A fuzzy lifetime-based particle swarm optimisation with varying swarm size to solve a production inventory model

Partha Guchhait*

Department of Mathematics,
Vidyasagar University,
Midnapore, Paschim-Medinipur, West Bengal, 721102, India
Email: parthaguchhait@gmail.com
*Corresponding author

Manas Kumar Maiti

Department of Mathematics,
Mahishadal Raj College,
Mahishadal, Purba-Medinipur, West Bengal, 721628, India
Email: manasmaiti@yahoo.co.in.

Abstract: Here, a modified particle swarm optimisation (MPSO) algorithm with varying swarm size for constrained optimisation problem is proposed. In this MPSO, a life time is assigned to each particle at the time of generation depending on its fitness. After completion of a generation, if no movement is made by the particle, its age is increased by unity. When age of a particle exceeds the lifetime, it is discarded from the swarm. Diversity in the swarm is maintained using information entropy theory. A fuzzy possibility/necessity-based fitness evolution is proposed to deal with fuzzy optimisation problems using this MPSO. Efficiency of the algorithm is tested against a list of crisp valued standard benchmark nonlinear test functions. This algorithm is used to solve a production inventory model with fuzzy costs, where lifetime of the product is random in nature. At the beginning of planning horizon price discount is offered to the customers for few cycles to boost the demand. Demand also depends on stock and selling price. The model is illustrated with numerical examples and some sensitivity analyses have been made.

Keywords: modified PSO algorithm; fuzzy lifetime; varying swarm size; possibility/necessity measure; EPQ model; price discounted promotional demand; random planning horizon.

Reference to this paper should be made as follows: Guchhait, P. and Maiti, M.K. (2016) 'A fuzzy lifetime-based particle swarm optimisation with varying swarm size to solve a production inventory model', *Int. J. Computational Complexity and Intelligent Algorithms*, Vol. 1, No. 1, pp.68–98.

Biographical notes: Partha Guchhait is a Teacher of Mathematics at Dherua Anchal Satabala High School, Paschim-Medinipur, West Bengal, India. He received his BSc in Mathematics and MSc in Applied Mathematics from Vidyasagar University, Midnapore, Paschim-Medinipur, West Bengal, India. He recieved his PhD (2013) in Applied Mathematics from Vidyasagar

University. He has ten publications so far in reputed research journals. His research interests include inventory control, fuzzy mathematics, soft computing techniques, multi-objective optimization, transportation problems, knapsack problems, etc.

Manas Kumar Maiti is working in the Department of Mathematics, Mahishadal Raj College, Mahishadal, Purba-medinipur, West Bengal, India. He received his MSc and MPhil in Applied Mathematics from Calcutta University, Midnapore, Paschim-Medinipur, West Bengal, India, and MTech in Computer Science from I.I.T Kharagpur, West Bengal, India. He received his PhD (2006) in Applied Mathematics from Vidyasagar University, India. He has more than 40 publications in reputed research journals. His fields of interest are inventory control, transportation problem, travelling salesman problem, portfolio management, fuzzy mathematics, soft computing techniques, etc.

1 Introduction

The particle swarm optimisation (PSO) is a exhaustive search algorithm based on the motion of a flock of birds searching for food. A PSO starts with a set of potential solutions which is called swarm of the decision-making problem under consideration. Individual solutions are called particles and food is analogous to optimal solution. Since the introduction of the PSO, it is modified and the process is still going on. Here, this algorithm is modified with the introduction of life-time for each particle of a swarm at the time of their birth and it is done depending on their fitness. After completion of a generation, if no movement is made by a particle, its age is increased by unity. It is discarded from the swarm when age exceeds its life-time. At the time of generation of initial swarm, diversity in the swarm is maintained using information entropy theory. Diversity in the swarm is checked after a fixed number of generations, called period. After each period, swarm size may change depending on the diversity of the swarm. If diversity level drops below the fixed threshold, some new (child) particles are included in the swarm. Child particles are obtained using fuzzy life-time-based crossover and mutation operations on the particles of the swarm. Similarly, if the diversity exceeds a predefined upper limit, some particles are discarded. In the algorithm, crossover probability of a pair of parents is a function of their age-type (young, middle-aged, old, etc.) and is obtained using a fuzzy rule-base and possibility theory (cf., Appendix A). Inertia weight (w) and probability of mutation (p_m) are initially taken very high and gradually decreases to a certain limit with iteration using a particular law [cf., Section 4.1 (h)]. Due to these improvements, this algorithm [called modified PSO i.e., modified particle swarm optimisation (MPSO)] has more global search ability at the beginning of the run and has more local search ability near the end of the run. Performance of this algorithm is tested against a set of well established benchmark nonlinear test functions (TFs) of which local and global extrema are known. These TFs (cf., Appendix C) and their global/local optima obtained by the proposed MPSO are presented. This algorithm is used to find optimal decision of a production inventory model with stock and price discounted promotional demand in a random planning horizon.

In this paper, a production inventory model of an item is developed, where lifetime of the product is assumed as random in nature and follows normal distribution with known mean and standard deviation. It is also assumed that producer offers a price discount period to his customers in few cycles at the beginning of the planning horizon. During these period, demand increases in each cycle depending on the discount rate. Demand also depends on stock and selling price. After withdrawals of price discount period, demand depends on stock and price for the rest of the cycles. There is only one stochastic constraint – sum of the production cycle length is less than the length of random planning horizon. Models are formulated for both the crisp and fuzzy inventory costs. For crisp inventory parameters, total profit under the above mentioned constraint is maximised using the proposed MPSO to take optimal decisions. When some inventory parameters are fuzzy in nature, total profit through out the system is fuzzy in nature too. In this case, optimal decisions are made using the proposed MPSO with fuzzy objective function. Again, the model is also solved by another soft computing technique i.e., genetic algorithm (GA) (proposed by Guchhait et al., 2010). Obtained results with these two techniques are compared and presented. The models are illustrated with numerical examples and some sensitivity analyses have been made.

2 Literature review

The PSO algorithm has been introduced by Kennedy and Eberhart (1995) and is inspired by the emergent motion of a flock of birds searching for food. It is not only a recently invented high performance optimiser that is easy to understand and implement but, it also requires little computational bookkeeping and generally only a few lines of code (Boeringer and Werner, 2004). Since its introduction, PSO has seen many improvements and applications. Like other heuristics, most of the improvements on PSO are directed towards improving the convergence and increasing the diversity of the swarm. A brief discussion on its improvement in different directions and applications are given in Engelbrecht (2005). Some researchers tried to improve the performance of PSO by variable parameters (Shi and Eberhart, 1998; Zheng et al., 2003; Ratnaweera et al., 2004), some other change the updating equation (Ueno et al., 2005), or adapted operators of the GA (Michalewicz, 1992) or evolutionary strategy (ES) such as crossover, mutation, and sharing (Shi et al., 2005). Similar to other heuristics, PSO also has the problem of converging to undesired local optima because of the diversity of swarm decreasing in latter evolutionary periods and excessive combination with other evolutionary methods may weaken the powerful advantages of PSO. Chen and Zhao (2009) proposed a PSO that uses an adaptive variable swarm size method. In their work, the setting generation is divided into several equal periods; each period is called a ‘ladder’ (a fixed number of generations). This heuristic algorithm is developed with periodic partial increasing or declining of individuals according to diversities in the end of every ladder. Also, in each ladder, the current swarm maintains the same size and the diversity are estimated only at the end of this ladder. If the current diversity is larger than the threshold, the swarm size will be decreased and particles with small score in the ladder will be removed, if the current diversity values is smaller than the threshold, some new individuals produced by the crossover operator will appended to the swarm, otherwise, the swarm size is maintained in current level. So, the swarm size of the ladder is determined by the diversity of the swarm in the terminal of their front ladder (previous ladder).

Now-a-days, it is found (in the subcontinent countries) that some manufacturers offer price discount in the form of putting additional unit(s) in every pack of that item. This process of boosting a product is commonly practiced by the manufacturer especially when a product is newly launched in the market. Pal et al. (2009) developed an EPQ model incorporating price discounted promotional demand in a fuzzy planning horizon. They assume that producer offers price discount in every production cycle for some time, but ignore the effect of product availability in stimulating demand, though this effect is well established (Chung et al., 2000; Maiti and Maiti, 2006). Liang and Zhou (2011) published a two-warehouse inventory model for deteriorating items under conditionally permissible delay in payment. Recently, Molamohamadi et. al. (2014) presented a review article of inventory models under trade credit which really helps the young researchers in this direction.

Production-inventory/inventory models are normally developed under the common assumption that product lifetime is infinite and models are developed under infinite planning horizon (Abad, 2000; Sarkar et. al., 2011; Guchhait et. al., 2010, 2013; De and Sana, 2013; De et. al., 2014). According to this assumption, product specification remains unchanged for ever. But, in reality it is observed that rapid development of technology leads to rapid change in product specifications with new features, new packets and name. Naturally, lifetime of a product is finite and normally it is imprecise (stochastic or fuzzy) in nature. In the literature, there are number of papers with this assumption of stochastic parameters (Gurnani, 1983; Moon and Yun, 1993; Roy et al., 2007). In fact estimation of stochastic parameters is made on sufficient amount of past data. On the other hand, estimation of fuzzy parameters is done by expert's opinion. So when past data is insufficient (especially for the newly lunched products) one has to depend on fuzzy parameters. But, only few number of production inventory models have been developed incorporating this realistic feature (Taleizadeh et. al., 2013; Guchhait et. al., 2012; Chakraborty et. al., 2013; De and Sana, 2013). Accordingly, the problem is developed with fuzzy parameters.

Production cost of a manufacturing system mainly depends on the cost of raw materials and labour. Normally, raw material costs are imprecise in nature. In the existing literature of production/inventory control problems, labour cost is usually assumed as constant. However, in many realistic situations, because of the firms and employees perform the same task repeatedly; they learn how to perform rapidly. Therefore, processing cost of per unit product decreases to a certain limit in every cycle. At the same time, part of ordering cost may also decreases to a certain limit in every cycle. This phenomenon is known as the ‘learning effect’, in the literature. Although, different types of learning effects have been studied in various areas (Kuo and Yang, 2006), it has rarely been studied in the context of inventory control problems (Pal et al., 2009).

3 Mathematical prerequisite

The following notations and assumptions are used in developing the model.

- *Notations*

T	length of one cycle
K	production rate
Q_i	maximum inventory level in i^{th} cycle
c_h	holding cost per unit in \$

$q_i(t)$	inventory level at any time t in ith cycle
Z	total profit from N cycles
α	probability level for the stochastic constraint on the planning horizon.

- *Assumptions*

1 Inventory system involves only one item.

2 The time horizon H is random in nature and follows normal distribution with known mean m_H and standard deviation σ_H and h is real planning horizon such that $P(H \leq 0)$ is negligible.

3 The time horizon accommodates N full cycles, i.e., $NT \leq h$.

4 Price discount is offered to the customers in the first M cycles.

5 Production cost per unit in ith cycle $c_{pi} = c_r + L_i$, where $\mathrm{c_r}$ is raw material cost and L_i is labour cost. L_i decreases in each cycle due to learning effect and is of the form $L_i = L_0 + L_1 / i^{\beta_1}$, where L_0, L_1 and β_1 are constants so chosen to best fit the labour cost function.

6 Setup cost in ith cycle c_{si}(in \$) is partly constant and partly decreases in each cycle due to learning effect of the employees and is of the form: $c_{si} = c_0 + c_1 / i^{\beta_1}$ where c_0, c_1 and β_1 are constants so chosen to best fit the set-ut cost function.

7 Selling price in ith cycle s_{pi} is a mark-up of production cost c_{pi}. m_1 and m_2 are mark-ups during price discount period and normal period respectively, i.e.,

$$s_{pi} = \begin{cases} m_1 c_{pi} & \text{for} \quad 0 \leq t \leq MT; \quad m_1 > 1, \\ m_2 c_{pi} & \text{for} \quad MT \leq t \leq NT; \quad m_2 > m_1, \end{cases}$$

8 Demand of the item in ith cycle D_i is of the form:

$$D_i(t) = \begin{cases} \left.\begin{array}{ll} \dfrac{A + Bq_i - CR^{(m_2 - m_1)iT}}{s_{pi}{}^{\gamma}} & \text{for} \quad 0 \leq q_i \leq Q_0 \\ \dfrac{A + BQ_0 - CR^{(m_2 - m_1)iT}}{s_{pi}{}^{\gamma}} & \text{for} \quad Q_0 \leq q_i \leq Q_i \end{array}\right\} 0 \leq t \leq MT,\, s_{pi} = m_1 c_{pi} \\ \left.\begin{array}{ll} \dfrac{A + Bq_i - CR^{(m_2 - m_1)MT}}{s_{pi}{}^{\gamma}} & \text{for} \quad 0 \leq q_i \leq Q_0 \\ \dfrac{A + BQ_0 - CR^{(m_2 - m_1)MT}}{s_{pi}{}^{\gamma}} & \text{for} \quad Q_0 \leq q_i \leq Q_i \end{array}\right\} MT \leq t \leq NT,\, s_{pi} = m_2 c_{pi} \end{cases}$$

where A, $B(0 < B < 1)$, C, and $R(0 < R < 1)$ are four parameters so chosen to best fit the demand function. Q_i is the maximum inventory level in ith cycle and displayed inventory has impact on demand up to level Q_0.

9 Here, T, N, M, m_1 and m_2 are decision variables.

In the development of the model, it is assumed that the life time of the product, $\hat{H}$, is random in nature and this is taken as planning horizon of the model. 'N' cycles are completed during the real time horizon 'h', here length of each cycle is 'T' and then $NT <= h$ clearly. At the beginning of ith cycle, i.e., at $t = (i - 1)T$, item is produced at a rate 'K' and inventory is built up at a rate $K - D_i(t)$. When $t = (i - 1)T + t_{0i}$, inventory level reaches Q_0. Production stopped when $t = (i - 1)T + t_{1i}$ and at that time inventory level reaches Q_i. After that, inventory is depleted due to the demand rate $D_i(t)$ and its level reaches Q_0 again at $t = (i - 1)T + t_{2i}$. Finally, inventory vanishes at $t = iT$ and production for next cycle starts. Inventory levels over time are depicted in Figures 1 and 2.

Figure 1 Inventory levels in ith cycles when $Q_0 < Q_i$

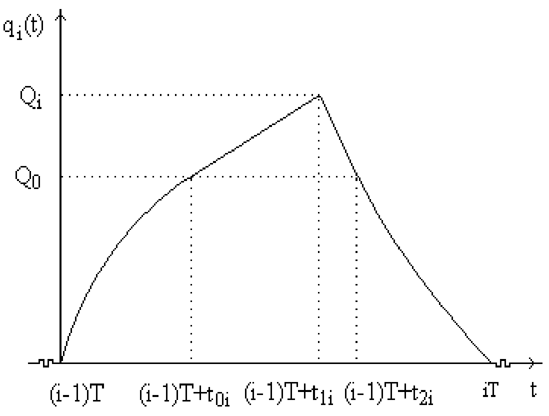
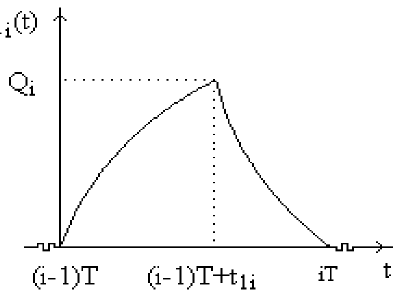


Figure 2 Inventory levels in ith cycle when $Q_0 > Q_i$



3.1 Formulation for first M cycles

Depending upon the values of Q_0 and Q_i two cases may arise.

Case 1 ($Q_0 <= Q_i$)*:* according to the above assumptions, instantaneous state $q_i(t)$ of the item in ith ($i <= M$)cycle ($(i - 1)T <= t <= iT$) is given by

$$\frac{dq_i(t)}{dt}=\begin{cases} K-\dfrac{A+Bq_i-CR^{(m_2-m_1)iT}}{m_1{c_{pi}}^{\gamma}} & (i-1)T\le t\le (i-1)T+t_{0i} \\ K-\dfrac{A+BQ_0-CR^{(m_2-m_1)iT}}{m_1{c_{pi}}^{\gamma}} & (i-1)T+t_{0i}\le t\le (i-1)T+t_{1i} \\ -\dfrac{A+Bq_i-CR^{(m_2-m_1)MT}}{m_1{c_{pi}}^{\gamma}} & (i-1)T+t_{1i}\le t\le (i-1)T+t_{2i} \\ -\dfrac{A+BQ_0-CR^{(m_2-m_1)MT}}{m_1{c_{pi}}^{\gamma}} & (i-1)T+t_{2i}\le t\le iT \end{cases} \tag{1}$$

where $q_i((i-1)T=0$, $q_i((i-1)T+t_{0i})=Q_1$, $q_i((i-1)T+t_{1i}=Q_i$, $q_i((i-1)T+t_{2i}=Q_0$, $q_i(iT)=0$.

Solving the equation (1) and using the boundary conditions we have

$$t_{0i}=\frac{\left(m_1c_{pi}\right)^{\gamma}}{B}\log\left[\frac{K\left(m_1c_{pi}\right)^{\gamma}-A+CR^{(m_2-m_1)iT}}{K\left(m_1c_{pi}\right)^{\gamma}-A-BQ_0+CR^{(m_2-m_1)iT}}\right] \tag{2}$$

$$t_{2i}=T-\frac{\left(m_1c_{pi}\right)^{\gamma}}{B}\log\left[\frac{A+BQ_0-CR^{(m_2-m_1)iT}}{A-CR^{(m_2-m_1)iT}}\right] \tag{3}$$

$$t_{1i}=t_{0i}+\left(t_{2i}-t_{0i}\right)\left[\frac{A+BQ_0-CR^{(m_2-m_1)iT}}{K\left(m_1c_{pi}\right)^{\gamma}}\right] \tag{4}$$

$$Q_i=Q_0+\left(t_{2i}-t_{0i}\right)\left[\frac{A+BQ_0-CR^{(m_2-m_1)iT}}{K\left(m_1c_{pi}\right)^{\gamma}}\right]\left[\frac{K\left(m_1c_{pi}\right)^{\gamma}-\left\{A+BQ_0-CR^{(m_2-m_1)iT}\right.}{K\left(m_1c_{pi}\right)^{\gamma}}\right] \tag{5}$$

So, the holding cost for ith cycle, $c_hHOC_i=c_h\,(H_1+H_2+H_3+H_4)$, where

$$H_1=\int_{(i-1)T}^{(i-1)T+t_{0i}}q_i(t)dt=\int_0^{Q_0}\left[\frac{\left(m_1c_{pi}\right)^{\gamma}q_i(t)dq_i}{K\left(m_1c_{pi}\right)^{\gamma}-\left\{A+Bq_i-CR^{(m_2-m_1)iT}\right\}}\right]$$

$$=\frac{\left(m_1c_{pi}\right)^{\gamma}}{B^2}\left[\left\{K\left(m_1c_{pi}\right)^{\gamma}-A+CR^{(m_2-m_1)iT}\right\}\times\log\left\{\frac{K\left(m_1c_{pi}\right)^{\gamma}-A+CR^{(m_2-m_1)iT}}{K\left(m_1c_{pi}\right)^{\gamma}-\left\{A+Bq_i-CR^{(m_2-m_1)iT}\right\}}\right\}-BQ_0\right] \tag{6}$$

$$H_2 = \int_{(i-1)T+t_{0i}}^{(i-1)T+t_{1i}} q_i(t)dt = \int_{Q_0}^{Q_1} \left[\frac{(m_1c_{pi})^\gamma q_i(t)dq_i}{K(m_1c_{pi})^\gamma - \{A + BQ_0 - CR^{(m_2-m_1)iT}\}} \right]$$
$$= \frac{(Q_i^2 - Q_0^2)(m_1c_{pi})^\gamma}{2\left[K(m_1c_{pi})^\gamma - \{A + BQ_0 - CR^{(m_2-m_1)iT}\}\right]} \tag{7}$$

$$H_3 = \int_{(i-1)T+t_{1i}}^{(i-1)T+t_{2i}} q_i(t)dt = \int_{Q_i}^{Q_0} \frac{-(m_1c_{pi})^\gamma q_i(t)dq_i}{A + BQ_0 - CR^{(m_2-m_1)iT}} = \frac{(Q_i^2 - Q_0^2)(m_1c_{pi})^\gamma}{2\left[A + BQ_0 - CR^{(m_2-m_1)iT}\right]} \tag{8}$$

$$H_4 = \int_{(i-1)T+t_{2i}}^{iT} q_i(t)dt = \int_{Q_0}^{0} \frac{-(m_1c_{pi})^\gamma q_i(t)dq_i}{A + BQ_0 - CR^{(m_2-m_1)iT}}$$
$$= \frac{(m_1c_{pi})^\gamma}{B^2}\left[BQ_0 - \{A - CR^{(m_2-m_1)iT}\} \times \log\left\{\frac{A + BQ_0 + CR^{(m_2-m_1)iT}}{A - CR^{(m_2-m_1)iT}}\right\}\right] \tag{9}$$

Case2 ($Q_0 > Q_i$)*:* in this case, instantaneous state $q_i(t)$ of the item in ith ($i <= M$) cycle ($(i-1)T <= t <= iT$) is given by

$$\frac{dq_i(t)}{dt} = \begin{cases} K - \dfrac{A + Bq_i - CR^{(m_2-m_1)iT}}{m_1c_{pi}^\gamma} & (i-1)T \le t \le (i-1)T + t_{1i} \\ -\dfrac{A + Bq_i - CR^{(m_2-m_1)MT}}{m_1c_{pi}^\gamma} & (i-1)T + t_{1i} \le t \le iT \end{cases} \tag{10}$$

where $q_i((i-1)T) = 0$, $q_i((i-1)T + t_{1i}) = Q_i$, $q_i(iT) = 0$.

Solving the equation (10) and using the above boundary conditions we have

$$Q_i = \frac{\left\{e^{TB/(m_1c_{pi})^\gamma} - 1\right\}\{A - CR^{(m_2-m_1)iT}\}\left\{(m_1c_{pi})^\gamma K - A + CR^{(m_2-m_1)iT}\right\}}{B\left[\left\{(m_1c_{pi})^\gamma K - A + CR^{(m_2-m_1)iT}\right\} + \{A - CR^{(m_2-m_1)iT}\}e^{TB/(m_1c_{pi})^\gamma}\right]} \tag{11}$$

$$t_{1i} = \frac{(m_1c_{pi})^\gamma}{B}\log\left[\frac{(m_1c_{pi})^\gamma K - A + CR^{(m_2-m_1)iT}}{(m_1c_{pi})^\gamma K - A - BQ_i + CR^{(m_2-m_1)iT}}\right] \tag{12}$$

So, the holding cost for ith cycle, $c_hHOC_i = c_h(H_1 + H_2)$, where

$$H_1 = \int_{(i-1)T}^{(i-1)T+t_{1i}} q_i(t)dt \quad \text{and} \quad H_2 = \int_{(i-1)T+t_{1i}}^{iT} q_i(t)dt$$

Can be obtained easily as in Case 1 of Section 3.1

3.2 Formulation for last N-M cycles

Depending on the values of Q_0 and Q_i, again two cases may arises.

Case 1 ($Q_0 <= Q_i$)*:* according to the above assumptions, instantaneous state $q_i(t)$ of the item in ith ($i > M$) cycle ($(i-1)T = t = iT$) is given by

$$\frac{dq_i(t)}{dt} = \begin{cases} K - \dfrac{A + Bq_i - CR^{(m_2-m_1)iT}}{m_2 c_{pi}^{\gamma}} & (i-1)T \le t \le (i-1)T + t_{0i} \\ K - \dfrac{A + BQ_0 - CR^{(m_2-m_1)iT}}{m_2 c_{pi}^{\gamma}} & (i-1)T + t_{0i} \le t \le (i-1)T + t_{1i} \\ -\dfrac{A + BQ_0 - CR^{(m_2-m_1)MT}}{m_2 c_{pi}^{\gamma}} & (i-1)T + t_{1i} \le t \le (i-1)T + t_{2i} \\ -\dfrac{A + BQ_0 - CR^{(m_2-m_1)MT}}{m_2 c_{pi}^{\gamma}} & (i-1)T + t_{2i} \le t \le iT \end{cases} \tag{13}$$

where $q_i((i-1)T = 0$, $q_i((i-1)T + t_{0i}) = Q_0$, $q_i((i-1)T + t_{1i}) = Q_i$, $q_i((i-1)T + t_{2i}) = Q_0$, $q_i(iT) = 0$.

Solving the equation (13) and using the boundary conditions we have,

$$t_{0i} = \frac{\left(m_2 c_{pi}\right)^{\gamma}}{B} \log\left[\frac{K\left(m_2 c_{pi}\right)^{\gamma} - A + CR^{(m_2-m_1)MT}}{K\left(m_2 c_{pi}\right)^{\gamma} - A - BQ_0 + CR^{(m_2-m_1)MT}}\right] \tag{14}$$

$$t_{2i} = T - \frac{\left(m_2 c_{pi}\right)^{\gamma}}{B} \log\left[\frac{A + BQ_0 - CR^{(m_2-m_1)?MT}}{A - CR^{(m_2-m_1)MT}}\right] \tag{15}$$

$$t_{1i} = t_{0i} + \left(t_{2i} - t_{0i}\right)\left[\frac{A + BQ_0 - CR^{(m_2-m_1)MT}}{K\left(m_2 c_{pi}\right)^{\gamma}}\right] \tag{16}$$

$$Q_i = Q_0 + \left(t_{2i} - t_{0i}\right)\left[\frac{A + BQ_0 - CR^{(m_2-m_1)MT}}{K\left(m_2 c_{pi}\right)^{\gamma}}\right] \left[\frac{K(m_2 c_{pi})^{\gamma} - \{A + BQ_0 - CR^{(m_2-m_1)MT}\}}{K(m_2 c_{pi})^{\gamma}}\right] \tag{17}$$

So, the holding cost for ith cycle, $c_h HOC_i = c_h(H_1 + H_2 + H_3 + H_4)$, where

$$H_1 = \int_{(i-1)T}^{(i-1)T+t_{0i}} q_i(t)dt, \quad H_2 = \int_{(i-1)T+t_{0i}}^{(i-1)T+t_{1i}} q_i(t)dt, \quad H_3 = \int_{(i-1)T+t_{1i}}^{(i-1)T+t_{2i}} q_i(t)dt$$

$$\text{and} \quad H_4 = \int_{(i-1)T+t_{2i}}^{iT} q_i(t)dt$$

Can be obtained easily as in Case 1 of Section 3.1

Case 2 ($Q_0 > Q_i$)*:* in this case, instantaneous state $q_i(t)$ of the item in i^{th} ($i > M$) cycle ($(i-1)T <= t< = iT$) is given by

$$\frac{dq_i(t)}{dt} = \begin{cases} K - \dfrac{A + Bq_i - CR^{(m_2-m_1)MT}}{m_2 c_{pi}^{\gamma}} & (i-1)T \le t \le (i-1)T + t_{1i} \\ -\dfrac{A + Bq_i - CR^{(m_2-m_1)MT}}{m_2 c_{pi}^{\gamma}} & (i-1)T + t_{1i} \le t \le iT \end{cases} \tag{18}$$

where $q_i((i-1)T = 0$, $q_i((i-1)T + t_{1i}) = Q_i$, $qi(iT) = 0$.

Solving the equation (18) and using the above boundary conditions we have

$$Q_i = \frac{\left\{e^{TB/(m_2 c_{pi})^{\gamma}} - 1\right\}\left\{A - CR^{(m_2-m_1)MT}\right\}\left\{(m_2 c_{pi})^{\gamma} K - A + CR^{(m_2-m_1)MT}\right\}}{B\left[\left\{(m_2 c_{pi})^{\gamma} K - A + CR^{(m_2-m_1)MT}\right\} + \left\{A - CR^{(m_2-m_1)MT}\right\} e^{TB/(m_1 c_{pi})^{\gamma}}\right]} \tag{19}$$

$$t_{1i} = \frac{(m_2 c_{pi})^{\gamma}}{B} \log\left[\frac{(m_2 c_{pi})^{\gamma} K - A + CR^{(m_2-m_1)MT}}{(m_2 c_{pi})^{\gamma} K - A - BQ_i + CR^{(m_2-m_1)MT}}\right] \tag{20}$$

So, the holding cost for i^{th} cycle, $c_h HOC_i = c_h(H_1 + H_2)$, where

$$H_1 = \int_{(i-1)T}^{(i-1)T+t_{1i}} q_i(t)dt \quad \text{and} \quad H_2 = \int_{(i-1)T+t_{1i}}^{iT} q_i(t)dt$$

Can be obtained easily as in Case 1 of Section 3.1

3.3 *Formulation of objective function*

Incorporating all the cases, the total profit from N cycles can be represented by

$$Z = SP - PC - HOC - OC \tag{21}$$

where

$$SP = \text{total selling price} = \sum_{i=1}^{M} Kt_{1i} m_1 c_{pi} + \sum_{i=M+1}^{N} Kt_{1i} m_2 c_{pi}$$

$$PC = \text{Total production cost} = \sum_{i=1}^{N} Kt_{1i} c_{pi}$$

$$HOC = \text{Total holding cost} = \sum_{i=1}^{N} HOC_i$$

$$OC = \text{Total setup cost} = \sum_{i=1}^{N} c_{si} = \sum_{i=1}^{N}\left\{c_0 + \frac{c_1}{i^{\beta}}\right\}$$

3.4 Model with crisp objective

When all the inventory parameters are crisp, then the problem reduces to determine T, N, M, m_1 and m_2 to

$$\left.\begin{array}{ll}\text{Maximize} & Z \\ \text{subject to} & P\left(NT \le \hat{H}\right) > \alpha\end{array}\right\} \tag{22}$$

Using the optimisation process with stochastic constraints by Charnes and Cooper (1959) (cf., Appendix B), the above problem reduces to determine T, N, M, m_1 and m_2 to

$$\left.\begin{array}{ll}\text{Maximize} & Z \\ \text{subject to} & NT \le m_H - \varepsilon\sigma_H, \quad \text{where} \quad \alpha = \dfrac{1}{\sqrt{2\pi}}\displaystyle\int_{-\infty}^{\varepsilon} e^{-t^2/2}dt\end{array}\right\} \tag{23}$$

For different values of 'α', values of 'ε' are taken from standard normal distribution table and the problem is solved using a modified heuristic technique MPSO (cf., §4).

3.5 Model with fuzzy objective

Here, it is assumed that the holding cost 'c_h', the setup cost coefficients 'c_0' and 'c_1' as triangular fuzzy number (TFN), $\tilde{c}_h = (c_{h1}, c_{h2}, c_{h3})$, $\tilde{c}_0 = (c_{01}, c_{02}, c_{03})$, $\tilde{c}_1 = (c_{11}, c_{12}, c_{13})$ respectively. Then, the total holding cost HOC becomes a TFN, $HOC_F = (H_{F1}, H_{F2}, H_{F3})$ and the total setup cost OC also becomes a TFN, $OC_F = (OC_{F1}, OC_{F2}, OC_{F3})$ where

$$H_{Fj} = c_{hj}\sum_{i=1}^{N} HOC_i \quad \text{and} \quad OC_{Fj} = \sum_{i=1}^{N}\left\{c_{0j} + \frac{c_{1j}}{i^{\beta}}\right\}, \quad j = 1, 2, 3.$$

Hence, the total profit Z also becomes a TFN, $\tilde{Z} = (Z_1, Z_2, Z_3)$ where $Z_1 = SP - PC - OC_{F3} - H_{F3}$, $Z_2 = SP - PC - OC_{F2} - H_{F2}$ and $Z_3 = SP - PC - OC_{F1} - H_{F1}$. So, the problem reduces to

Determine T, N, M, m_1 and m_2 to

$$\left.\begin{array}{ll}\text{Maximize} & \tilde{Z} \\ \text{subject to} & P(NT \le \hat{H}) > \alpha\end{array}\right\} \tag{24}$$

Using the optimisation process with stochastic constraints by Charnes and Cooper (1959) (cf., Appendix B), the above problem reduces to

Determine T, N, M, m_1 and m_2 to

$$\left.\begin{array}{ll}\text{Maximize} & \tilde{Z} \\ \text{subject to} & NT \le m_H - \varepsilon\sigma_H, \quad \text{where} \quad \alpha = \dfrac{1}{\sqrt{2\pi}}\displaystyle\int_{-\infty}^{\varepsilon} e^{-t^2/2}dt\end{array}\right\} \tag{25}$$

This problem is solved using a modified MPSO algorithm (which is discussed below in Section 4) for fuzzy objective.

4 Modified particle swarm optimisation

PSOs are exhaustive search algorithms based on the emergent motion of a flock of birds searching for food (Eberhart and Kennedy, 1995; Kennedy and Eberhart, 1995) and have been extensively used/modified to solve complex decision-making problems in different field of science and technology (Engelbrecht, 2005; Esmin et al., 2002; Feng, 2005; Liang et al., 2006). A PSO normally starts with a set of potential solutions (called swarm) of the decision-making problem under consideration. Individual solutions are called particles and food is analogous to optimal solution. In simple terms the particles are flown through a multi-dimensional search space, where the position of each particle is adjusted according to its own experience and that of its neighbours. Each particle '*i*' has a position vector $X_i(t)$, a velocity vector $V_i(t)$, the position at which the best fitness (*pbesti*) encountered by the particle so far, and the best position of all particles (*gbest*) in current generation t. In generation $(t + 1)$, the position and velocity of the particle are changed to $X_i(t + 1)$ and $V_i(t + 1)$ using following rules:

$$V_i(t+1) = wV_i(t) + c_1 r_1 \left(X_{pbesti}(t) - X_i(t)\right) + c_2 r_2 \left(X_{gbesti}(t) - X_i(t)\right) \tag{26}$$

$$X_i(t+1) = X_i(t) + V_i(t+1) \tag{27}$$

The parameters c_1 and c_2 are set to constant value, which are normally taken as 2, r_1 and r_2 are two random values, uniformly distributed in [0, 1], $w(0 < w < 1)$ is inertia weight which controls the influence of previous velocity on the new velocity. Global search performance is good with large inertial weight while a small inertia weight facilitates a local search. As a result, in the beginning of searching, w is set to high value and decreasing to a lower limit at the end of evolutionary processing. Due to these improvements/changes, PSO have more global search ability at the beginning of the run and have more local search ability near the end of the run. Better performance is derived in many function optimisation problems (Chen and Zhao, 2009). The second part in equation (26) is called cognition character and the third part is social character of particles. The position of each particle is updated every generation. When a particle discovers a pattern that is better than any that it has found previously, it will track the better position by updating equations (26) and (27).

Similar to other heuristics, in PSO, poor parameter settings usually leads to several problems such as premature convergence. PSO also has the problem of converging to undesired local optima because of the diversity of swarm decreasing in latter periodic of evolutionary. Again excessive combination with other evolutionary methods may weaken the powerful advantages of PSO. Overcoming these drawbacks, Chen and Zhao (2009) presented a PSO algorithm that uses an adaptive variable swarm size and periodic partial increasing or declining of particles in the form of ladder function. The aim was to enhance the overall performance of PSO. Their algorithm adjusts the swarm size automatically according to the value of diversity of the swarm in ultimate time of current ladder. For increasing the particles they use simple crossover and mutation operation of GA and for decreasing, particles with lower fitness are removed.

In this paper, the PSO algorithm is modified (in this case it is called as modified PSO i.e., MPSO), where at the time of generation of each particle, a lifetime is assigned depending upon its fitness. Fitness of a particle depends on the objective function value due to it. After completion of a generation (iteration), if no movement is made by the particle, its age is increased by unity. When age exceeds lifetime, particles are discarded from the swarm at the beginning of every iteration. It avoids the local convergence of the algorithm. At the time of generation of initial swarm, diversity in the swarm is maintained using information entropy theory. Every time a new particle X_i is generated, the entropy between this one and previously generated individuals is calculated. If this information quantity is higher than a threshold E_T, fixed at the beginning, X_i is included in the swarm otherwise X_i is again generated until diversity exceeds the threshold E_T. This method induces a good distribution of initial swarm. Diversity in the swarm is also checked after a fixed number of generations called period. After each period, swarm size may change depending on the diversity of the swarm. If diversity level drops threshold level E_T, some new (child) particles are included in the swarm. Child particles are obtained using fuzzy lifetime-based crossover and mutation operations on the swarm. Similarly, if the diversity exceeds predefined upper limit, some particles are discarded. For crossover operation particles are classified into young, middle age and old (in fuzzy sense) according to their age and lifetime. Following comparison of fuzzy numbers using possibility theory (cf., Appendix A), here crossover probability is measured as a function of parent's age interval (a fuzzy rule base on parents age limit is also used for this purpose). In this MPSO, a subset of better child particles are included with the parent swarm for next period and maximum size of this subset is a percentage of the size of its parent set. To control memory overflow at the runtime of the algorithm, an upper limit of swarm size is imposed (Maxsize). Similarly, a lower limit (Minsize) of swarm size is maintained. Algorithm terminates when difference between maximum fitness (Maxfit) and average fitness (Avgfit) of the swarm becomes negligible (less than a predefined small value ε). Here inertia weight (w) and probability of mutation (p_m) are initially very high and follows a decreasing law in function of the generation number. General structure of this MPSO for a maximization problem is presented below. The same algorithm can be used for minimisation problem, by negating the objective value or changing the inequalities of step 22 and step 23 of the algorithm.

In the algorithm B_l and B_r are lower and upper limit vector respectively of variation of search space. Check constraint (X_i) checks whether the solution X_i satisfies the constraints of the problem or not. If X_i satisfies the constraints of the problem, it returns a value 1 otherwise return 0.

Proposed MPSO algorithm

```
1   Initialize period length $T$, $E_T$, $Maxsize$, $Minsize$, $\varepsilon$, $\omega$ and $p_m$
2   Set iteration counter $t = 1$ and randomly generate initial swarm $S(t)$, where diversity in
    the swarm is maintained using entropy originating from information theory.
3   Find lifetime and set velocity $V_i(t)$ of each particles $X_i(t)$ of initial swarm $S(t)$.
4   Do
5       For $j = 1: T$ do
6           For $i = 1: pop\_size$ do
7               $V_i(t+1) = \omega V_i(t) + c_1 r_1(X_{pbesti}(t) - X_i(t)) + c_2 r_2(X_{gbest}(t) - X_i(t))$
8               If ($V_i(t+1) > V_{\max}$) then set $V_i(t+1) = V_{\max}$
9               If ($V_i(t+1) < -V_{\max}$) then set $V_i(t+1) = -V_{\max}$
10              $X_i(t+1) = X_i(t) + V_i(t+1)$
11              If ($X_i(t+1) > B_r$) then set $X_i(t+1) = B_r$
```

```
12                If ($X_i(t + 1) < B_l$) then set $X_i(t + 1) = B_l$
13                If check_constraint ($X_i(t + 1)$) = 0
14                    Increase age of $X_i(t)$ by 1
15                    If age of $X_i(t)$ > life_time
16                        Remove $X_i(t)$ from swarm.
17                    else
18                        Set $X_i(t + 1) = X_i(t)$, $V_i(t + 1) = V_i(t)$
19                    End If
20                Else
21                    Set age of $X_i(t + 1) = 0$
22                    if $f(X_i(t + 1)) > f(X_{pbesti})$ then set $X_{pbesti} = X_i(t + 1)$
23                    if $f(X_i(t + 1)) > f(X_{gbest})$ then set $X_{gbest} = X_i(t + 1)$
24                End If
25            End For
26            Set $t = t + 1$
27        End For
28        Set $Div$ = diversity of $S(t)$.
29        If $Div < E_T$
30            Include child particles in S(t) using fuzzy crossover and mutation such that its size
              does not exceeds Maxsize.
31        End If
32        If $Div > E_T$
33            Remove some particles from $S(t)$ using q-tournament method Chen and Zhao
              (2009) such that its size does not drops Minsize.
34        End If
35        Decrease value of w and $p_m$.
36        Set $Maxfit$ = Maximum fitness in $S(t)$ and $Avgfit$ = Average fitness of $S(t)$.
37    While ($Maxfit - Avgfit \geq \alpha$)
38    Output: Best particle of $S(t)$
39    End Algorithm
```

Algorithm for inclusion of particles into $S(t)$:

```
1     For each pair of particles in $S(t)$ do
2     Determine probability of crossover $\tilde{p}_c$ or the selected pair of parents using fuzzy
      rule-base and possibility theory.
3         Perform crossover with probability $\tilde{p}_c$.
4         If crossover occurs and child particles satisfy the constraints of the problem store
          them into offspring set.
5     End For
6     For each particles $X_i(t)$ in $S(t)$ do
7         Perform mutation with probability $p_m$
```

```
8         If mutation occurs and child particle satisfies the constraints of the problem store
          them into offspring set.
9     End For
10    Rank the particles of offspring set using q-tournament method (Chen and Zhao, 2009).
11    Select a percent of better offsprings from the offspring set and insert into $S(t)$, such that
      maximum size of $S(t)$ does not exceed Maxsize.
12    Remove all offsprings from the offspring set.
13    End Algorithm
```

4.1 Different steps of MPSO algorithm

a *Representation of particles:* a 'n dimensional real vector', $X_i = (x_{i1}, x_{i2},\dots, x_{in})$, is used to represent ith solution, where $x_{i1}, x_{i2},\dots, x_{in}$ represent n decision variables of the decision-making problem under consideration. Xi is called ith chromosome and x_{ij} is called jth gene of ith chromosome.

b *Initialisation:* N solutions $X_i = (x_{i1}, x_{i2},\dots,x_{in})$, $i = 1, 2,\dots,N$ are randomly generated by random number generator within the boundaries of each variable $[B_{j1}, B_{jr}]$, $j = 1, 2,\dots,n$. Initialise (S(1)) sub-function is used for this purpose.

c *Constraint checking:* for constrained optimisation problems, at the time of generation of each individuals X_i of S(1), constraints are checked using a separate sub-function check_constraint(X_i), which returns 1 if X_i satisfies the constraints otherwise returns 0. If check_constraint(x_i) = 1, then X_i is included in S(1) otherwise X_i is again generated and it continues until constraints are satisfied.

d *Diversity preservation:* at the time of generation of S(1) diversity is maintained using entropy originating from information theory. Following steps are used for this purpose.

1 Probability, pr_{jk}, that the value of the ith variable of the jth particle is different from the ith variable of the kth particle is calculated using the formula

$$pr_{jk} = 1 - \frac{x_{ji} - x_{ki}}{B_{ir} - B_{il}}$$

where $[B_{il}, B_{ir}]$ is the variation domain of the ith variable.

2 Entropy of the ith variable, $E_i(M)$, $i = 1, 2,\dots,n$ is calculated using the formula.

$$E_i(M) = \sum_{j=1}^{M-1} \sum_{k=j+1}^{M} -pr_{jk} \log pr_{jk}$$

where M is the size of the swarm.

3 Average entropy of the current swarm is calculated by the formula

$$E(M) = \tfrac{1}{n} \sum_{i=1}^{n} E_i(M)$$

Incorporating the above three steps a separate sub-function check_diversity(X_i) is developed. Every time a new particle X_i is generated, the entropy between this one and previously generated individuals is calculated. If this information quantity is higher than a threshold, E_T, fixed at the beginning, X_i is included in the swarm otherwise X_i is again generated until diversity exceeds the threshold, E_T. This method induces a good distribution of initial swarm.

e *Determination of fitness and lifetime:* fitness $Z(X_i)$ of the particle X_i is determined using the following formula:

$$\begin{aligned} Z(X_i) &= f(X_i) \text{ for maximization problem, and} \\ &= L - f(X_i) \text{for minimizationproblem} \end{aligned} \tag{28}$$

where $f(x)$ is objective function and L is a sufficiently large positive real to make fitness positive. Following Michalewicz (1992), at the time of birth life-time of X_i is computed using the following formula:

$$\text{If Avgfit} \geq Z(X_i), \text{lifetime}(X_i) = \text{Minlt} + \frac{K(Z(X_i) - \text{Minfit})}{\text{Avgfit} - \text{Minfit}}$$

$$\text{If Avgfit} < Z(X_i), \text{lifetime}(X_i) = \frac{\text{Minlt} + \text{Maxlt}}{2} + \frac{K(Z(X_i) - \text{Avgfit})}{\text{Maxfit} - \text{Avgfit}}$$

where Maxlt and Minlt are maximum and minimum allowed lifetime of a particle, $K = (\text{Maxlt} - \text{Minlt})/2$. Maxfit, Avgfit, Minfit represent the best, average and worst fitness of the current population. To optimise different TFs it is assumed that Maxlt = 7 and Minlt = 1, N = 10. According to the age, a particle can belongs to any one of age intervals – young, middle-age or old, whose membership functions are presented in Figure 3.

f Crossover:

- Determination of probability of crossover $(\tilde{p}_c)$: probability of crossover $\tilde{p}_c$, for a pair of parents (X_i, X_j) is determined as below:
  1. At first age intervals (young, middle-age, old) of X_i and X_j are determined by making possibility measure of fuzzy numbers – young, middle-age, old with respect to their age using Lemma 4 (cf., Appendix A). If age of X_i is a_i, then possibility measures of fuzzy numbers young, middle-age and old with respect to a_i are calculated. Then age interval having maximum possibility measure is taken as age interval of X_i. Similarly, age interval of X_j is determined,
  2. After determination of age intervals of the parents their crossover probability $(\tilde{p}_c)$ is determined as a linguistic variable (low, medium or high) using a fuzzy rule-base as presented in Table 1. Membership function of these linguistic variables is presented in Figure 4.

3 *Crossover process:* for each pair of parent solutions X_i, X_j a random number c is generated from the range [0, 1] and if $\text{nes}(c < \tilde{p}_c) > \beta$ (cf., Lemma 1 in Appendix A), crossover operation is made on X_i, X_j, where β $(0 < \beta < 1)$ is a predefined necessity level. For the proposed model it is assumed that $\beta = 0.5$. To made crossover operation on each pair of coupled solutions X_i, X_j a random number c_1 is generated from the range [0, 1] and their offsprings Y_1 and Y_2 are determined by the formula:

$$Y_1 = c_1 X_i + (1 - c_1)\, X_j, \quad Y_2 = c_1\, X_j + (1 - c_1)\, X_i$$

For constrained optimisation problems, if a child solution satisfies the constraints of the problem then it is included in the offspring set otherwise it is not included in the offspring set.

g Mutation:

1 *Selection for mutation:* for each offspring generate a random number 'r' from the range [0, 1].

If $r < p_m$ then the solution is taken for mutation, where p_m is the probability of mutation.

2 *Mutation process:* random mutation process is used for this purpose. According to this process a randomly selected variable of the solution is replaced by its regenerated value with its search boundary. So to mutate a solution $X = (x_1, x_2, \ldots, x_n)$ a random integer i in the range $[1, n]$ has to be selected. Then replace x_i by randomly generated value within the boundary $[B_{il}, B_{ir}]$ of i^{th} variable of X. New solution (if satisfies constraints of the problem) included in the offspring set. Constraint checking of a child solution C_i is made using check_constraint(C_i) function.

h *Reduction process of ω and p_m:* let $\omega(0)$ and $p_m(0)$ is the initial value of ω and p_m respectively. Then the value of w and p_m in T^{th} generation $w(T)$ and $p_m(T)$ are calculated by the following formulae:

$\omega(0) = \omega\ (0)\exp(-T/\alpha_1)$ and $p_m(T) = p_m(0)\ \exp(-T/\alpha_2)$, where α_1 is calculated so that the final value of ω is small (0.2 in our case). So

$$\alpha_1 = \text{Maxgen}/\log\left[\frac{\omega(0)}{0.2}\right],$$

where Maxgen is the expected number of generations that the GA can run for convergence. Similarly, α_2 is calculated so that the final value of p_m is small enough (10^{-2} in our case). So

$$\alpha_1 = \text{Maxgen}/\log\left[\frac{p_m(0)}{10^{-2}}\right]$$

i Inclusion of particles in $S(t)$:

- Following Chen and Zhao (2009) maximum number of offsprings to be included in $S(t)$ is $N_I = N(t) \times \lambda \times Div(S(t))$, where $N(t)$ represents the current size of $S(t)$, $Div(S(t))$ is the diversity of $S(t)$ and λ is adding factor. The common value of λ belongs to the interval [0.2, 0.3].

- Particles in the offspring set are at first ranked using q-tournament process (Chen and Zhao, 2009).
- Particles in the offspring set are arranged in descending order of their rank.
- First N_I particles are included in $S(t)$. After inclusion, excess particles are removed if the number of particles in $S(t)$ exceeds Maxsize.

(j) Deletion of particles from $S(t)$:

- Again, following Chen and Zhao (2009), maximum number of offsprings to be deleted from $S(t)$ is $N_D = N(t) \times \eta \times Div(S(t))$, where $N(t)$ is the current size of $S(t)$, $Div(S(t))$ is the diversity of $S(t)$ and η is deletion factor, whose value is less than λ. The common value belongs to the interval [0.1, 0.2].
- Particles in $S(t)$ are at first ranked using the q-tournament process (Chen and Zhao, 2009).
- $S(t)$ is arranged in descending order of rank of particles.
- Last min$\{N_D, O(\mathrm{S(t)})-\mathrm{Minsize}\}$ particles are deleted from $S(t)$, so that after deletion, the number of particles in $S(t)$ does not drops below Minsize.

k *Termination condition:* algorithm terminates when difference between maximum fitness (Maxfit) of particle, i.e., fitness of the best solution of the swarm and average fitness (Avgfit) of the swarm becomes negligible.

l *Implementation:* with the above function and values the algorithm is implemented using C-programming language in a personal computer consist of Intel 3.07 GHZ processor, 448 MB RAM and Microsoft Windows XP operating system.

m *Convergence of the MPSO algorithm:* a set of TFs are used to test the efficiency of the proposed algorithm MPSO. It is observed that results obtained by MPSO for different TFs are equal to the global optimum in most of the cases and in other cases solutions are all most near to global optima. This implies that this algorithm can be used as decision-making tool for different constrained/unconstrained optimisation problems.

Figure 3 Membership functions of age intervals

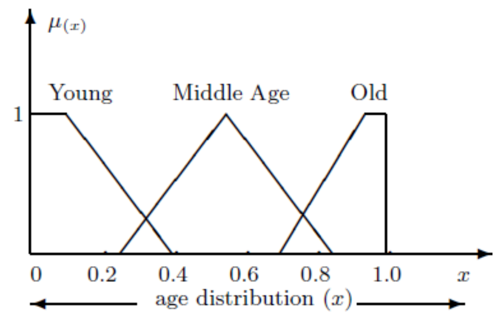


Figure 4 Membership functions of crossover probabilities

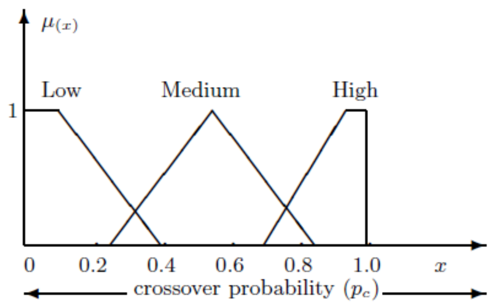


Table 1 Fuzzy rule-base for crossover probability

Parent 2	*Parent 1*		
	Young	*Middle-age*	*Old*
Young	Low	Medium	Low
Middle-age	Medium	High	Medium
Old	Low	Medium	Low

4.2 MPSO for fuzzy objective function

A general single-objective mathematical programming problem with fuzzy objective function can be written as:

$$\text{Maximize} \tilde{f}(X, \xi), \quad \text{subject to } g_i(\text{X}) \leq 0, \quad i = 1, 2, \ldots, m, \tag{29}$$

where $X = (x_1, x_2, \ldots, x_n)$ is a decision vector, $\tilde{\xi}$ is a vector of fuzzy parameters, $\tilde{f}(X, \tilde{\xi})$ is the return function and $g_i(X)$ are constraints.

The above MPSO algorithm can be used to determine optimal decision of the above problem. To deal with such a problem at first a fuzzy goal $(\tilde{F}_g)$ of the objective function $(\tilde{f})$ is determined using the above MPSO algorithm. Then to make decision of the fuzzy optimisation problem, possibility/necessity (optimistic/pessimistic sense) (cf., Appendix A) of the fuzzy objective $(\tilde{f})$ with respect to the fuzzy goal $(\tilde{F}_g)$ is taken as fitness of a particle (solution) of the swarm. The process is discussed below. Other functions and values are same as discussed above.

- Determination of fuzzy goal: fuzzy goal $\tilde{F}_g$ □ of the fuzzy objective function $\tilde{f}(X, \tilde{\xi})$ is considered as a linear fuzzy number (LFN) (F_{g1}, F_{g2}) and values of F_{g1}, F_{g2} can be determined in different ways. Here the following formulae are proposed and are used for numerical illustrations for the fuzzy models. In the formula S denotes the feasible search space of the problem.

$$F_{g_1} = \inf_{x_0 \in S} \inf_{\xi_0 \in \tilde{\xi}} \tilde{f}\left(x_0, \xi_0\right), \quad F_{g_2} = \sup_{x_0 \in S} spu_{\xi_0 \in \tilde{\xi}} \tilde{f}\left(x_0, \xi_0\right)$$

- Fitness evaluation: for optimistic decision maker (DM),

$$\prod_{\tilde{f}(X,\tilde{\xi})}(\tilde{F}_g)$$

can be taken as fitness of a solution X and for pessimistic DM, $N_{\tilde{f}(X,\tilde{\xi})}(\tilde{F}_g)$ can be taken as fitness. If analytical form of membership function of $\tilde{f}(X,\tilde{\xi})$ (obtained by the formula defined in equation (34) is a TFN $(F_1(X), F_2(X), F_3(X))$, then Lemma 2 gives

$$\prod_{\tilde{f}(X,\tilde{\xi})}(\tilde{F}_g) = U_1/E_1,$$

where $U_1 = F_3(X) - F_{g1}$ and $E_1 = F_3(X) - F_2(X) + F_{g2} - F_{g1}$. As MPSO search plays a role to maximise the fitness of the chromosomes in the swarm, so maximisation of

$$\prod_{\tilde{f}(X,\tilde{\xi})}(\tilde{F}_g)$$

implies maximisation of $F_2(X)$ (most feasible equivalent of $\tilde{f}(X,\tilde{\xi})$ and F$_3$(X) (least feasible equivalent of $\tilde{f}(X,\tilde{\xi})$ together. Again,

$$\prod_{\tilde{f}(X,\tilde{\xi})}(\tilde{F}_g) = 1$$

implies $F_2(x) \geq F_{g2}$, i.e., most feasible objective value achieves the highest level of profit goal (F_{g2}). Thus if DM is optimistic and allows some risk, he/she will find the fitness of a solution depending on the possibility measure. On the other hand, Lemma 3gives $N_{\tilde{f}(X,\tilde{\xi})}(\tilde{F}_g) = U_2 / E_2$, where $U_2 = F_2(X) - F_{g1}$ and $E_2 = F_2(X) - F_1(X) + F_{g2} - F_{g1}$. So in this case maximisation of $N_{\tilde{f}(X,\tilde{\xi})}(\tilde{F}_g)$ implies maximisation of $F_1(X)$ (worst possible equivalent of $\tilde{f}(X,\tilde{\xi})$ and $F_2(X)$ most feasible equivalent of $\tilde{f}(X,\tilde{\xi})$ together. $N_{\tilde{f}(X,\tilde{\xi})}(\tilde{F}_g) = 1$ implies $F_1(X) \geq F_{g2}$, i.e., worst possible objective value achieves the highest level of profit goal (F_{g2}). If DM is pessimistic and does not allow any risk, he/she will determine the fitness of solution using necessity measure approach. Set $Z(X)$ as fitness of X.

5 Numerical illustration

5.1 Testing of algorithm

Here, different models are solved using the proposed MPSO algorithm. Different parameter values of the MPSO algorithm used for this purpose are given below:

Number of variables in a solution $n = 5$. A five dimensional real vector $X = (x_1, x_2, x_3, x_4, x_5)$ is used to represent a solution. In which x_1 represents T, x_2 represents N, x_3 represents M, x_4 represents m_1 and x_5 represents m_2. Integral parts of x_2 and x_3 are taken as values of N and M respectively. Minimum swarm size Minsize = 10, Maximum swarm size Maxsize = 100. Initial value of probability of mutation $p_m(0) = 0.9$ and inertia weight w(0) = 0.9. Lower limit of inertia weight=0.2. Value of 'q' in q-tournament method is 5. Expected value of maximum number of generation to converge the MPSO, Maxgen =

200. MPSO algorithm is run for the TFs (cf., Appendix C) using different seeds of random number generators for 40 times each and number of success of finding optimal solutions for each function are noted. If optimal solution is found in a run of the heuristic algorithm using a seed, we say that the run is successful. % of successful runs for different TFs due to that technique is listed in Table 2. It is observed that the performance of MPSO is good enough.

Table 2 Results of TFs following the proposed MPSO algorithm

TF	*Results obtained*	*Maximum iterations required for convergence*	*% of success for 40 runs*
TF1	ES(3.1432, 3.1438) = −0.9998	164	96%
TF2	MZ(2.2025, 1.5744) = −1.8004	180	92%
TF3	F(1.0001, 0.9998) = 1.0002	159	98%
TF4	F2(2.0481, 0.0001) = −205.8479	176	98%
TF5	SH(x_1) = −186.7306	176	94%
TF6	RC(x_2) = 0.39773	98	96%
TF7	F_2(1.0013, 0.9998) = 0.0	178	94%
TF7	F_4(0.9996, 0.9998, 0.9997, 0.9996) = 0.0	Does not converge for all runs	90%
TF8	BH(0.0000, 0.0001) = 0.0	125	98%
TF9	F(0.5, 0.2499) = 0.25	102	98%
TF10	Z_3(0.0000, 0.0001, 0.0001) = 0.0	144	98%

Notes: For TF-5, 18 global minima are x_1 = [(–0.8004,–1.4262), (–1.4252, –0.8003), (4.8584, –7.0835), (–7.0836, 4.8579), (–7.7083, –7.0835), (–7.0839, –7.7083), (5.4826, 4.8582), (4.8581, 5.4828), (–1.4254, –7.0839), (–7.0835, –1.4253), (4.8583, –0.8001), (–0.8005, 4.8583) (5.4827, –7.7082), (–7.7086, 5.4830), (–7.7085, –0.8001), (–0.8001, –7.7086), (5.4821, –1.4250), (–1.4252, 5.4823)]. For TF–6, three global minima are x_2 = [(–3.1423, 12.2504), (3.1431, 2.2517), (9.4253, 2.2474)].

Figure 5 $p_m(0)$ vs. % of success (see online version for colours)

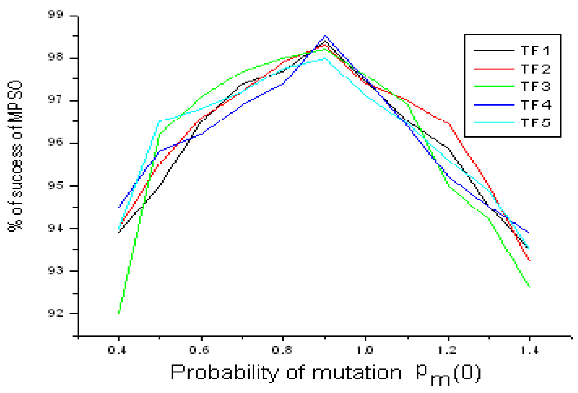
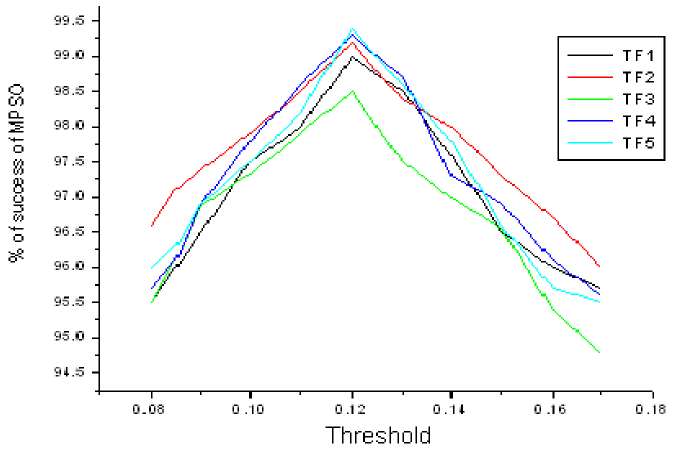


Figure 6 Threshold vs. % of success (see online version for colours)



Performance of the proposed MPSO algorithm is improved by setting the appropriate values of the parameters of the algorithm against the above mentioned TFs (Here, TF-1 to TF-5 are taken). For different values of initial probability of mutation $p_m(0)$ and threshold E_T, performance of the proposed heuristic algorithm for these TFs are evaluated and plotted graphically (cf., Figure 5 and Figure 6). Figure 5 and Figure 6 reflected that the % of success of the MPSO is good enough for $p_m(0) = 0.9$ and $E_T = 0.12$ taking reasonable number of iterations for convergence. This study is made to fix-up the values of $p_m(0)$ and E_T for which optimum results can be found in most of the runs of the algorithm and numerical experiments of the proposed models are made with these values of $p_m(0)$ and E_T.

5.2 *Numerical illustration of the proposed model*

5.2.1 *Results of the model with crisp objective function*

The following parameter values are used to illustrate the model:

$$K = 2{,}500,\ Q_0 = 1{,}000,\ A = 2{,}000,\ B = 0.25,\ C = 500,\ R = 0.7,\ c_r = 1.5,$$
$$L_0 = 0.5,\ L_1 = 1.25,\ \alpha = 0.90,\ \beta_1 = 0.4,\ c_0 = 30,\ c_1 = 35,\ \beta_2 = 0.5,\ \gamma = 2.5,$$
$$m_H = 20,\ \sigma_H = 1 \text{ and upper limit of } m_2 \text{ is } 2.$$

As discussed earlier, to compare the results, the present crisp model (Section 3.4) is solved following the two algorithms, i.e., MPSO and GA (presented by Guchhait et al., 2010). For the above parametric values, the optimum cycle length (T), the total number of cycles (N), the number of cycles for which price discount is offered (M), the mark-ups of selling price (m_1 and m_2) and the profit (Z) are obtained by both the methods and presented. Obtained results for different values of 'α' (degree of probability of satisfying the constraint on random planning horizon) are presented in Table 3.

Table 3 Results of crisp model obtained by MPSO and GA

MPSO							GA						
α	T	N	M	m_1	m_2	Z	α	T	N	M	m_1	m_2	Z
0.90	1.8715	10	6	1.6509	2	618.10	0.90	1.8675	10	6	1.6561	2	617.64
0.91	1.8655	10	6	1.6516	2	615.23	0.91	1.8611	10	6	1.6546	2	614.27
0.92	1.8595	10	6	1.6575	2	612.41	0.92	1.8567	10	6	1.6642	2	611.55
0.93	1.8525	10	7	1.6807	2	609.25	0.93	1.8541	10	7	1.6723	2	608.26
0.94	1.8445	10	7	1.6808	2	605.34	0.94	1.8421	10	7	1.6733	2	604.68

As expected, profit decreases with increase of 'α'. In fact, increase of 'α' decreases the effective length of planning horizon, which in turn decreases the profit. Also, it is observed that m_1 increases with 'α' to keep the profit high. In fact, increase of m_1 decreases the demand which decreases the profit, but m_1 directly increases the profit. As a result, though increase of m_1 decreases demand, increase in profit due to increase in m_1 dominated decrease in profit due to decrease in demand. So, the resultant effect keeps the total profit high. Again, from this table it is observed that after the certain increasing value of 'α', the number of cycles for price discount is more than the previous one, which in turn decreases the profit also. Similar trends of the results are observed for both the algorithms, but, the profit obtained by MPSO is better than that of GA. Of course, these better results do not contribute much in real sense for the present set of parametric values of the model. The differences are in the decimal places. As our problem is to maximise the profit, so we can say that the proposed algorithm performed better compared to the GA.

Table 4 Results of crisp model obtained by MPSO and GA

MPSO							GA						
R	T	N	M	m_1	m_2	Z	R	T	N	M	m_1	m_2	Z
0.71	1.8715	10	6	1.6580	2	613.99	0.71	1.8717	10	6	1.6572	2	613.03
0.72	1.8715	10	6	1.6571	2	609.99	0.72	1.8716	10	6	1.6567	2	609.11
0.73	1.8715	10	7	1.6806	2	605.72	0.73	1.8718	10	7	1.6630	2	604.87
0.74	1.8715	10	7	1.6799	2	601.38	0.74	1.8716	10	7	1.6753	2	600.22
0.75	1.8715	10	7	1.6796	2	596.75	0.75	1.8717	10	7	1.6649	2	595.02

Again, for the above parameter values, results are also obtained by both the algorithms MPSO and GA for different values of 'R' and presented in Table 4. It is observed that the profit decreases with 'R'. It happens because increase of 'R' decreases the effect of price discount on demand, i.e., demand decreases with the increase of 'R'. As demand of the item decreases; the profit decreases automatically. It is also observed from the Table 4 that, to keep the demand high (with the increase of 'R'), either m_1 decreases or M increases. Because, increasing demand reflects more profit. As the proposed model is to maximise the profit so, all these observations agree with reality. Here also, it is observed that the profit obtained by MPSO is more than that of GA.

5.2.2 Results of the model with fuzzy objective function

To illustrate the fuzzy model, the setup cost coefficients $\tilde{c}_0, \tilde{c}_1$ and the holding cost $\tilde{c}_h$ are considered as TFNs, i.e., $\tilde{c}_0 = (30, 35, 40)$, $\tilde{c}_1 = (35, 40, 45)$ and $\tilde{c}_h = (0.45, 0.5, 0.55)$ respectively. Other parameter values are same as in the crisp model. From the previous results (i.e., the results of crisp model as well as TFs) it can be concluded that the MPSO algorithm performed better compared to the other. For this reason the fuzzy model is solved using only MPSO algorithm. For the assumed parameter values, calculated value of F_{g1} = 447 and F_{g2} = 650.16. The results are obtained by MPSO algorithm for the fuzzy model (25) using both the measure of fuzzy number (i.e., possibility and necessity measures) and presented in Tables 5 and 6 respectively. It is observed from these tables (Tables 5 and 6) that maximum possible profit due to possibility measure (Z_3 = 649.25) is more than that of the maximum profit due to necessity measure (Z_3 = 647.65). But, there is some risk in possibility measure approach. Because, in that case minimum assured profit (Z_1 = 444.01) is less than that of the obtained following necessity measure (Z_1 = 445.29). It happens because possibility measure approach is followed by optimistic DMs. Using this approach actually Z_2 and Z_3 are optimised [cf., §4.1(2)]. On the other hand necessity measure approach is followed by pessimistic DMs. Using this approach, actually Z_1 and Z_2 are optimised [cf., §4.1(2)]. As a result, Z_3 in necessity measure approach may be less compared to the value of Z_3 in possibility measure approach. In this case, minimum possible profit (i.e., Z_1 in necessity measure) is optimised, so DM is free from risk.

Table 5 Results of fuzzy model following possibility measure for fitness

Algorithm	*T*	*N*	*M*	m_1	m_2	Z_1	Z_2	Z_3
MPSO	2.0794	9	6	1.6774	2	444.01	546.63	649.25

Table 6 Results of fuzzy model following necessity measure for fitness

Algorithm	*T*	*N*	*M*	m_1	m_2	Z_1	Z_2	Z_3
MPSO	2.0794	9	5	1.6627	2	445.29	546.47	647.65

6 Managerial implications and insights

In the present investigation, several cases of the models with the credit period are formulated with respect to retailer and presented. Manager of a retail shop can take the managerial decisions depending upon the actual prevailing situation which fits best with the given models for maximum profit.

1 This model suggests to a manager of a sector how to determine the number of cycles for which price discount is offered so that the profit can be maximised.

2 Results of the models with crisp and imprecise parameters are presented. Manager of a retail shop can choose their appropriate model depending upon the nature of parameters.

7 Conclusions

In this paper, a fuzzy lifetime-based MPSO algorithm is proposed which is efficient in solving constrained optimisation problem with crisp as well as fuzzy objective. The main features of the developed algorithm are summarised below:

- at the time of birth of a swarm, diversity is maintained using information entropy theory
- initial value of probability of mutation $p_m(0)$ and inertia weight $w(0)$ are taken very high and gradually decreases (towards a minimum limiting value) with increase of iteration counter
- crossover operations are made on the particles whose movements are not made using normal PSO functions
- moreover, if no movement of a particle is made during a finite number of generation (i.e., if age exceeds the lifetime), it is discarded from the swarm.

These realistic features make the algorithm more powerful in searching the global optima (if it exists). Performance of the MPSO algorithm (tested against a list of TFs) concludes that it is good enough. As the developed algorithm is more efficient, so that it can be used to solve different decision-making problems in different fields of science and technology. Here also, a production inventory model is developed incorporating the effect of stock and price on demand when price discount is offered to the customers for few cycles. Moreover, learning effects on production cost and setup cost are incorporated in this EPQ model. The aim of adaptation of the model is fourfold.

- study the effect of both the stock and price on demand when price discount is offered to the customers
- incorporate employees learning effects on production and set-up cost in a finite rate production model
- introduce lifetime of a product as random in nature in a EPQ model
- propose a MPSO algorithm which can deals with crisp as well as fuzzy objective function

Here, in the fuzzy model, production cost coefficients are not considered as fuzzy. It is due to the fact that if production cost is fuzzy then demand becomes fuzzy. If demand is fuzzy then cycle length will vary from cycle to cycle, as well as due to the impreciseness of demand, production quantities for different cycles may finish before the end of some cycles as well as may excess at the end of other cycles. So it is difficult to find optimal decision in that case following this approach. On the other hand, there are also some limitations in this algorithm which are given below.

- solutions obtained in this approach are normally near optimal, not exact
- near optimal solutions may not occur in some runs of the algorithm.

Further research work can be made on the optimisation problem with other environments like – rough, fuzzy-rough, fuzzy-random etc. Also, this model can be extended to a 'multi-item inventory model' by considering more than one item in the system. Again due to generalised solution methodology, this approach can be applied to solve other inventory control problems along with the problems in other disciplines.

References

Abad, P.L. (2000) 'Optimal lot-size for a perishable good under conditions of finite production and partial backordering and lost sale', *Computers & Industrial Engineering*, Vol. 38, No. 4, pp.457–465.

Bessaou, M. and Siarry, P. (2001) 'A genetic algorithm with real-value coding to optimize multimodal continuous function', *Structural and Multi-Disciplinary Optimization*, Vol. 23, No. 1, pp.63–74.

Boeringer, D.W. and Werner, D.H. (2004) 'Particle swarm optimization versus genetic algorithms for phased array synthesis', *IEEE Transactions on Antennas and Propagation*, Vol. 52, No. 3, pp.771–779.

Chakraborty, S., Pal, M. and Nayak, P.K. (2013) 'Intuitionistic fuzzy optimization technique for Pareto optimal solution of manufacturing inventory models with shortages', *Eur. J. Oper. Res.*, Vol. 228, No. 2, pp.381–387.

Charnes, A. and Cooper, W.W. (1959) 'Chance constrained programming', *Management Science*, Vol. 6, No. 1, pp.73–79.

Chen, D. and Zhao, C. (2009) 'Particle swarm optimization with adaptive population size and its application', *Applied Soft Computing*, Vol. 9, No. 1, pp.39–48.

Chung, K.J., Chu, P. and Lan, S.P. (2000) 'A note on EOQ models for deteriorating items under stock dependent selling rate', *European Journal of Operational Research*, Vol. 124, No. 3, pp.550–559.

De, S.K. and Sana, S.S. (2013) 'Fuzzy order quantity inventory model with fuzzy shortage quantity and fuzzy promotional index', *Economic Modelling*, Vol. 31, pp.351–358.

De, S.K., Goswami, A. and Sana, S.S. (2014) 'An interpolating by pass to Pareto optimality in intuitionistic fuzzy technique for a EOQ model with time sensitive backlogging', *Applied Mathematics and Computation*, Vol. 230, No. 1, pp.664–674.

Dubois, D. and Prade, H. (1980) *Fuzzy Sets and System – Theory and Application*, Academic, New York.

Eberhart, R.C. and Kennedy, J. (1995) 'A new optimizer using particle swarm theory', *Proceedings of the Sixth International Symposium on Micromachine and Human Science*, pp.39–43.

Engelbrecht, A.P. (2005) *Fundamentals of Computational Swarm Intelligence*, 1st ed., John Wiley and Sons, Ltd., November.

Esmin, A.A.A., Aoki, A. and Lambert-Torres, R.G. (2002) 'Particle swarm optimization for fuzzy membership functions optimization', *IEEE International Conference on System Man Cybernatics*, 6–9 October, Vol. 3, pp.6–9.

Feng, H.M. (2005) 'Particle swarm optimization learning fuzzy systems design', *Proceedings of the ICITA, 3rd International Conference on Information Technology and Applications*, July, Vol. 1, No. 47, pp.363–366.

Guchhait, P., Maiti, M.K. and Maiti, M. (2012) 'Imperfect production policy of a breakable item with variable breakability and demand in random planning horizon', *Int. J. Mathematics in Operational Research*, Vol. 4, No. 6, pp.622–637.

Guchhait, P., Maiti, M.K. and Maiti, M. (2013) 'Two storage inventory model of a deteriorating item with variable demand under partial credit period', *Applied Soft Computing*, Vol. 13, No. 1, pp.428–448.

Guchhait, P., Maiti, M.K. and Maiti, M., (2010) 'Multi-item inventory model of breakable items with stock-dependent demand under stock and time dependent breakability rate', *Computers & Industrial Engineering*, Vol. 59, No. 4, pp.911–920.

Gurnani, C. (1983) 'Economic analysis of inventory systems', *International Journal of Production Research*, Vol. 21, No. 2, pp.261–277.

Kennedy, J. and Eberhart, R.C. (1995) 'Particle swarm optimisation', *Proceedings of the IEEE International Joint Conference on Neural Network*, IEEE Press, Vol. 4, pp.1942–1948.

Kuo, W.H. and Yang, D.L. (2006) 'Minimizing the total completion time in a single-machine scheduling problem with a time-dependent learning effect', *European Journal of Operational Research*, Vol. 174, No. 2, pp.1184–1190.

Liang, J.J., Qin, A.K., Suganthan, P.N. and Baskar, S. (2006) 'Comprehensive learning particle swarm optimizer for global optimization of multimodal functions', *IEEE Transactions on Evolutionary Computation*, Vol. 10, No. 3, pp.281–295.

Liang, Y. and Zhou, F. (2011) 'A two-warehouse inventory model for deteriorating items under conditionally permissible delay in payment', *Applied Mathematical Modelling*, Vol. 35, No. 5, pp.2221–2231.

Liu, B. and Iwamura, K. (1998) 'Chance constraint programming with fuzzy parameters', *Fuzzy Sets and Systems*, Vol. 94, No. 2, pp.227–237.

Maiti, M.K. and Maiti, M. (2006) 'Fuzzy inventory model with two warehouses under possibility constraints', *Fuzzy Sets and Systems*, Vol. 157, No. 1, pp.52–73.

Michalewicz, Z. (1992) *Genetic Algorithms + Data Structures = Evolution Programs*, Springer, Berlin.

Molamohamadi, Z., Ismail, N., Leman, Z. and Zulkifli, N. (2014) 'Reviewing the literature of inventory models under trade credit contact', *Discrete Dynamics in Nature and Society*, 2014, Article ID 975425, 19pp. [online] http://dx.doi.org/10.1155/2014/975425.

Moon, I. and Yun, W. (1993) 'An economic order quantity model with random planning horizon', *The Engineering Economist*, Vol. 39, No. 1, pp.77–86.

Pal, S., Maiti, M.K. and Maiti, M. (2009) 'An EPQ model with price discounted promotional demand in an imprecise planning horizon via genetic algorithm', *Computers & Industrial Engineering*, Vol. 57, No. 1, pp.181–187.

Ratnaweera, A. Halgamuge, S.K. and Watson, H.C. (2004) 'Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients', *IEEE Transactions on Evolutionary Computation*, Vol. 8, No. 3, pp.240–255.

Roy, A., Maiti, M.K., Kar, S. and Maiti, M. (2007) 'Two storage inventory model with fuzzy deterioration over a random planning horizon', *Mathematical and Computer Modeling*, Vol. 46, Nos. 11–12, pp.1419–1433.

Sarkar, B., Sana, S.S. and Chaudhuri, K.S. (2011) 'An imperfect production process for time varying demand with inflation and time value of money – an EMQ model', *Expert System with Applications*, Vol. 38, No. 11, pp.13543–13548.

Shi, X.H., Liang, Y.C., Lee, H.P., Lu, C. and Wang, L.M. (2005) 'An improved GA and a novel PSOGA based hybrid algorithm[J]', *Inform. Process. Lett.*, Vol. 93, No. 5, pp.255–261.

Shi, Y. and Eberhart, R.C. (1998) 'A modified swarm optimizer', *Proceedings of IEEE International Conference on Evolutionary Computation*, IEEE press, Piscataway, NJ, pp.69–73.

Taleizadeh, A.A., Niaki, S.T.A., Aryanezhad, M.B. and Nima Shafii. (2013) 'A hybrid method of fuzzy simulation and genetic algorithm to optimize constrained inventory control systems with stochastic replenishments and fuzzy demand', *Inf. Sci.*, Vol. 220, pp.425–441.

Ueno, G., Yasuda, K. and Iwasaki, N. (2005) 'Robust adaptive particle swarm optimization', *IEEE International Conference on System Man and Cybernetics*, Vol. 4, No. 1012, pp.3915–3920.

Zadeh, L.A. (1978) 'Fuzzy sets as a basis for a theory of possibility', *Fuzzy Sets and Systems*, Vol. 1, pp.3–28.

Zheng, Y.L., Ma, L.H. and Zhang, L.Y. (2003) 'On the convergence analysis and parameter selection in particle swarm optimization', *Proceedings of 2nd International Conference on Machine Learning and Cybernetics*, Xian, 25 October, Vol. 10, pp.1802–1807.

Appendix A

Let $\tilde{a}$ and $\tilde{b}$ be two fuzzy numbers with membership functions $\mu_{\tilde{a}}(x)$ and $\mu_{\tilde{b}}(x)$ respectively. Then according to Dubois and Prade (1980), Liu and Iwamura (1998) and Zadeh (1978),

$$pos(\tilde{a} * \tilde{b}) = \sup\left\{\min\left(\mu_{\tilde{a}}(x), \mu_{\tilde{b}}(y)\right), x, y \in \Re\right\}, x * y \tag{30}$$

where the abbreviation pos represents possibility, * is any one of the relations >, <, =, ≤, ≥ and $\Re$ represents set of real numbers.

$$nes(\tilde{a} * \tilde{b}) = 1 - pos(\tilde{a} * \tilde{b}) \tag{31}$$

where the abbreviation *nes* represents necessity.

Similarly, possibility and necessity measures of $\tilde{a}$ with respect to $\tilde{b}$ are denoted by $\prod_{\tilde{b}}(\tilde{a})$ and $N_{\tilde{b}}(\tilde{a})$ respectively and are defined as

$$\prod_{\tilde{b}}(\tilde{a}) = \sup\left\{\min\left(\mu_{\tilde{a}}(x), \mu_{\tilde{b}}(x)\right), x, \in \Re\right\} \tag{32}$$

$$N_{\tilde{b}}(\tilde{a}) = \min\left\{\sup\left(\mu_{\tilde{a}}(x), 1 - \mu_{\tilde{b}}(x)\right), x, \in \Re\right\} \tag{33}$$

If $\tilde{a}, \tilde{b} \subseteq \Re$ and $\tilde{c} = f(\tilde{a}, \tilde{b})$ where $f : \Re \times \Re \to \Re$ is a binary operation then membership function $\mu_{\tilde{c}}$ of $\tilde{c}$ is defined as Dubois and Prade (1980)

$$\text{For each } z \in \Re, \mu_{\tilde{c}}(z) = \sup\left\{\min\left(\mu_{\tilde{a}}(x), \mu_{\tilde{b}}(y)\right), x, y \in \Re\right\}, \text{and } z = f(x, y) \tag{34}$$

LFN: a LFN $\tilde{a} = (a_1, a_2)$ has two parameters a_1, a_2, where $a_1 < a_2$ and is characterised by the membership function $\mu_{\tilde{a}}(x)$, given by

$$\mu_{\tilde{a}}(x) = \begin{cases} 0 & \text{for} \quad x \le a_1 \\ \dfrac{x - a_1}{a_2 - a_1} & \text{for} \quad a_1 \le x \le a_2 \\ 1 & \text{for} \quad x \ge a_2 \end{cases} \tag{35}$$

Triangular fuzzy number (TFN): a TFN $\tilde{a} = (a_1, a_2, a_3)$ has three parameters a_1, a_2, a_3, where $a_1 < a_2 < a_3$ and is characterised by the membership function $\mu_{\tilde{a}}(x)$, given by

$$\mu_{\tilde{a}}(x) = \begin{cases} \dfrac{x - a_1}{a_2 - a_1} & \text{for} \quad a_1 \le x \le a_2 \\ \dfrac{a_3 - x}{a_3 - a_2} & \text{for} \quad a_2 \le x \le a_3 \\ 0 & \text{otherwise} \end{cases} \tag{36}$$

According to above definitions following lemmas can easily be derived.

Lemma 1: if $\tilde{a} = (a_1, a_2, a_3)$ be a TFN with $0 < a_1$ and b is a crisp number then

$$nes(\tilde{a} > b) \geq \alpha \text{ iff } (b - a_1)/(a_2 - a_1) \leq 1 - \alpha$$

Lemma 2: if $\tilde{a} = (a_1, a_2, a_3)$ be a TFN and $\tilde{b} = (b_1, b_2)$ be a LFN with $0 < a_1$ and $0 < b_1$ then

$$\Pi_{\tilde{a}}(\tilde{b}) = \begin{cases} 1 & \text{if } \quad a_2 \geq b_2 \\ \dfrac{a_3 - b_1}{a_3 - a_2 + b_2 - b_1} & \text{if } \quad a_2 \leq b_2 \text{ and } a_3 > b_1 \\ 0 & \text{otherwise} \end{cases}$$

Lemma 3: if $\tilde{a} = (a_1, a_2, a_3)$ be a TFN and $\tilde{b} = (b_1, b_2)$ be a LFN with $0 < a_1$ and $0 < b_1$ then

$$N_{\tilde{a}}(\tilde{b}) = \begin{cases} 1 & \text{if } \quad a_1 \geq b_2 \\ \dfrac{a_2 - b_1}{a_2 - a_1 + b_2 - b_1} & \text{if } \quad a_2 > b_1 \text{ and } b_2 > a_1 \\ 0 & \text{otherwise} \end{cases}$$

Lemma 4: if $\tilde{a} = (a_1, a_2, a_3)$ be a TFN and b be a crisp number with $0 < a_1$ and $0 < b$,

$$\prod\nolimits_{\tilde{a}}(b) = N_{\tilde{a}}(b) = \begin{cases} \dfrac{b - a_1}{a_2 - a_1} & \text{if } \quad a_2 \geq b \geq a_1 \\ \dfrac{a_3 - b}{a_3 - a_2} & \text{if } \quad a_3 \geq b \geq a_2 \\ 0 & \text{otherwise} \end{cases}$$

Appendix B

Optimisation with stochastic constraints (Charnes and Cooper, 1959): let $x = (x_1, x_2, \ldots, x_n)^T$, be the decision vector, $y = (y_1, y_2, \ldots, y_N)^T$ be the vector of N random variables with known mean and standard deviation, where $y_1, y_2, \ldots, y_N$ represents N parameters of the problem, then a stochastic nonlinear programming problem (SNLP) can be stated in standard form as follows:

$$\begin{array}{ll} \text{Find} & x = (x_1, x_2, \ldots x_n)^T \\ \text{which minimize/maximize} & f(x, y) \\ \text{subject to} & P(\varphi_r(x, y) \geq 0) \geq p_r \ (r = 1, \ldots, m) \end{array} \tag{37}$$

where $P(\varphi_r(x, y) \geq 0)$ representsprobability of the event $\varphi_r(x, y)$

According to Charnes and Cooper (1959), if all $y_i (i = 1,2,\ldots,N)$ follow independent normal distribution, the stochastic problem stated above is equivalent to following crisp nonlinear programming problem.

Find $x = (x_1, x_2, \ldots x_n)^T$

which minimize/maximize $f(x, \overline{y})$

subject to $\overline{\varphi}_r - \varepsilon_r \left[\sum \left(\frac{\partial \varphi_r (x, \overline{y})}{\partial y_i} \right) \sigma_{y_i}^2 \right]^{\frac{1}{2}} \geq 0 \; (r = 1, \ldots, m)$

where $\overline{\varphi}_r$ and σ_{φ_r} are mean and standard deviation of $\varphi_r(x, y)$respectively. (38)

Also, $\varepsilon_r$is given by $p_r = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\varepsilon_r} e^{-t^2/2} dt$

Appendix C

Following are the list of TFs which are used to test the efficiency of the proposed MPSO algorithm. It is observed that results obtained by MPSO for different TFs are equal to the global optimum in most of the cases and in other cases solutions are very near to global optima. This implies that this algorithm can be used as decision-making tool for different constrained/unconstrained optimisation problems.

TF-1 (taken from Bessaou and Siarry, 2001)

$$ES(x_1, x_2) = -\cos(x_1) \times \cos(x_2) \times \exp\left\{-\left[(x_1 - \pi)^2 + (x_2 - \pi)^2\right]\right\},$$
$$-10 \leq x_1, x_2 \leq 10.$$

This TF has one global minima at $(x_1, x_2) = (\pi, \pi)$ and ES$(\pi, \pi) = -1$.

TF-2 (taken from Bessaou and Siarry, 2001)

$$MZ(x_1, x_2, \ldots, x_n) = -\sum \sin(x_i)\left[\text{in}(i.(x_i)^2 / \pi\right]^{2m}, \; -\pi \leq x_1, x_2, \ldots, x_n \leq \pi$$

where $m = 10$. For $n = 2$, it has one global minima at $(x_1, x_2) = (2.25, 1.57)$ and $MZ(2.25, 1.57) =-1.80$.

TF-3 (taken from Michalewicz, 1992):

Minimize $F(x_1, x_2) = (x_1 - 2)_2 + (x_2 - 1)_2$,

such that $-x_1^2 + x_2^2 >= 0, \; x_1 + x_2 <= 2, -5 <= x_1, x_2 <= 5$.

It has one global minima at (x_1, x_2) =(1, 1), and $F(1, 1) = 1$.

TF-4 (taken from Bessaou and Siarry, 2001):

$$F2(x_1, x_2) = 100 \times (x_2^2 - x_1) + (1 - x_1), -2.048 \leq x_1, x_2 \leq 2.048.$$

It has one minima at $(x_1, x_2) = (2.048, 0)$ and F2(2.048, 0) = −205.8480.

TF-5 (taken from Bessaou and Siarry, 2001):

$$SH(x_1, x_2)\sum_{j=1}^{5} j\times\cos[(j+1)\times x_1 + j]\times\sum_{j=1}^{5} j\times\cos[(j+1)\times x_2 + j], -10 \le x_1, x_2 \le 10$$

This problem has 760 local minima and 18 global minima. At global minima (x_1, x_2), SH(x_1, x_2) = −186.7309.

TF-6 (taken from Bessaou and Siarry, 2001):

$$RC(x_1, x_2) = \left\{x_2 - \left[5/(4\times\pi^2)\right].x_1^2 + (5/\pi)\times x_1 - 6\right\}^2 + 10\times\left\{1-[1/(8\pi)]\right\}$$
$$\times\cos(x_1)+10, -5 \le x_1 \le 10, 0 \le x_2 \le 15$$

This problem has three global minima at (x_1, x_2) = ($-\pi$, 12.275), (π, 2.275), (9.42478, 2.475) and RC(x_1, x_2) = 0.397887 at any one of these minima.

TF-7 (taken from Bessaou and Siarry, 2001)

$$\text{Minimize } F_n(x_1, x_2, \ldots, x_n) = \sum_{j=1}^{n-1}\left[100\times\left(x_j^2 - x_{j+1}\right)^2 + \left(1-x_j\right)^2\right],$$
$$-1 \le x_1, x_2, \ldots, x_n \le 5.$$

Two functions F_2 and F_4 are used. This problems has one global minima at (x_1, x_2,…, x_n) = (1, 1,…, 1) and F_n(1, 1,…,n) = 0.

TF-8 (taken from Bessaou and Siarry, 2001):

$$BH(x_1, x_2) = x_1^2 + 2\times x_2^2 - 0.3\times\cos(3\pi x_1)\times\cos(4\pi x_2) + 0.3, -5 \le x_1, x_2 \le 5$$

This problem has one global minima at (x_1, x_2) = (0, 0) and BH(0, 0) = 0.

TF-9 (taken from Michalewicz, 1992):

$$\text{Minimize } F(x_1, x_2) = 100\times\left(x_2 - x_1^2\right)^2 + (x_1 - 1),$$
$$\text{such that } x_1 + x_2^2 \ge 0, x_1^2 + x_2 \ge 0, \quad -0.5 \le x_1 \le 0.5, -1.0 \le x_2 \le 1.0$$

It has one global minima at (x_1, x_2) = (0.5, 0.25), and F(0.5, 0.25) = 0.25.

TF-10 (taken from Bessaou and Siarry, 2001):

$$Z_n(x_1, x_2, \ldots, x_n) = \left(\sum_{j=1}^{n} x_j^2\right) + \left(\sum_{j=1}^{n} 0.5\times x_j\right)^2 + \left(\sum_{j=1}^{n} 0.5j\times x_j\right)^4,$$
$$-5 \le x_1, x_2, \ldots, x_n \le 5$$

It has one global minima at (x_1, x_2,…, x_n) = (0, 0,…, 0) and Z_n(0, 0,…, 0) = 0.