Particle Swarm Optimization for Tackling Continuous Review Inventory Models

K.E. Parsopoulos[1,3], K. Skouri[2], and M.N. Vrahatis[1,3]

[1] Computational Intelligence Laboratory (CI Lab), Department of Mathematics, University of Patras, GR–26110 Patras, Greece
{kostasp,vrahatis}@math.upatras.gr
[2] Department of Mathematics, University of Ioannina, GR–45110 Ioannina, Greece
kskouri@uoi.gr
[3] University of Patras Artificial Intelligence Research Center (UPAIRC), University of Patras, GR–26110 Patras, Greece

Abstract. We propose an alternative algorithm for solving continuous review inventory model problems for deteriorating items over a finite horizon. Our interest focuses on the case of time–dependent demand and backlogging rates, limited or infinite warehouse capacity and taking into account the time value of money. The algorithm is based on Particle Swarm Optimization and it is capable of computing the number of replenishment cycles as well as the corresponding shortage and replenishment instances concurrently, thereby alleviating the heavy computational burden posed by the analytical solution of the problem through the Kuhn–Tucker approach. The proposed technique does not require any gradient information but cost function values solely, while a penalty function is employed to address the cases of limited warehouse capacity. Experiments are conducted on models proposed in the relative literature, justifying the usefulness of the algorithm.

1 Introduction

Inventory maintenance of deteriorating items is a major concern in the supply chain of business organizations, since many products undergo decay or deterioration over time. Deterioration and demand rates play a crucial role in such problems. For this purpose, relative models have been proposed in the literature for different deterioration rates [1, 2, 3]. Also, a constant demand rate is usually valid in the mature stage of a product's life cycle, while it can be linearly approximated in the growth and/or end stage of the life cycle. Such models with linearly time varying demand were studied initially in [4, 5], while most recent trends are reported in [6, 7].

Another important issue of inventory systems is the management of unsatisfied demand. Often, complete backlogging of unsatisfied demand is assumed. However, in practice, there are customers who are willing to wait and receive their orders at the end of shortage period, while others are not. To this extent, considerable attention has been paid in the last few years to inventory

M. Giacobini et al. (Eds.): EvoWorkshops 2008, LNCS 4974, pp. 103–112, 2008.

models with partial backlogging, where the backlogging rate can be modeled taking into account the behavior of customers [8, 9, 10]. In addition, the effects of inflation and time value of money are vital in practical environments especially in developing countries. Recently, Chern *et al.* [11] studied an inventory model for deteriorating items with time varying demand and partial backlogging, taking into account the time value of money. This model can be considered as a generalization of older models. Basic assumption of the model is the unlimited storage capacity. However, this assumption does not often hold in practice.

Particle Swarm Optimization (PSO) was introduced in 1995 as a stochastic population–based algorithm for numerical optimization by Eberhart and Kennedy [12, 13]. It belongs to the class of *swarm intelligence* algorithms, whose dynamics are based on principles that govern socially organized groups of individuals [14]. Up–to–date, PSO has received a lot of attention from researchers due to its efficiency in solving different problems in science and engineering [15, 14, 16, 17].

This paper is devoted to the investigation of the efficiency of PSO on solving an extended version of the model of Chern *et al.* [11], where limited storage is also considered. The detection of replenishment cycles, replenishment instances and replenishment orders is required, while warehouse capacity constraints can be present. The underlying optimization problem is mixed–integer with the solutions having variable length as well as posing constraints on the magnitude of their components, since the ordering of time instances must be preserved. For this purpose, hard bounding constraints are posed on the search points, while a penalty function is employed to tackle capacity constraints. The workings of the proposed approach are illustrated on three test problems considered in [11, 9].

The rest of the paper is organized as follows: Section 2 contains the necessary background information on the considered inventory models and PSO. Section 3 describes the proposed approach, while experimental results are reported in Section 4. The paper concludes in Section 5.

2 Background Information

In the following subsections we describe the basic concepts of the continuous review inventory model proposed in [11] as well as the PSO algorithm.

2.1 The Considered Review Inventory Model

The model under investigation is an extension of the model of Chern *et al.* [11], assuming that the storage can also have limited capacity. The selection of this model was based on its generality due to the time varying demand, deterioration and backlogging rates. Thus, it can be considered to include different previously proposed models as special cases. The assumptions under which the model is developed are:

Table 1. Notation used for the parameters of the model

Param.	Description
n	Number of replenishment cycles during the planning horizon.
s_i	Time at which shortage starts during the i–th cycle, $i = 1, 2, \ldots, n$.
t_i	Time at which the i–th replenishment is made, $i = 1, 2, \ldots, n$.
r	Discount rate.
i_1	Internal inflation rate, which is varied by the company operation status.
i_2	External inflation rate, which is varied by the social economical situation.
r_1	$r - i_1$, discount rate minus the internal inflation rate.
r_2	$r - i_2$, discount rate minus the external inflation rate.
c_0	Internal fixed purchasing cost per order.
c_p	External variable purchasing cost per unit.
c_{h_1}	Internal inventory holding cost per unit and per unit of time.
c_{h_2}	External inventory holding cost per unit and per unit of time.
c_{b_1}	Internal backlogging cost per unit and per unit of time.
c_{b_2}	External backlogging cost per unit and per unit of time.
c_{l_1}	Internal cost of lost sales per unit and per unit of time.
c_{l_2}	External cost of lost sales per unit and per unit of time.
W	Storage area or volume.

1. The planning horizon is finite and equal to H time units. The initial and final inventory levels during the planning horizon are both set to zero.
2. Replenishment is instantaneous (replenishment rate is infinite).
3. The lead–time is zero.
4. The on hand inventory deteriorates at time varying deterioration rate $\theta(t)$.
5. The demand rate at time $t \in [0, H]$, is a continuous function $f(t)$.
6. The system allows for shortages in all cycles, and each cycle starts with shortages.
7. Shortages are backlogged at a rate $\beta(x)$, which is a non–increasing function of the waiting time x up to the next replenishment, with $0 \leqslant \beta(x) \leqslant 1$ and $\beta(0) = 1$.

The notation that will be used hereafter is reported in Table 1, along with the descriptions of the parameters. Let

$$\delta(t) = \int_0^t \theta(u)du,$$

then the total cost of the inventory system during the planning horizon H, as defined by Chern *et al.* [11], is:

$$
\begin{aligned}
TC(n, s_i, t_i) = &\sum_{i=1}^{n} c_0 e^{-r_1 t_i} \\
&+ \sum_{i=1}^{n} c_p e^{-r_2 t_i} \left(\int_{s_{i-1}}^{t_i} \beta\left(t_i - t\right) f(t) dt + \int_{t_i}^{s_i} e^{\delta(t)-\delta(t_i)} f(t) dt \right) \\
&+ \sum_{i=1}^{n} \sum_{j=1}^{2} c_{h_j} \int_{t_i}^{s_i} e^{-r_j t} \int_{t}^{s_i} e^{\delta(u)-\delta(t)} f(u) du dt \\
&+ \sum_{i=1}^{n} \sum_{j=1}^{2} \frac{c_{b_j}}{r_j} \int_{s_{i-1}}^{t_i} \left(e^{-r_j t} - e^{-r_j t_i}\right) \beta(t_i - t) f(t) dt \\
&+ \sum_{i=1}^{n} \sum_{j=1}^{2} c_{l_j} \int_{s_{i-1}}^{t_i} e^{-r_j t} \left[1 - \beta\left(t_i - t\right)\right] f(t) dt, \qquad (1)
\end{aligned}
$$

subject to $s_0 = 0$, $s_{i-1} < t_i \leqslant s_i$, and $s_n = H$. Considering additionally the capacity constraints, we end up with the following constrained, mixed–integer minimization problem:

$$
\begin{aligned}
&\min_{n, t_i, s_i} TC(n, t_i, s_i) \\
&\text{s.t.} \quad \int_{t_i}^{s_i} e^{\delta(u)-\delta(t_i)} f(u) du \leqslant W, \qquad (2) \\
&s_0 = 0, \;\; s_n = H, \;\; s_{i-1} < t_i \leqslant s_i, \;\; i = 1, 2, \ldots, n.
\end{aligned}
$$

Ignoring the constraints, $s_{i-1} < t_i \leqslant s_i$, $i = 1, 2, \ldots, n$, and for given n, the application of the classical Kuhn–Tucker approach can find the optimal solution after solving 2^n nonlinear systems of equations with $2n$ up to $3n$ variables [11,9]. Clearly, the computational cost for solving the problem for unknown n using the Kuhn–Tucker approach is heavy. For this purpose, we propose a technique for concurrent computation of n, t_i, and s_i (which are used for determining the size of the replenishment order). The approach is based on the application of the PSO algorithm, which is described in the next section.

2.2 Particle Swarm Optimization

PSO employs a population of search points that probe the search space simultaneously. The population is called a *swarm*, while the search points are called the *particles*. The particles are initialized randomly in the search space and move with an adaptive velocity within it. Also, each particle has a memory where it stores its best experience during the search, i.e., the best position it has ever visited in the search space. An iteration of the algorithm corresponds to an update of the positions of all particles. The update for each particle is performed by computing the new velocity of the particle, taking into account both its own experience as well as the experience of other particles. These particles are said to constitute its *neighborhood*.

Let $S \subset \mathbb{R}^D$ be a D–dimensional search space and $F : S \rightarrow \mathbb{R}$ be the objective function (without loss of generality only the minimization case is considered). A swarm is a set of N particles, $\mathbb{S} = \{x_1, x_2, \ldots, x_N\}$, each of which, is a D–dimensional search point, $x_i = (x_{i1}, x_{i2} \ldots, x_{iD})^\top \in S$, $i = 1, \ldots, N$, and it has an adaptive velocity, $v_i = (v_{i1}, v_{i2}, \ldots, v_{iD})^\top$. Also, each particle x_i remembers the best position, $b_i = (b_{i1}, b_{i2}, \ldots, b_{iD})^\top \in S$, it has ever visited.

A neighborhood, NB_i, is defined for each particle x_i, $i = 1, 2, \ldots, N$. There are several different neighborhood schemes (also called *topologies*) presented in the literature [18, 19]. Most of them are defined based on the indices of the particles rather than their actual positions in S. The most common scheme is the *ring topology*, where the particles are assumed to be organized on a ring, communicating with their immediate neighbors. Under this topology, a neighborhood of radius q of x_i is defined as the set $NB_i^q = \{x_{i-q}, \ldots, x_i, \ldots, x_{i+q}\}$, where x_1 follows immediately after x_N. We denote with g_i the index of the best particle in NB_i, i.e., the particle that has visited the best position in S in terms of its function value, $F(b_{g_i}) \leqslant F(b_j)$, for all j such that $x_j \in NB_i$.

Let t to be the iteration counter. Then, the swarm is updated using the equations [20],

$$v_{ij}(t+1) = \chi \Big[v_{ij}(t) + c_1 R_1 \Big(b_{ij}(t) - x_{ij}(t) \Big) + c_2 R_2 \Big(b_{g_i,j}(t) - x_{ij}(t) \Big) \Big], \quad (3)$$

$$x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t+1), \quad (4)$$

where $i = 1, 2, \ldots, N$, $j = 1, 2, \ldots, D$. The parameter χ is called *the constriction coefficient* and it is used to constrain the magnitude of the velocities during the search. The positive constants c_1 and c_2 are referred to as the *cognitive* and *social* parameter, respectively; while R_1, R_2 are random variables uniformly distributed in $[0, 1]$. Default values for χ, c_1 and c_2 are determined in the theoretical analysis of Clerc and Kennedy [20]. The best positions of the particles are updated at each iteration according to the relation:

$$b_i(t+1) = \begin{cases} x_i(t+1), & \text{if } F(x_i(t+1)) < F(b_i(t)), \\ b_i(t), & \text{otherwise,} \end{cases} \qquad i = 1, 2, \ldots, N.$$

The particles are usually constrained to move strictly in the search space, posing explicit bounds on each component of the particles.

3 The Proposed Approach

In the proposed approach, PSO is used to determine both the number of replenishment cycles, n, as well as the corresponding solution, $(s_0, t_1, s_1, \ldots, t_n, s_n)$, concurrently. Since the first and the last component of a possible solution vector are known *a priori* for a specific problem instance ($s_0 = 0$, $s_n = H$), it is sufficient to determine only the remaining $(2n-1)$ solution components, t_1, s_1, ..., t_n, which will be called the *time components* hereafter. However, in our approach, n is also a variable, rising questions regarding the encoding of the variable length PSO particles. For this purpose, we consider a fixed, user–defined maximum number of cycles, $n_{\max} \geqslant n$, which is used to fixate the particles' dimension to $D = 2n_{\max}$. Then, the i–th particle of the swarm is defined as a D–dimensional vector of the form

$$x_i = (x_{i1}, x_{i2}, \ldots, x_{iD}) = (n, t_1, s_1, \ldots, t_n, s_n, \ldots, t_{n_{\max}}) ,$$

in order to facilitate arithmetic operations, while only the first $2n$ components that correspond to n, t_1, s_1, ..., t_n, are used for the evaluation of x with the cost function $TC(x)$, i.e., $TC(x) = TC(n, s_0, t_1, s_1, \ldots, t_n, s_n)$. The rest of the components are simply ignored. Alternatively, one could incorporate special operators for updating particles of variable length. Such operators have been introduced in the literature for PSO [21]. However, their use in our case requires special handling regarding the ordering of the particle's components, due to constrictions posed by the problem on the time components. Therefore, the simpler solution of assuming a reasonable maximum number of cycles and defining particles of fixed and equal dimensionality, was adopted.

The values of n must be integers and lie within the range $[1, n_{\max}]$. The restriction of the corresponding particle component, x_{i1}, to integer values would require the use of special operators for the particle's update. In order to retain the simplicity and straightforward applicability of the algorithm, we allowed x_{i1} to assume real values in the range $[0.6, n_{\max}]$ in the particle's update procedure, while, for the function evaluation, we round its value to the nearest integer. Such rounding approaches have been shown to work efficiently also in different problems with PSO [22]. The initialization of x_{i1} for each particle is performed randomly and uniformly within $[0.6, n_{\max}]$.

The rest of the components of the particle (i.e., the time components) must lie within the range $[0, H]$, preserving the ordering $t_1 \leqslant s_1 \leqslant \cdots \leqslant t_{n_{\max}}$. For this purpose, the j–th component, x_{ij}, is constrained within the range

$$x_{i,j-1} \leqslant x_{ij} \leqslant x_{i,j+1}, \qquad j = 2, 3, \ldots, D-1, \tag{5}$$

in the particle update procedure at each iteration. Clearly, this restriction fosters the danger of biasing the components of the particles towards H, if one of the preceding time components assumes a large value close to H. If this effect takes place in the initialization phase, then it can be detrimental for the algorithm's performance, since it will inhibit the initialization of particles in specific parts of the search space. In order to avoid such an effect, we initialize each time component of x_i randomly in equidistant intervals within the range $[0, H]$, i.e.,

$$x_{ij}^{\text{initial}} = (j - 2 + \text{rand})\Delta, \qquad j = 2, 3, \ldots, D,$$

where $\Delta = H/(D-1)$ and "rand" is a random variable uniformly distributed in $[0, 1]$.

In the cases where constraints on the capacity of the warehouse were considered, the following penalty function was used:

$$TC_{\text{pen}}(n, s_i, t_i) = TC(n, s_i, t_i) + \sum_{k=1}^{K} \frac{TC(n, s_i, t_i)}{n}, \tag{6}$$

where K is the number of violated constraints in Eq. (2), $0 \leqslant K \leqslant n$. Thus, for each violated constraint, a fixed portion of the cost function is added to the actual cost function value.

4 Experimental Analysis

The proposed approach was applied on the following test problems, denoted as TP1, TP2, and TP3, respectively:

Test Problem 1 [9]. This problem is based on a simplified version of the model described by Eq. (1), with demand rate $f(t) = 20+2t$, $\beta(x) = e^{-\alpha x}$, $r_1 = r_2 = 0$, $c_0 = 100$, $c_p = 0.2$, $c_{b_2} = 1.5$, $c_{l_2} = 0.5$, $c_{h_2} = 55$, $c_{h_1} = c_{b_1} = c_{l_1} = 0$ and $\theta(t) = 0.01$. The problem was considered for three different levels of the parameter α, namely $\alpha = 0.08$, 0.05, and 0.02.

Test Problem 2 [11]. In this problem, the shortages are completely backlogged, i.e., $\beta(x) = 1$ for all t, the demand rate is $f(t) = 200 + 50t$, and the parameters assume the values: $H = 10$, $c_0 = 80$, $c_{h_1} = 0.2$, $c_{h_2} = 0.4$, $c_{b_1} = 0.5$, $c_{b_2} = 0.4$, $c_p = 9$, $r = 0.2$, $i_1 = 0.08$, $i_2 = 0.09$, and $\theta(t) = 0.01$.

Test Problem 3 [11]. In this problem, the shortages are also completely backlogged, i.e., $\beta(x) = 1$ for all t, the demand rate is $f(t) = 200 + 50t - 3t^2$, and the parameters assume the values: $H = 10$, $c_0 = 80$, $c_{h_1} = 0.2$, $c_{h_2} = 0.4$, $c_{b_1} = 0.8$, $c_{b_2} = 0.6$, $c_p = 15$, $r = 0.2$, $i_1 = 0.08$, $i_2 = 0.1$, and $\theta(t) = 0.01$.

Regarding the parameters of PSO, the typical values $\chi = 0.729$, $c_1 = c_2 = 2.05$, derived from the theoretical analysis of Clerc and Kennedy [20] were used. The neighborhood radius was equal to 1 for all particles, while the swarm size was set to $N = 100$ in all experiments, and the algorithm was terminated after a maximum number of 15000 iterations, in all test problems. The maximum number of replenishments, which is used for the determination of the particles' dimension, was equal to $n_{\max} = 20$ for all test problems, resulting in 40–dimensional optimization problems (recall that $D = 2n_{\max} = 40$). The maximum inventory size in the constrained cases of TP1 was equal to $W = 90$, while for TP2 and TP3 it was $W = 300$. For each test problem and case, 50 independent experiments were performed to derive statistics regarding the performance of the proposed approach.

The first component, x_{i1}, of the particle, which corresponds to n (it is not a time component), is initialized randomly and uniformly within $[0.6, n_{\max}]$. The obtained results are reported in Table 2. More specifically, the first column of the table specifies the test problem, while the second and third columns specify the

Table 2. The obtained results in terms of the required number of iterations

Problem	α	W	n^*	TC^*	Suc.	Mean	St.D.	Min	Max
1	0.08	∞	3	685.888	50/50	223.96	66.47	106	421
	0.08	90	3	688.354	50/50	2985.30	2364.64	444	11568
	0.05	∞	3	687.686	50/50	205.84	44.36	98	305
	0.05	90	3	690.702	50/50	1676.90	971.32	466	5130
	0.02	∞	3	689.405	50/50	219.66	58.69	111	432
	0.02	90	3	693.010	50/50	988.74	557.28	339	2826
2		∞	1	21078.04	50/50	8.20	1.88	5	13
		300	1	21078.04	50/50	46.48	28.01	10	133
3		∞	1	29990.68	50/50	7.64	1.86	5	13
		300	1	29990.68	50/50	28.30	19.29	9	86

corresponding value of the parameter α (applicable only to TP1) and the value of W. Infinite warehouse capacity in the unconstrained cases is denoted as "∞". In the rest of the columns, the detected optimal number of replenishment cycles, n^*, is reported per case, along with the corresponding value, TC^*, of the cost function. Also, the number of experiments where the algorithm was successful, i.e., it detected the optimal solution, is reported, along with the mean, standard deviation, minimum and maximum number of iterations required to obtain the solution. Since PSO is a stochastic algorithm, the obtained solution at each experiment for the same problem and case is expected to vary slightly. One of the obtained optimal solutions for each case is reported in Table 3.

Table 3. The obtained solutions rounded up to 6 decimal digits

Problem	α	W	n^*	s_0	t_1	s_1	t_2	s_2	t_3	s_3
1	0.08	∞	3	0.0	1.518444	4.546430	5.776002	8.463058	9.536687	12.0
	0.08	90	3	0.0	1.644925	4.854017	6.164020	8.711135	9.873837	12.0
	0.05	∞	3	0.0	1.470722	4.529375	5.732727	8.448895	9.506381	12.0
	0.05	90	3	0.0	1.609384	4.881638	6.175431	8.721005	9.873838	12.0
	0.02	∞	3	0.0	1.434863	4.523818	5.702551	8.447381	9.487877	12.0
	0.02	90	3	0.0	1.575821	4.907751	6.179812	8.724783	9.873839	12.0
2		∞	1	0.0	10.0	10.0				
		300	1	0.0	10.0	10.0				
3		∞	1	0.0	10.0	10.0				
		300	1	0.0	10.0	10.0				

It is clear that the imposition of constraints in TP1 increases its difficulty significantly, as it is revealed by the increased mean number of iterations required by the algorithm. Nevertheless the algorithm was successful in all cases, detecting both n^* and the corresponding solution without any user intervention. The hard constraints posed on the particles do not prevent PSO from detecting the optimal value, although the bounds change continuously for each particle and iteration, in order to preserve the ordering of the time components of the particles. The penalty function defined in Eq. (6) was adequate to prevent PSO from converging to unfeasible solutions in the cases with constrained warehouse capacity, without any assumptions needed regarding the feasibility of the initial population. Thus, all solutions reported in Table 3 were feasible.

The same observations can be made also for TP2 and TP3. However, in these cases, the reported solutions coincide for the unconstrained and constrained warehouse capacity, since the optimal solution corresponds to a single cycle, and time components lie exactly on the bounds of the time horizon. Overall, the considered test problems were addressed efficiently, rendering the proposed approach a useful alternative for solving continuous review inventory models of the considered type.

5 Conclusions

A major concept in supply chain is the maintenance of inventories of deteriorating items. Such problems are usually addressed through analytical approaches, based on the theory of Kuhn–Tucker. However, the corresponding computational cost is high and the problems reported in the literature usually do not take into account the limited warehouse capacity.

We proposed an alternative approach for solving such problems through PSO. The proposed approach computes the number of replenishment cycles as well as the corresponding shortage and replenishment instances, concurrently, without the need of gradient information. Experiments conducted on an extension of a recently proposed model indicate that the proposed approach can tackle the problem efficiently. Future work will consider further test problems as well as the development of specialized operators that can incorporate model information in the PSO update schemes.

References

1. Ghare, P.M., Shrader, G.F.: A model for exponentially decying inventories. J. Ind. Eng. 14, 238–243 (1963)
2. Covert, R.P., Philip, G.C.: An EOQ model for items with Weibull distribution deterioration. Am. Inst. Ind. Eng. Trans. 5, 323–326 (1973)
3. Tadikamalla, P.R.: An EOQ inventory model for items with Gamma distribution. AIIE Trans. 5, 100–103 (1978)
4. Resh, M., Friedman, M., Barbosa, L.C.: On a general solution of the deterministic lot size problem with time–proportional demand. Oper. Res. 24, 718–725 (1976)

5. Donaldson, W.A.: Inventory replenishment policy for a linear trend in demand: An analytical solution. Oper. Res. Quart. 28, 663–670 (1977)
6. Goyal, S.K., Giri, B.C.: Recent trends in modeling of deteriorating inventory. Eur. J. Oper. Res. 134, 1–16 (2001)
7. Raafat, F.: Survey of literature on continuously deteriorating inventory model. J. Oper. Res. Soc. 42, 27–37 (1991)
8. Chang, H.J., Dye, C.Y.: An EOQ model for deteriorating items with time varying demand and partial backlogging. J. Oper. Res. Soc. 50, 1176–1182 (1999)
9. Skouri, K., Papachristos, S.: A continuous review inventory model, with deteriorating items, time–varying demand, linear replenishment cost, partially time–varying backlogging. Applied Mathematical Modelling 26, 603–617 (2002)
10. Teng, J.T., Chang, H.J., Dye, C.Y., Hung, C.H.: An optimal replenishment policy for deteriorating items with time–varying demand and partial backlogging. Oper. Res. Let. 30, 387–393 (2002)
11. Chern, M.S., Yang, H.L., Teng, J.T., Papachristos, S.: Partial backlogging inventory lot–size models for deteriorating items with fluctuating demand under inflation. European Journal of Operational Research (in press 2007)
12. Eberhart, R.C., Kennedy, J.: A new optimizer using particle swarm theory. In: Proc. 6th Symp. Micro Machine and Human Science, pp. 39–43 (1995)
13. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: Proc. IEEE Int. Conf. Neural Networks, vol. IV, pp. 1942–1948 (1995)
14. Kennedy, J., Eberhart, R.C.: Swarm Intelligence. Morgan Kaufmann, San Francisco (2001)
15. Engelbrecht, A.P.: Fundamentals of Computational Swarm Intelligence. Wiley, Chichester (2006)
16. Parsopoulos, K.E., Vrahatis, M.N.: Recent approaches to global optimization problems through particle swarm optimization. Natural Computing 1(2–3), 235–306 (2002)
17. Parsopoulos, K.E., Vrahatis, M.N.: On the computation of all global minimizers through particle swarm optimization. IEEE Transactions on Evolutionary Computation 8(3), 211–224 (2004)
18. Kennedy, J.: Small worlds and mega–minds: Effects of neighborhood topology on particle swarm performance. In: Proc. IEEE Congr. Evol. Comput., Washington, D.C., USA, pp. 1931–1938. IEEE Press, Los Alamitos (1999)
19. Suganthan, P.N.: Particle swarm optimizer with neighborhood operator. In: Proc. IEEE Congr. Evol. Comput., Washington, D.C., USA, pp. 1958–1961 (1999)
20. Clerc, M., Kennedy, J.: The particle swarm–explosion, stability, and convergence in a multidimensional complex space. IEEE Trans. Evol. Comput. 6(1), 58–73 (2002)
21. Ó Neill, M., Leahy, F., Brabazon, A.: Grammatical swarm: A variable–length particle swarm algorithm. In: Nedjah, N., Macedo Mourelle, L. (eds.) Studies in Computational Intelligence, vol. 26, pp. 59–74. Springer, Heidelberg (2006)
22. Laskari, E.C., Parsopoulos, K.E., Vrahatis, M.N.: Particle swarm optimization for integer programming. In: Proceedings of the IEEE 2002 Congress on Evolutionary Computation, Hawaii (HI), USA, pp. 1582–1587. IEEE Press, Los Alamitos (2002)