Optimization of An Inventory System Using PSO And FA Algorithm

Nitin Sangana , Aravindan Nallappa Ravi, Vishnu Prasadh Sugumar
Department of Mechanical Engineering, University of Texas At Dallas
nxs180045@utdallas.edu, anr18000@utdallas.edu, vxs170013@utdallas.edu

***Abstract* — Nowadays, a key concern for all the industries is to reduce the inventory and inventory driven costs and maintain them at an optimum level in the supply chain system. The dynamic nature of the excess stock level and shortage level in the supply chain system poses a challenge for efficient inventory management. The complexity of the problem increases with the inclusion of multiple products from multiple factories and involvement of more distribution centers and agents. The supply chain costs can be significantly reduced by implementing optimal inventory control methodologies. These techniques control the inventory in such a way that, the members of the supply chain will remain unaffected by surplus as well as shortage of inventory. This paper presents a comparison between two different optimization techniques and the results which provide a proof for showing an efficient algorithm of the two to optimize an inventory system.**

I. INTRODUCTION

In today's competitive business world, companies require small lead times, low costs and high customer service levels to survive. So, companies are working to continuously optimize the processes to reduce the mentioned factors [7]. Organizations that have focused on cycle time as a productivity measure can reduce delivery time and improve quality, thereby creating more satisfied customers and increasing the profit by reducing the costs associated.

Inventory is a part and parcel of every facet of business life. Without it, no business activity can be performed whether, it being a service organization like hospitals and banks etc. or manufacturing or trading organization. Irrespective of specific organizational setting, inventories are reflected by way of a conversion process of inputs to outputs depending on the business goals and needs [10]. Inventory is simply a stock of physical assets having some economics value, which can be either in the form of money, labor or material. It can also be regarded as those goods which are procured, stored and used for day-to-day or periodic functioning of the organization. This can usually be in the form of physical resource such as raw materials, semi-finished goods (work in progress goods), finished goods, goods for resale, stocks in transit, consignment stocks and maintenance goods[11].

Most of the management do not like the inventories because they are like money placed in a drawer, assets tied up in investments a borrowing cost. Inventories also incur various other costs like : (i) administrative cost of placing an order, called reorder cost or set cost; (ii) cost of maintaining an inventory, called inventory holding cost a carrying cost, which includes storage charge, interest, insurance, etc., a (iii) shortage cost is a loss of profit, goodwill, etc., when run out of stock (iv)ordering cost [12]. All the above should be optimized for efficient supply chain management. In this paper our focus is to optimize the cost which comes which the ordering of goods/stocks from vendor or manufacturer and cost used to storing of goods. This uses the optimal value of products which is to be stored to minimize both mentioned costs. We significantly used firefly and particle swarm algorithm and compared the results to identify the efficient way which can be used to optimize the costs for inventory [13].

II. PARTICLE SWARM OPTIMIZATION

Particle swarm optimization is originally attributed to Kennedy and Eberhart. (PSO) is a stochastic population based evolutionary computational method that optimizes a problem by improving the solutions in the search space by performing multiple iterations. It mimics natural phenomena like flock of birds or school of fish, as shown in figure 1. PSO is a metaheuristic as it makes few or no assumptions about the problem being optimized and can search very large spaces of candidate solutions [1]. However, metaheuristics such as PSO do not guarantee an optimal solution is ever found. PSO flowchart is shown in figure 2. Also, PSO does not use the gradient of the problem being optimized, which means PSO does not require that the optimization problem be differentiable as is required by classic optimization methods such as gradient descent and quasi-newton methods.

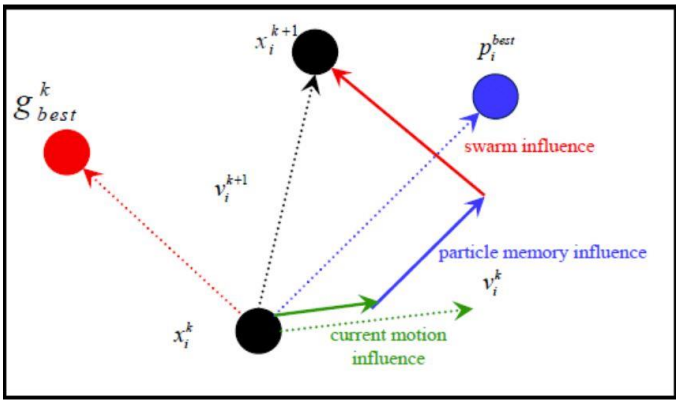

Figure 1. PSO algorithm

Initially, a set of solutions (150 input data points in our case) are randomly spread across the search space. Then, they are made to move around in the search space based on mathematical model over the particle's(solution) position and velocity [2]. Here, the particles have memory and keep track of their personal best position (P_{best}) . Moreover, the particle with best fitness function value will become the global best (G_{best}).

These influence the motion of the particle's movement and they are guided towards best known positions in search space, which are further updated as better positions are found by other particles. The two components (velocity and position) of particles need to be updated in each iteration. So, this moves the swarm towards the best solution. The mathematical model of PSO for updating velocity equation (1) and equation(2) is as follows:

$$V_i^{t+1} = wV_i^t + c_1 r_1 (P_i^t - X_i^t) + c_2 r_2 (G^t - X_i^t) \quad -(1)$$

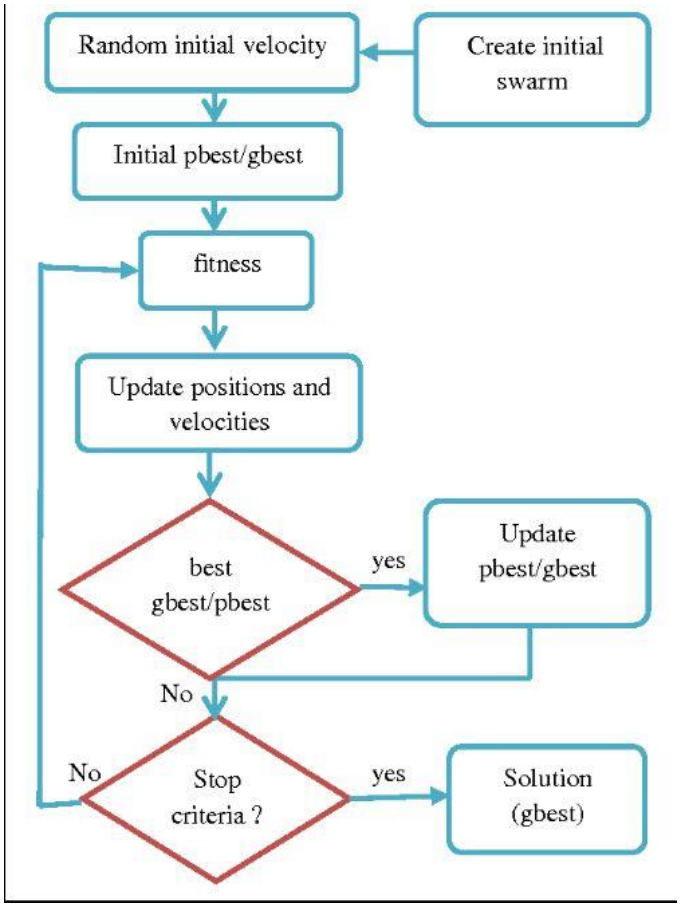

Figure 2. PSO algorithm flowchart

Here, V_i^{t+1} – velocity of the particle in the next iteration

w – inertia
V_i^t – velocity of the particle in the current iteration
c_1 – cognitive acceleration coefficient ;
c_2 – social acceleration coefficient;
r_1, r_2 – random numbers $\in \{0,1\}$;
P_i^t – personal best;
G_i^t – global best;

$$X_i^{t+1} = X_i^t + V_i^{t+1} - (2)$$

Where, X_i^t – Particle position;
X_i^{t+1} – New particle position.

In Eq (1), first term provides momentum, second term is the cognitive component which represents personal thinking of the particle and third term represents social component indicating collaborating effect [3]. After the position and velocity of the individual particle in the swarm is updated, the stopping criteria is checked. If the criteria is met, it stops, otherwise the next iteration is performed.

There are two types of behavior exhibited by the swarm:

1. Exploratory behavior - searching a broader region of the search-space (related to personal thinking).
2.Exploitative behavior - a locally oriented search so as to get closer to a (possibly local) optimum. (related to social component).

The choice of PSO parameters can have a large impact on optimization performance. These parameters are 'w','c_1' and 'c_2'. It is observed that, exploration was at the lowest level when 'w' value is low and highest at a higher value of 'w'. Similarly, consider the values of 'c_1' and 'c_2'. When,'c_1' > 'c_2', exploitation is at the lowest level and when 'c_1' < 'c_2', exploitation is at a higher level. PSO algorithm and its parameters must be chosen to properly balance between exploration and exploitation to avoid premature convergence to a local optimum yet still ensure a good rate of convergence to the optimum [14].

III. FIREFLY ALGORITHM

Firefly algorithm (FA) was first developed by Yang in 2007.The firefly algorithm is one of the most efficient bio-inspired algorithms, which has a strong capability to solve continuous and discrete optimization problems. It solves optimization problems based on the flashing patterns and behavior of fireflies [4]. Fireflies are insects characterized by their magnificent flashing light produced by a process of bioluminescence. The main role of such flashes can be reduced to the following two tasks:
(i) First task is to attract mating partners and the,
(ii) second is to remind potential predators of the bitter taste of fireflies.

The movement of a firefly I is attracted to another more attractive (brighter) firefly J is determined by equation (3):

$$x_i^{t+1} = x_i^t + \beta_o e^{-\gamma r_{ij}^2}(x_j^t - x_i^t) + \alpha \varepsilon_i^t - (3)$$

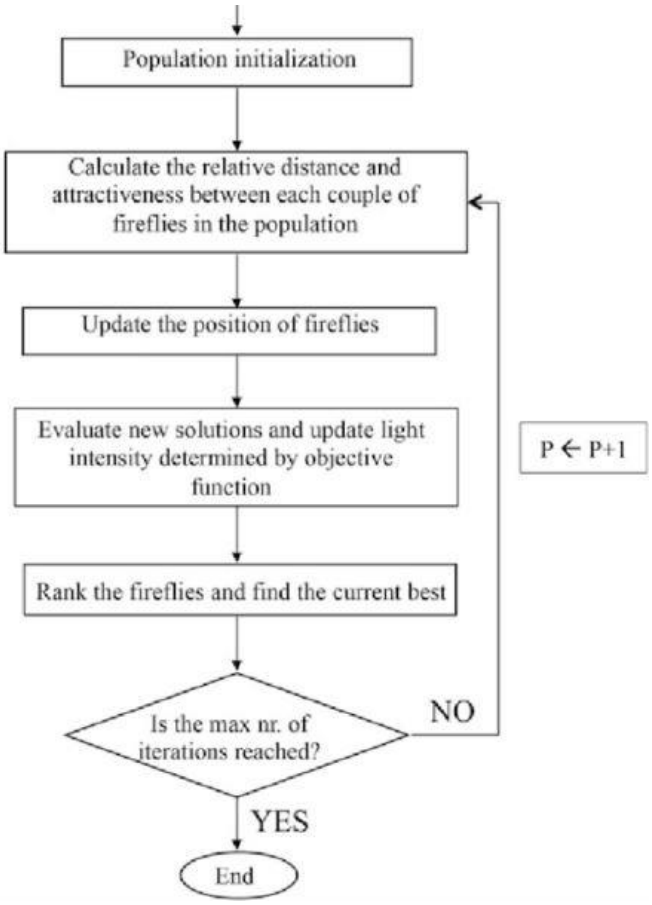

Figure 3. FA algorithm flowchart

Where,$'\beta_o'$is the attractiveness at the distance $r = 0$, and the second term is due to the attraction. The third term is randomization with $'\alpha'$ being the randomization parameter ,and $'\varepsilon_i^t'$ is a vector of random numbers drawn from a Gaussian distribution or uniform distribution at time t. If $'\beta_o' = 0$, it becomes a simple random walk.

Figure 3 shows the flowchart that represents the processes done in the FA algorithm [6]. If the model is analyzed more precisely, the following can be inferred, (i) If γ is very large, then attractiveness or light intensity decreases too quickly, this means that the second term in equation (3) becomes negligible, leading to the standard simulated annealing (SA). On the other hand, (ii) If γ is very small (i.e., $\gamma \to 0$), then the exponential factor exp $(-\gamma_{ij}^2) \to 1$.

IV. EXPLANATION OF THE CODE

P1	59	34	84	69	28	25	55	65	74	36
P2	84	34	31	46	78	36	80	64	22	40
P3	57	56	50	79	40	54	22	90	41	37

Table 1. Demand for each product

P1	188	138	176	104	153	133	200	189	163	181
P2	149	129	117	181	196	173	182	117	188	183
P3	147	173	188	182	185	103	120	171	200	154

Table 2. Product Ordering Cost

P1	3	6	10	2	4	7	3	7	3	10
P2	8	10	4	8	5	5	5	4	7	6
P3	3	9	5	7	8	8	8	4	3	6

Table 3. Inventory Holding Cost

Table 1, table 2 and table 3 shows the inventory model considered for optimization. P1, P2 and P3 are three different products. It shows the product quantities at ten different time intervals. For implementing the two algorithms for the inventory model, we made use of MATLAB. The code starts with giving an option to choose between Firefly algorithm and Particle Swarm Optimization, according to the user's wish.

The two decision variables (product quantity and time horizon) are generated randomly with the given bounds. The fitness function here is the total inventory cost, and the objective is to minimize it.

Min I.C = O.C + I.H.C

With decision variables = Product Quantity and Time Interval where,

I.C – Inventory Cost
O.C – Ordering Cost
I.H.C – Inventory Holding Cost

For simplicity, only the product ordering cost and inventory holding cost for each product are considered. First, the product ordering cost is to be calculated, and for this, we need the ordering cost of each product (which is already defined in the inventory model) and the order quantity of each product is required. The ordering quantity is calculated by defining limits and then randomly generating numbers. This would give the ordering cost. For the inventory holding cost, the inventory level at each interval is needed. The current inventory level is calculated by subtracting the demand from the sum of ordered product quantity and the inventory at previous stage. So, once the inventory holding cost and the ordering cost of each product is determined, the inventory cost is determined for each particle in the swarm.

The code initially starts by defining the inventory model, which include parameters like demand for each product, ordering cost of each product, inventory holding cost of each product, initial inventory level, volume occupied by each product in the inventory, maximum quantity of products the inventory can hold.

For the PSO algorithm, first define parameters like limits for velocity parameter and particle parameter, the fitness function, number of decision variables. Then define parameters needed for the PSO loop like the maximum

number of iterations, population size, constants like 'w','c_1' ,'c_2',w_{damp}, etc.

Once all the required parameters are defined, the matrices like position, cost (fitness function value), velocity, etc required to calculate for each particle in the swarm at each iteration are initialized to zero. Once all required parameters are defined and initialized, the PSO loop starts by calculating the P_{best}, G_{best}, value of fitness function, velocity and position of each particle in the swarm. Then according to each particle's value of the fitness function, the P_{best} and G_{best} matrices are updated for each variable. This process is repeated for a specific number of iterations or until a criterion is satisfied, and the G_{best} value at the end would be the optimized solution for the system.

The Firefly algorithm employs the same mechanism, except that at each iteration, unlike the PSO where a lot of parameters are calculated, only the distance by which a particle should move toward the on which has a better fitness function value is determined.

V. RESULTS AND DISCUSSION

The graphs show us the results obtained when the inventory is optimized using PSO algorithm. Figure 4 shows us the convergence of the optimal solution as the number of iterations increase. The optimal solution pretty much appears after roughly 100 iterations, and it does not change substantially after that. Figure 5 shows the product quantities at different intervals after 500 iterations. The graphs show us the results obtained when the inventory is optimized using FA algorithm.

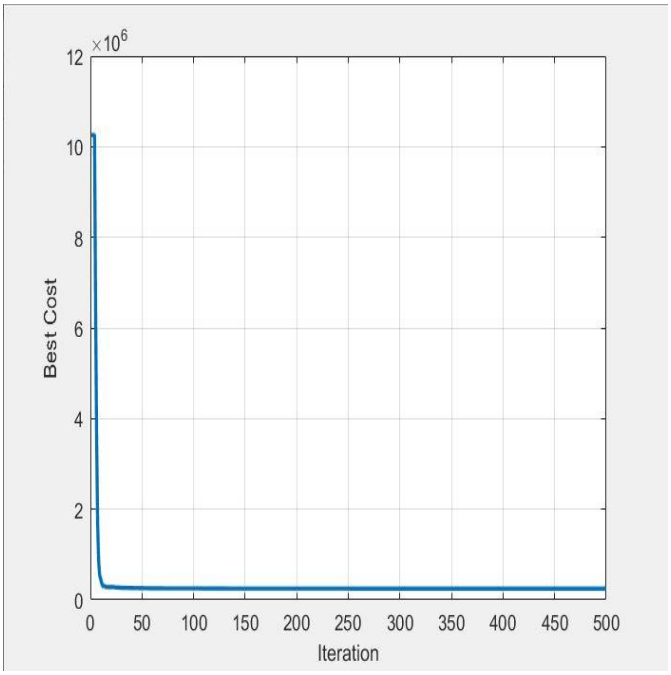

Figure 4. Fitness function (vs) Iteration level for PSO algorithm

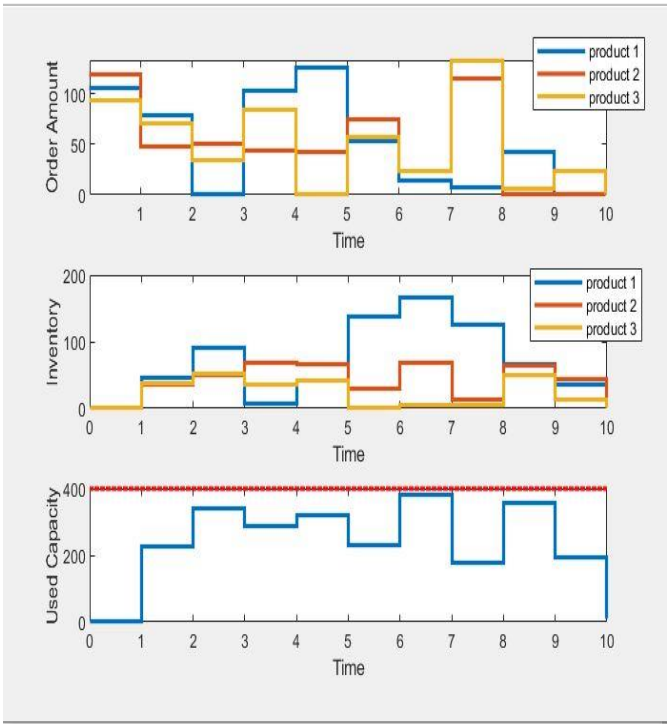

Figure 5. Product quantity (vs) time intervals for PSO algorithm

The optimal solution pretty much appears after roughly 100 iterations in this case also (as shown in figure 6), and it does not change substantially after that. Figure 7 shows the product quantities at different intervals after 1000 iterations.

From the above obtained results, Firefly algorithm gives better results that the PSO algorithm. For example, considering the order quantity graph, in the interval between time 0 and 1, it is evident that FA gives us a less quantity of product 1 to be ordered when compared to PSO. This intern reduces the overall inventory cost.

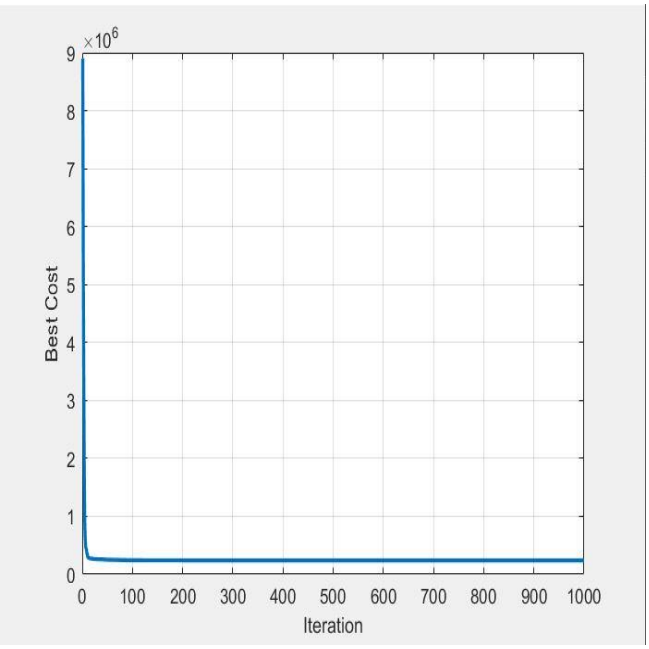

Figure 6. Fitness function (vs) Iteration level for FA algorithm

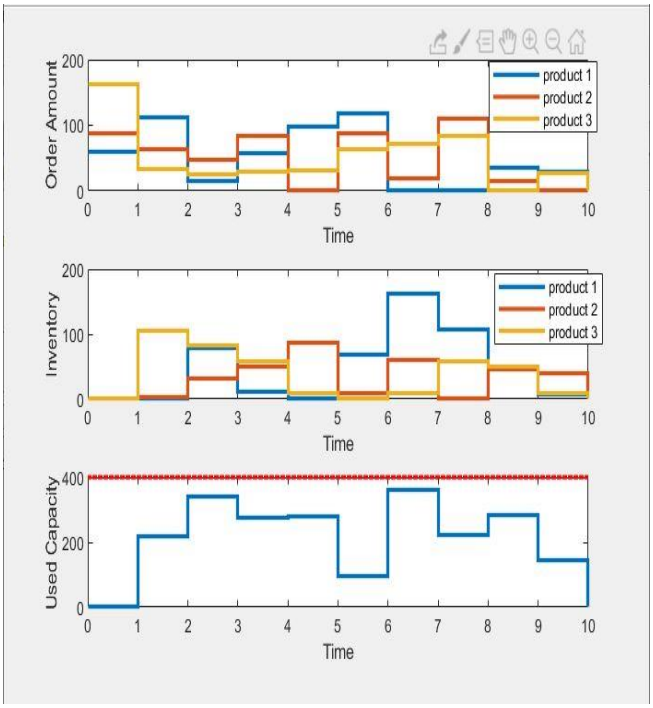

Figure 7. Product quantity (vs) time intervals for FA algorithm

VI. CONCLUSION:

This project utilizes two different optimization techniques to optimize an inventory system and significantly reduce the costs. Thus, upon comparing the results obtained from both the algorithms, Firefly algorithm gives a more optimized result when compared to PSO algorithm. Firefly algorithm has been introduced recently and it utilizes fewer parameters when compared to the PSO algorithm. When adding this to the fact that the math behind the FA algorithm is not complicated, has higher overall efficiency, this algorithm implementation becomes a lot easier. The total inventory cost obtained by optimizing the inventory model considered is shown below:

Particle Swarm Optimization	$ 245,421
Firefly Algorithm	$ 240,012

REFERENCES

[1] KyoungJong Park, Gyouhyung Kyung, "Optimization of total inventory cost and order fill rate in a supply chain using PSO", The International Journal of Advanced Manufacturing Technology, 2014, Volume 70, Number 9-12, Page 1533.

[2] Seyed Mohsen Mousavi, Vahid Hajipour, Seyed Taghi Akhavan Niaki, Najmeh Aalikar, "A multi-product multi-period inventory control problem under inflation and discount: a parameter-tuned particle swarm optimization algorithm", The International Journal of Advanced Manufacturing Technology, 2014, Volume 70, Number 9-12, Page 1739.

[3] Chi-Yang Tsai, Szu-Wei Yeh, "A multiple objective particle swarm optimization approach for inventory classification", International Journal of Production Economics, Volume 114, Issue 2, 2008, Pages 656-666, ISSN 0925-5273.

[4] Mariam Elkhechafi, Zoubida Benmamoun, Hanaa Hachimi, Aouatif Amine, Youssfi Elkettani, "Firefly Algorithm for Supply Chain Optimization", Lobachevskii Journal of Mathematics, 2018, Volume 39, Number 3, Page 355.

[5] Mohammad Kazem Sayadi, Ashkan Hafezalkotob, Seyed Gholamreza Jalali Naini , "Firefly-inspired algorithm for discrete optimization problems: An application to manufacturing cell formation", Journal of Manufacturing Systems, Volume 32, Issue 1, 2013, Pages 78-84, ISSN 0278-6125.

[6] Shu, T.; Gao, X.; Chen, S.; Wang, S.; Lai, K.K.; Gan, L. , "Weighing Efficiency-Robustness in Supply Chain Disruption by Multi-Objective Firefly Algorithm.", Sustainability **2016**, *8*, 250.

[7] Taleizadeh, Ata Allah and Leopoldo Eduardo Cárdenas-Barrón., "Metaheuristic Algorithms for Supply Chain Management Problems", In Meta-Heuristics Optimization Algorithms in Engineering, Business, Economics, and Finance, ed. Pandian M. Vasant, 110-135 (2013).

[8] M. K. Marichelvam, T. Prabaharan and X. S. Yang, "A Discrete Firefly Algorithm for the Multi-Objective Hybrid Flowshop Scheduling Problems," in IEEE Transactions on Evolutionary Computation, vol. 18, no. 2, pp. 301-305, April 2014.

[9] Markus Ettl, Gerald E. Feigin, Grace Y. Lin, David D. Yao, "A Supply Network Model with Base-Stock Control and Service [20] Requirements", OPERATIONS RESEARCH Vol. 48, No. 2, March-April 2000, pp. 216-232DOI: 10.1287/opre.48.2.216.12376.

[10] S.R. Singh, Tarun Kumar, "Inventory Optimization in Efficient Supply Chain Management", International Journal of Computer Applications in Engineering Sciences, VOL I, ISSUE IV, DECEMBER 2011, ISSN: 2231-4946

[11] B.M. Beamon, "Supply chain design and analysis: Models and methods", International Journal on Production Economics, 55: 281-294 1998.

[12] Y. Barlas andA. Aksogan,1996, "Product Diversification and Quick Response Order Strategies in Supply Chain Management", 14th International Conference of the System Dynamics Society1996Cambridge, Massachusetts, USA, pp. 51-56.[16].

[13] H.L.Leeand C. Billington, 1995. The evolution of supply-chain-management models and practice at Hewlett-Packard. Interface, 25: 42-63.

[14] S. Narmadha, Dr. V. Selladurai ,G. Sathish, " Efficient Inventory Optimization of Multi Product, Multiple Suppliers with Lead Time using PSO", International Journal of Computer Science and Information Security, IJCSIS, Vol. 7, No. 1, pp. 180-189, January 2010, USA.