2Q_2) \times \frac{D}{Q_1 + Q_2}$$

$$TOC = (A_1 + A_2) \times \frac{1}{T} = (A_1 + A_2) \times \frac{D}{Q_1 + Q_2}$$

$$THC = h \times \frac{1}{T}\left[\frac{(Q_1+R)(t_1)}{2} + \frac{(2R+I_1)(t_p)}{2} + \frac{(I_1+R)(t_D)}{2}\right] =$$

$$h \times \frac{D}{Q_1+Q_2}\left[\frac{(Q_1+R)(Q_1-R)}{2D} + \frac{\left(2R+Q_2(1-\frac{D}{P})\right)(Q_2)}{2P} + \frac{\left(Q_2(1-\frac{D}{P})+R\right)^2}{2D}\right]$$

Hence, TIC can be formulated as follows:

$$TIC = (C_1Q_1 + C_2Q_2) \times \frac{D}{Q_1+Q_2} +$$

$$(A_1 + A_2) \times \frac{D}{Q_1+Q_2} +$$

$$\frac{hD}{Q_1+Q_2}\left[\frac{(Q_1+R)(Q_1-R)}{2D} + \frac{\left(2R+Q_2(1-\frac{D}{P})\right)(Q_2)}{2P} + \frac{\left(Q_2(1-\frac{D}{P})+R\right)^2}{2D}\right]$$

TIC can be rewritten as follows, we name it Model 1.

Model 1*:*

$$TIC =$$

$$\frac{Q_1^2\left[\frac{h}{2}\right] + Q_2^2\left[\frac{h}{2}\left(1-\frac{D}{P}\right)\right] + Q_1(C_1D) + Q_2(C_2D) + RQ_2(h) + (A_1+A_2)\,D}{Q_1+Q_2}$$

Q_1, Q_2, and R are decision variables.

4.2 make-with-buy modelling: single item ($P < D$)

In this model, production rate is less than demand rate. So, net inventory diagram is as Fig. 2. According to Fig. 2, we can conclude that total inventory cost is as follows:

$$TMC = (C_1Q_1 + C_2Q_2) \times \frac{1}{T}$$

$$TOC = (A_1 + A_2) \times \frac{1}{T}$$

$$THC = h \times \frac{1}{T}\left[\frac{(Q_1+R_2)\times t_1 + R_2t_2}{2}\right]$$

$$TIC = TMC + TOC + THC$$

So, *Model* 2 is rewritten as follows:

TIC =

$$\frac{Q_1^2\left[\frac{h}{2}\right] + Q_2^2\left[\frac{h}{2}\left(1-\frac{D}{P}\right)\right] + Q_1(C_1D) + Q_2(C_2D) + RQ_2(h) + (A_1+A_2)\,D}{Q_1+Q_2}$$

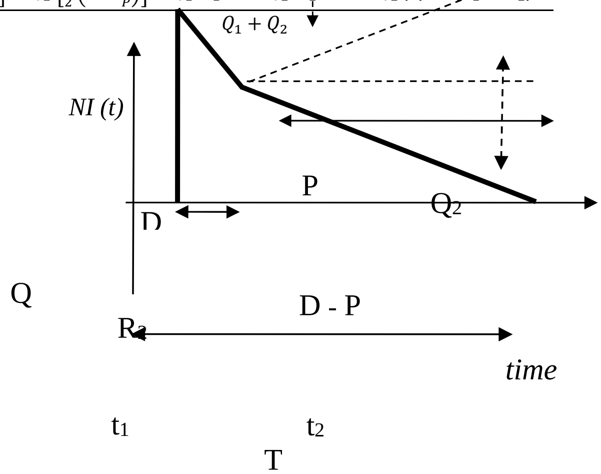

Figure 2. Net inventory diagram when P < D, single item

4.3 Make-with-buy modelling: Multi-item

We extend the model of section *A* and *B* to multi items produced by single machine. We express all parameters with index *i* so that *i* represents item *i*; e.g. C_{1i} expresses price of buy for product *i*.

Note that, we can produce all products by single machine if restriction 1 and 2 are satisfied:
Restriction 1:
$$\sum_{i=1}^{n} t_{pi} + \sum_{i=1}^{n} S_i \leq T \; ; \;\; i = 1,2,\dots,n$$
Restriction 2:
$$\frac{Q_{1i} + Q_{2i}}{D_i} = T \;\; ; \;\;\; i = 1,2,\dots,n$$
According to mathematical models presented in sections *A* and *B* as well as restriction 1 and 2, we can conclude that the mathematical modeling of proposed model with multi items can be summarized as follows:
Model 3:
$$Min\ STIC = \sum_{\forall i} \frac{D_i(C_{1i}Q_{1i} + C_{2i}Q_{2i}) + D_i(A_{1i} + A_{2i})}{Q_{1i} + Q_{2i}}$$
$$+ \frac{\frac{h_i}{2}\left(Q_{1i}{}^2 + Q_{2i}{}^2\left(1 - \frac{D_i}{P_i}\right) + 2R_{1i}Q_{2i}\right)}{Q_{1i} + Q_{2i}}$$
$$+ \sum_{\forall j} \frac{D_j}{Q_{1j} + Q_{2j}} \left[\left(C_{1j}Q_{1j} + C_{2j}Q_{2j}\right) + \left(A_{1j} + A_{2j}\right) + \frac{h_j}{2}\left(\frac{Q_{1j}{}^2\left(D_j - P_j\right) + R_{2j}{}^2 P_j}{D_j\left(D_j - P_j\right)} \right) \right]$$
$subject\ to$:
$$\sum_{k=1}^{n} \frac{Q_{2k}}{P_k} + \sum_{k=1}^{n} S_k \leq T$$
$$T_k = T \;\; ; \;\; k = 1,\dots,n$$
$$T_k = \frac{Q_{1k} + Q_{2k}}{D_k} ; \;\; k = 1,\dots,n$$
$$R_{1i} \leq Q_{1i} \;\; ; \forall\, i$$
$$Q_{2j} = \frac{P_j}{D_j - P_j} \times R_{2j} \;\; ; \;\;\; \forall\, j$$
$$R_{2i} \leq Q_{1j} \;\; ; \forall\, j$$
$$Q_{1k}, Q_{2k}, R_{2i}, R_{2i} \geq 0 \;\; ; \;\; k = 1,\dots,n$$
$$T \geq 0$$
For large *n*, the model is obviously so complicated that an optimal solution cannot be obtained by any analytical approach. To solve this model, it is required to apply a meta-heuristic method. In the next section, we will present an algorithm to efficiently solve it.

5. Solution Algorithm

The formulation given in *Model* 3 is a nonlinear and fractional integer-programming model. These characteristics cause the model to be adequately hard to solve by exact methods. Accordingly, we need a heuristic search algorithm to solve the proposed model. Historically, among the search algorithms, genetic algorithm (GA) and particle swarm optimization (PSO) have been successful in solving models similar to *Model* 3.

5.1 Genetic Algorithm

The usual form of GA was described by [7]. Genetic algorithms are stochastic search techniques based on the mechanism of natural selection and natural genetics. GA is differing from conventional search techniques in a sense that it starts with an initial set of random solutions called population. Each individual in the population is called a chromosome, representing a solution to the problem. The chromosomes evolve through successive iterations, called generations. During each generation, the chromosomes are evaluated, using some measures of fitness. To create the next generation, new chromosomes, called offspring, are formed by both crossover operator and mutation operator. A new generation is formed according to the fitness values of chromosomes. After several generations, the algorithm converges to the best chromosome. In the GA method, we present a chromosome by a matrix that has three rows and *n* columns. Each column shows the decision variables (EPQ, EOQ and R_1 or R_2) for each product. In addition, the rows of the matrix show EPQ, EOQ, and R_1 or R_2, respectively. Fig. 4 presents the general form of a chromosome.

Q_{21}	Q_{22}	Q_{23}	…	Q_{2n}
Q_{11}	Q_{12}	Q_{13}	…	Q_{1n}
R_{11}	R_{12}	R_{23}	…	R_{2n}

Figure 4. Chromosome representation

It should be noted that in chromosome representation products with D < P are inserted first and products with D > P are inserted second.

Initial population: A collection of chromosomes is randomly generated: we randomly generate Q_{2i} then, $Q_{1i} = T \times D_i - Q_{2i}$. Also R_{1i} are randomly generated so that $R_{1i} < Q_{1i}$. Finally, $R_{2j} = \frac{D_j - P_j}{P_j} \times Q_{2j}$. Therefore, GA operations including crossover and mutation are only applied to Q_{2i}. As an optimal solution may have $Q_{2i} = 0$, so we randomly generate some chromosomes with the $Q_{2i} = 0$. The population size is set to be 500. Additionally, ranking method is applied to selection chromosomes [9].

Crossover method: A uniform crossover is employed in this algorithm. In this type of crossover, at first, a random mask string whose length is the length of chromosome is generated. The bits of this string determine the parent whose corresponding bit will supply the offspring. This process is illustrated in Fig. 5 (this Figure shows just part of strings). The offspring 1 is generated by taking the genes from parent 1 if the corresponding mask bit is 1 and the genes from parent 2 if the corresponding mask bit is 0. Offspring 2 is created using the inverse of the mask string. Moreover, after several trials, we concluded that the appropriate crossover rate is 0.95.

Parent 1	1760	0	2356	1280	0
Parent 2	0	1250	0	1950	2340
mask	0	0	1	0	1
Offspring 1	1760	0	0	1280	2340
Offspring 2	0	1250	2356	1950	0

Figure 5. An example of the crossover operation

Mutation method: In this paper, the reverse mutation method proposed by [9] is used. In this method, two genes are randomly selected and then the genes between them are reversed. This process is illustrated in Fig. 6 (this Figure shows just part of strings). Moreover, after several trials, we can conclude that the appropriate mutation rate is 0.2.

Parent 1	1670	345	2356	1790	0

After mutation:

Parent 2	1670	0	1790	2356	345

Figure 6. An example of the mutation operation

Termination criteria: There are several stopping criteria for GA. In this study the algorithm is terminated when the maximum number of generation is reached. In this study, GA after 500 generations is terminated. Maximum number of generation is depends on the expectation of the user.

5.2 particle swarm optimization

In this paper, particle swarm optimization (PSO) is adopted as another meta-heuristic search method for make-with-buy model for comparison to GA. The algorithm of PSO used in this paper is summarized as follows:

Let f: $R^n \rightarrow R$ be the fitness or cost function that must be minimized. Let *S* be the number of particles in the swarm, each having a position $X_i \in R^n$ in the search-space and a velocity $V_i \in R^n$. Let P_i be the current best position of particle *i* and let g be the best known position out of the entire swarm. A basic PSO algorithm is then:

For each particle *i* = 1,…, *S* do:

- Initialize the particle's position: $X_i \sim U(b_{lo}, b_{up})$, where b_{lo}, b_{up} are the lower and upper boundaries of the search-space and $U(0,0)$ is a uniformly distributed random vector.

- Initialize the best known position: $P_i \leftarrow X_i$

- Initialize the velocity: $V_i \sim U(-d, d)$, where:$d = |b_{up} - b_{lo}|$

- Initialize the best known position of the swarm: $g \leftarrow \arg\min_{P_i} f(P_i)$

Until a termination criterion is met (e.g. number of iterations performed, or adequate fitness reached), repeat:

For each particle i = 1, … , S do:

Create random vectors $r_p, r_g \sim U(0,1)$.

Update the particle's velocity:

$V_i \leftarrow wV_i + \emptyset_p r_p \circ (P_i - X_i) + \emptyset_g r_g \circ (g - X_i)$, where the ° operator indicates element-by-element multiplication.
Update the particle's position: $X_i \leftarrow X_i + V_i$, note that this is done regardless of improvement to the fitness and Bound X_i as follows: $X_i \leftarrow \text{Bound}\ (X_i;\ b_{lo_i};\ b_{up_i})$
If $(f(X_i) < f(P_i))$ do:
Update the particle's best known position: $P_i \leftarrow X_i$
If $(f(P_i) < f(g))$ update the swarm's best position: $g \leftarrow P_i$
Now g holds the best found solution.
w is decreased linearly form 1 to 0.4 during the run as well as $\emptyset_p$ and $\emptyset_g$ are set 2.

C. Numerical Example

In this paper, we firstly construct some instance shown in table I. In this table, parameters are randomly generated from: D_i = U [1000, 4000], P_i = U [2000, 6000], C_{1i} = U [20, 30], C_{2i} = U [10, 20], A_{1i} = U [100000, 200000], A_{2i} = U [50000, 150000], h_i = U [100, 200], S_i = U [0.001, 0.01]; i = 1,…,15. In table II, 15 numerical examples for comparing GA and PSO are randomly constructed. The results in table II show that PSO in more stable and exact than GA. Therefore, PSO is selected as a suitable algorithm to solve the proposed model. In this table, columns represent the number of products, the products, the average of 20 times running for PSO, the average of 20 times running for GA, the standard division of 20 times running for PSO, the standard division of 20 times running for GA, the running time for PSO and GA, respectively. It should be emphasized that the running time for two algorithms is nearly the same. Fig. 7 and 8 show the GA and PSO convergence for instance 7 (7 products), respectively. The proposed algorithms have been coded with C and run on an IBM-compatible PC with a Pentium 2.33 GHz processor under Windows 7.

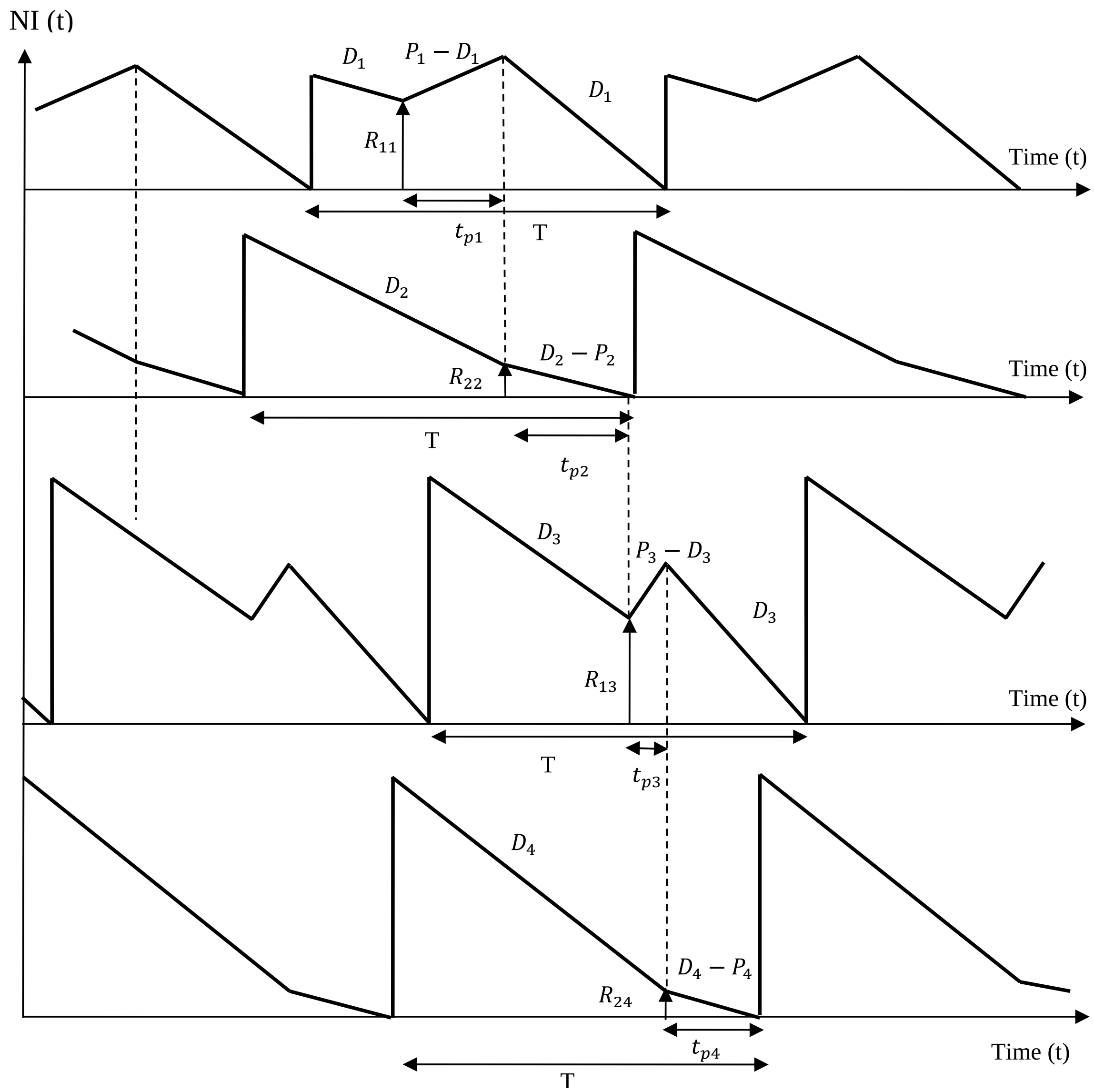

Figure 3. Net inventory diagram; *n*

Table 1. Product Related Data

S_i	h_i	A_{2i}	A_{1i}	C_{2i}	C_{1i}	P_i	D_i	Product
0.01	110	76846	100833	12	23	3915	2972	1
0.003	163	61519	102937	15	22	4469	3682	2
0.005	193	80730	129790	11	28	3890	2004	3
0.003	105	81437	131873	14	20	3820	3615	4
0.005	194	67816	123881	14	21	3584	1270	5
0.006	189	59418	124256	13	30	4446	3927	6
0.008	130	50659	126910	16	27	4690	1970	7
0.002	153	67609	123182	20	24	3057	3135	8
0.009	108	68380	117008	17	27	3529	4168	9
0.01	113	60220	108234	11	25	1938	5272	10
0.005	152	78659	103569	18	20	3690	5782	11
0.006	107	67226	119348	19	20	1110	4887	12
0.007	194	58144	118435	11	26	3381	5292	13
0.003	197	69102	115456	17	25	1297	5872	14
0.008	162	65731	116450	14	30	2971	4779	15

Table 2. Instances and Results

No. Products	Products	Mean. PSO	Mean. GA	Dev. PSO	Dev. GA	Time. PSO	Time. GA
2	1,8	580375	581029	25.1	52.81	5.43	7.11
3	3,6,15	620974	622389	85.1	170.55	5.56	7.29
4	5,8,9,15	878565	880031	189.1	336.7	6.76	7.49
5	1,8,9,11,14	955783	957152	102	328.6	7.35	8.71
6	3,6,9,11,13,15	1610776	1618806	142	962.1	8.68	9.83
7	4,6,7,8,9,10,12	2058481	2059786	407	395.1	9.36	10.33
8	5,7,9,10,12,13,14,15	2483047	2691292	163	461.70	10.57	11.09
9	1,3,4,6,9,10,12,13,14	3489225	3505852	160	760.2	10.34	12.15
10	1,2,6,7,8,9,10,11,12,13	3912132	3938976	185	441.24	11.67	13.76
11	1,2,4,5,6,7,8,9,12,13,15	4170227	4189946	132	1532.4	13.45	14.69
12	1,2,3,4,5,7,8,9,10,12,14,15	4488272	4492455	125	816.24	15.24	15.89
13	1,2,3,4,5,6,7,9,10,12,13,14,15	4630385	5640193	109.4	611.5	11.67	16.12
14	1,2,3,4,5,6,7,8,9,10,12,13,14,15	4700043	4700267	315.3	785.2	15.24	18.53
15	1,2,3,4,5,6,7,8,9,10,11,12,13,14,15	4801694	4818939	175.1	421.9	17.46	19.78

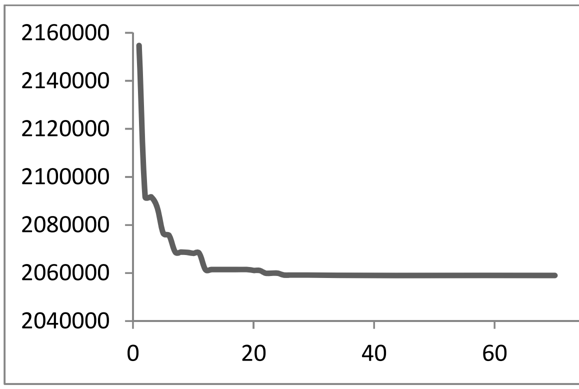

Figure 7. The graph of convergence path for GA with the following solution

0	28	0	58	0	15	0
123	151	320	239	210	12	56
0	55	0	104	0	7	0

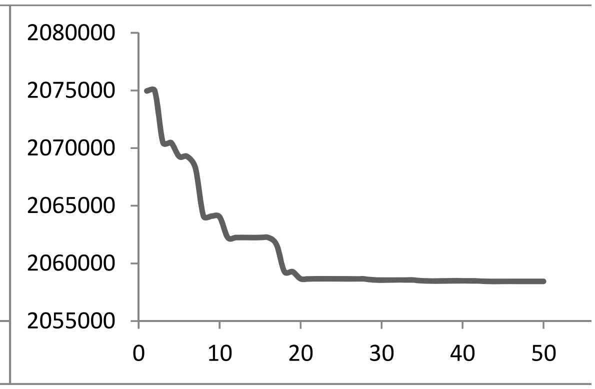

Figure 8. The graph of convergence path for PSO with the following solution

0	28	0	60	0	15	0
125	151	320	239	211	12	59
0	45	0	104	0	3	0

6. Conclusion

When a company thrives, its customer demands or the number of products increases. If we assume that the number of machines in this company is constant, it cannot respond to all customer demand, hence this company encounters shortage and its huge costs. Such companies to repel this condition decide to buy amount of some products from the outside. In this paper, we formulated this problem and determined optimum quantities of make and buy in which total inventory cost is minimized. One of the most important applications of proposed model is in outsourcing strategy and supply chain management. In these strategies, manager decides which and how much of products must be bought or outsourced. Finally, to solve the proposed model, we utilized genetic algorithm and particle swarm optimization as well as concluded that PSO in more stable and efficient than GA.

References

[1] Chelbi, A., and Ait-Kadi, D. (2004). Analysis of a production/inventory system with randomly failing production unit submitted to regular preventive maintenance, *European Journal of Operational Re*search, 156, 712–718.

[2] Chiu, Y.S.P. and Chiu, S.W., (2006). Determining the materials procurement policy based on the economic order/production models with backlogging, *The International Journal of Advanced Manufacturing Technology*, 30 (1-2), 156-165

[3] Daya, B.M. (2002). The economic production lot-sizing problem with imperfect production processes and imperfect maintenance. *International Journal of Production Economics*, 76, 257-264.

[4] Eduardo, L., and Barrón, C. (2009). Economic production quantity with rework process at a single-stage manufacturing system with planned backorders, *Computers & Industrial Engineering*,57, 1105–1113.

[5] Goyal, S.K., and Giri, B.C. (2003). The production-inventory problem of a product with time varying demand, production and deterioration rates, *European Journal of Operational Research*, 147 (3), 549-557.
[6] Goyal, S.K., and Gopalakrishnan, M. (1996). Production lot sizing model with insucient production capacity, *Production Planning & Control*, 7, 222± 224.
[7] Hariga, M.A. (1998). Economic production-ordering quantity models with limited production capacity, *Production Planning & Control*, 9 (7), 671± 674.
[8] Jamal, A.M.M., Sarker, B.R., and Mondal, S. (2004). Optimal manufacturing batch size with rework process at a single-stage production system, *Computers & Industrial Engineering*, 47, 77-89.
[9] Liao, G.L., HueiSheu, S. (2011). Economic production quantity model for randomly failing production process with minimal repair and imperfect maintenance, *International Journal of Production Economics*, 13, 118-124.
[10] Pasandideh, S.H.R., and Akhavan Niaki, S.T. (2008). A genetic algorithm approach to optimize a multi-products EPQ model with discrete delivery orders and constrained space, *Applied Mathematics and Computation*, 195, 506-514.
[11] Pasandideh, S.H.R., Akhavan Niaki, S.T., and Yeganeh, J.A. (2010). A parameter-tuned genetic algorithm for multi-product economic production quantity model with space constraint, discrete delivery orders and shortages, *Advances in Engineering Software*, 41, 306-314.
[12] Pentico, D.W., and Matthew, J. D. (2011). A survey of deterministic models for the EOQ and EPQ with partial backordering, *European Journal of Operational Research*, 214, 179-198.
[13] Taleizadeh, A.A., Akhavan Niaki, S.T., and Najafi, A.A. (2010,b). Multiproduct single-machine production system with stochastic scrapped production rate, partial backordering and service level constraint, *Journal of Computational and Applied Mathematics*, 233, 1834-1849,
[14] Teng, J.T., Ouyang, L.Y., and Chang, C.T. (2005). Deterministic economic production quantity models with time-varying demand and cost, *Applied Mathematical Modelling*. 29, 987-1003.