Techn. Report TR-2015-1, Department of Computer Science & Engineering, University of Ioannina, Greece, October 2015.

Unified Particle Swarm Optimization for Solving Multi-Item Inventory Models with Supplier Selection

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

[1]Department of Computer Science & Engineering,
University of Ioannina, GR-45110 Ioannina, Greece,
Email: kostasp@cs.uoi.gr
[2]Department of Mathematics, University of Ioannina, GR-45110
Ioannina, Greece, Email: kskouri@uoi.gr
[3]Department of Mathematics, University of Patras, GR-26110 Patras,
Greece, Email: vrahatis@math.upatras.gr

Abstract

The multi-item inventory optimization problem with supplier selection is considered to be among the most interesting problems in Operations Research. Constraints such as limited capacity and defective items render the problem a demanding challenge for most solvers. Metaheuristic algorithms have been considered as promising approaches for tackling problems of this type. The present work investigates the performance of the Unified Particle Swarm Optimization on this problem type. The employed model also assumes individual transaction costs and product-dependent holding cost. Different parametrizations of the algorithm are considered under various configurations. The results are statistically analyzed, offering valuable insight.

1 Introduction

Particle Swarm Optimization (PSO) [9] has been placed among the most popular metaheuristic optimization algorithms. This is a consequence of its efficiency as well as its easy implementation, which renders it accessible in diverse scientific fields. Operations Research (OR) has proved to be a challenging application field for PSO. Its effectiveness has been verified in various problems [14]. *Inventory management with supplier selection* is an essential problem in OR, usually formulated as a mixed-integer optimization task that includes purchase, transportation, and inventory costs over multiple periods, under multiple sourcing, criteria, and constraints. Extensions on lot-sizing with supplier selection for multi-period and multi-product cases have been widely studied [1, 3, 7].

In many relevant studies in literature, the proposed models make assumptions that differ from reality. For example, products are frequently considered to be of perfect quality, contradicting reality where they are often imperfect. Thus, the optimal policy shall take into account such quality implications when determining lot size. Examples of such models are [10, 16, 17]. Other models assume reworking of defective products [8], flexible production processes [6], as well as multi-stage lot sizing for imperfect production processes [2].

A model for lot sizing with supplier selection was proposed in [15]. The model takes into consideration imperfect items and limited storage capacity. The underlying optimization problem constitutes a highly constrained mixed-integer minimization task, which was originally solved through the Lindo software [1] as well as a Genetic Algorithm. The latter was shown to be very promising, motivating us for further inquiring different algorithms based on PSO. Besides that, we also admit some modifications that considerably reduce the problem's dimension by eliminating its binary variables. Finally, the constraints are handled by using a penalty function approach [14].

In our experiments, we considered the Unified PSO (UPSO) algorithm [13], which generalizes the original PSO algorithm. Highly competitive UPSO schemes with different exploration / exploitation trade-off have been proposed [11, 14]. Initialization in feasible points is not required, without raising efficiency issues as long as the penalty function is monotonically increasing in the infeasible region as we move away from the feasible one.

The rest of the this work is organized as follows: UPSO is briefly described in Section 2, and the considered model is exposed in Section 3. Experimental settings and results are reported in Section 4. The paper concludes in Section 5.

2 Particle Swarm Optimization

The *Particle Swarm Optimization* (PSO) algorithm was introduced in 1995 [9] as an alternative to Evolutionary Algorithms for numerical optimization. PSO probes the search space by iteratively updating a population, called a *swarm*, of potential solutions, called the *particles*, which move in the search space. The particles have an adaptable velocity and retain in a *memory* the best positions (i.e., positions with lowest function values) they have ever visited.

Exploration is promoted through an *information exchange* mechanism among the particles. Specifically, each particle assumes a (typically index-based) neighborhood. In the *global* PSO variant, also known as *gbest model*, this neighborhood is the whole swarm. Thus, the overall best position is the main information provider for all particles. On the other hand, in the *local* PSO variant, also known as *lbest model*, the neighborhoods are strictly smaller, usually comprising a few particles. Hence, each particle may have its own leading neighbor that influences its velocity update. The most common neighborhood topology is the *ring*, where each particle assumes as neighbors its mates with neighboring indices [14].

Let the bound-constrained global optimization problem,

$$\min_{x \in X \subset \mathbb{R}^n} f(x),$$

[1] http://www.lindo.com

be the problem under consideration. Then, the swarm is a set of N search points,

$$S = \{x_1, x_2, \ldots, x_N\}.$$

Each particle x_i is an n-dimensional point,

$$x_i = (x_{i1}, x_{i2}, \ldots, x_{in})^\top \in X, \quad i \in I = \{1, 2, \ldots, N\},$$

which moves in X by using a velocity (position shift), v_i, while retaining in memory the best position, $p_i \in X$, it has ever visited,

$$v_i = (v_{i1}, v_{i2}, \ldots, v_{in})^\top, \qquad p_i = (p_{i1}, p_{i2}, \ldots, p_{in})^\top \in X.$$

The ring neighborhood of radius m for x_i is defined as the set of indices,

$$NB_i = \{i-m, \ldots, i-1, i, i+1, \ldots, i+m\}.$$

Assume that g_i is the index of the best position found so far by the neighbors of x_i,

$$g_i = \arg \min_{j \in NB_i} f(p_j),$$

and let t denote the iteration counter. Then, according to the popular *constriction coefficient* version of PSO [5], the swarm is updated as follows:

$$v_{ij}^{(t+1)} = \chi \left[v_{ij}^{(t)} + \varphi_1 \left(p_{ij}^{(t)} - x_{ij}^{(t)} \right) + \varphi_2 \left(p_{g_i j}^{(t)} - x_{ij}^{(t)} \right) \right] \tag{1}$$

$$x_{ij}^{(t+1)} = x_{ij}^{(t)} + v_{ij}^{(t+1)}, \tag{2}$$

where $i \in I$, and $j = 1, 2, \ldots, n$. The constriction coefficient χ is used to amplify the magnitude of the velocities. The other two parameters are stochastic and defined as $\varphi_1 = c_1 \mathscr{R}_1$ and $\varphi_2 = c_2 \mathscr{R}_2$, where c_1 and c_2 are positive constants, called the *cognitive* and the *social* parameter, respectively, and $\mathscr{R}_1$, $\mathscr{R}_2$, are random variables uniformly distributed within $[0, 1]$. Thus, φ_1 and φ_2 assume different values for each value of i, j, and t.

The best positions of the particles are updated according to,

$$p_i^{(t+1)} = \begin{cases} x_i^{(t+1)}, & \text{if } f\left(x_i^{(t+1)}\right) < f\left(p_i^{(t)}\right), \\ p_i^{(t)}, & \text{otherwise,} \end{cases} \qquad i \in I.$$

Based on theoretical analysis [5], the proposed default parameter values for PSO are,

$$\chi = 0.729, \qquad c_1 = c_2 = 2.05. \tag{3}$$

In other works, alternative parameter sets were proposed such as the one in [18],

$$\chi = 0.6, \qquad c_1 = c_2 = 2.83, \tag{4}$$

and the one in [4],

$$\chi = 0.721, \qquad c_1 = c_2 = 1.654. \tag{5}$$

All these parameters were shown to produce competitive performance in various problems.

2.1 Unified Particle Swarm Optimization

Unified PSO (UPSO) is an enhanced PSO scheme that combines the exploration/ exploitation properties of the gbest and lbest PSO models [12, 13]. Originally, UPSO was based on the constriction coefficient variant of PSO, presented in the previous section [12]. However, it can be straightforwardly generalized to other variants.

Let $G_i^{(t+1)}$ and $L_i^{(t+1)}$ denote the velocity update of the i-th particle for the gbest and lbest PSO, respectively,

$$G_{ij}^{(t+1)} = \chi\left[v_{ij}^{(t)} + \varphi_1\left(p_{ij}^{(t)} - x_{ij}^{(t)}\right) + \varphi_2\left(p_{gj}^{(t)} - x_{ij}^{(t)}\right)\right], \quad (6)$$

$$L_{ij}^{(t+1)} = \chi\left[v_{ij}^{(t)} + \varphi_1'\left(p_{ij}^{(t)} - x_{ij}^{(t)}\right) + \varphi_2'\left(p_{g_ij}^{(t)} - x_{ij}^{(t)}\right)\right], \quad (7)$$

where g is the index of the swarm's overall best particle; g_i is the index of the best particle in the neighborhood of x_i; and t denotes the iteration counter. Then, UPSO updates the particle's position as follows [12],

$$U_{ij}^{(t+1)} = (1-u)\,L_i^{(t+1)} + u\,G_i^{(t+1)} \quad (8)$$

$$x_{ij}^{(t+1)} = x_{ij}^{(t)} + U_{ij}^{(t+1)}, \quad (9)$$

where the parameter u, called the *unification factor*, balances the trade-off between the global and local search direction. The plain lbest and gbest PSO models can be obtained for the marginal values $u = 0$ and $u = 1$, respectively. We will henceforth denote these models as UPSO$_\ell$ and UPSO$_g$, respectively.

The basic UPSO scheme can be further extended by introducing a stochastic parameter to imitate mutation in EAs. Mutation promotes diversity, which has a crucial impact on swarm's exploration capability. Thus, Eq. (8) can be modified either as,

$$U_i^{(t+1)} = (1-u)\,L_i^{(t+1)} + r_3\,u\,G_i^{(t+1)}, \quad (10)$$

which is mostly based on lbest PSO, or as,

$$U_i^{(t+1)} = r_3\,(1-u)\,L_i^{(t+1)} + u\,G_i^{(t+1)}, \quad (11)$$

which is mostly based on the gbest PSO. Typically, the mutation parameter r_3 is a normally distributed random variable. The convergence in probability of these variants were studied in [12] and the competitiveness of UPSO against the original PSO was experimentally verified in various problems [14].

3 Problem Formulation

As previously mentioned, we considered both the original model proposed in [15] as well as a simplified one, in combination with a penalty function approach. Let i denote the product, j denote the supplier, and t be the time period. We make the following assumptions according to [15]:

(1) The transaction cost o_j for supplier j is independent of the variety and quantity of the ordered products.

(2) The holding cost h_i of product i is product-dependent.

(3) The demand d_{it} for product i at period t is foreknown over the planning horizon.

(4) Imperfect quality items are kept in stock and sold prior to the next period in a single batch.

(5) A percentage ρ_{ij} of defective items is contained in each lot of product i from supplier j.

(6) The purchasing price of product i from supplier j is b_{ij}. Good quality items are sold in price s_{gi} per unit. Defective items are sold in a single batch at a discounted price s_{di}.

(7) A screening process of the lot is conducted with a unit screening cost c_i for product i.

(8) Each supplier has a limited capacity.

(9) All requirements must be fulfilled in the period in which they occur. No backordering or shortage is allowed.

(10) Storage space w_i is required for product i. The total storage capacity is W.

3.1 Original Model

The main scenario for the original model comprises of a supply chain with multiple products and multiple suppliers. All suppliers are assumed to have limited capacity [15]. The demand over the (finite) planning horizon is known and an optimal procurement strategy is to be determined. Each product can be obtained from a number of approved suppliers. Supplier-dependent transaction cost applies whenever an order is placed. A product-dependent holding cost per period applies for each product in the inventory that is carried across a period in the planning horizon. Finally, storage space can never exceed its maximum value.

The main objective for the decision maker is the determination of the products to order, their quantities, the suppliers, and the corresponding time periods of purchase, in order to maximize the total profit. Assuming that i denotes the product, j denotes the supplier, and t denotes time period, the product quantity under determination is denoted as x_{ijt}. Then, the original model is defined as [15],

$$\begin{aligned}\max f\left(x_{ijt}, y_{jt}\right) \quad = \quad & \left[\sum_i\sum_j\sum_t x_{ijt}\left(1-\rho_{ij}\right)s_{gi} + \sum_i\sum_j\sum_t x_{ijt}\rho_{ij}s_{di}\right] - \\ & \left[\sum_i\sum_j\sum_t x_{ijt}b_{ij} + \sum_j\sum_t o_j y_{jt} + \sum_i\sum_j\sum_t x_{ijt}c_i + \right. \\ & \left. \sum_i\sum_t h_i\left(\sum_{k=1}^{t}\sum_j x_{ijk}\left(1-\rho_{ij}\right) - \sum_{k=1}^{t} d_{ik}\right)\right]. \end{aligned} \tag{12}$$

The model consists of the sum of revenues from selling good and imperfect quality products, subtracting purchase, transaction, screening, and holding costs. Since the problem is defined as profit maximization, the negative of the objective function defines the corresponding minimization task.

The decision variables y_{jt} are binary and defined as [15],

$$y_{jt} = \begin{cases} 1, & \text{if an order is placed to supplier } j \text{ at time period } t, \\ 0, & \text{otherwise,} \end{cases} \tag{13}$$

Also, the problem is highly constrained, including the following types of constraints [15]:

Type I
All requirements must be fulfilled in the period in which they occur, i.e., backordering and shortage are not allowed,

$$C_I(i,j,t) = \sum_{k=1}^{t} \sum_{j} x_{ijk} \left(1-\rho_{ij}\right) - \sum_{k=1}^{t} d_{ik} \geqslant 0, \quad \forall i,t. \tag{14}$$

Type II
All orders are accompanied by the appropriate transaction cost,

$$C_{II}(i,j,t) = \left(\sum_{k=1}^{T} d_{ik} \right) y_{jt} - x_{ijt} \left(1-\rho_{ij}\right) \geqslant 0, \quad \forall i,j,t. \tag{15}$$

Type III
The total storage space is limited by W,

$$C_{III}(i,j,t) = \sum_{i} w_i \left(\sum_{k=1}^{t} \sum_{j} x_{ijk} \left(1-\rho_{ij}\right) - \sum_{k=1}^{t} d_{ik} \right) \leqslant W, \quad \forall t. \tag{16}$$

Type IV
Suppliers have limited capacities,

$$0 \leqslant x_{ijt} \leqslant k_{ij}, \quad \forall i,j,t, \tag{17}$$

where k_{ij} is the capacity of supplier j for product i.

Let I, J, and T denote the number of products, suppliers, and time periods, respectively. Then the number of model constraints is equal to,

$$M_c(I,J,T) = (I \times T) + (I \times J \times T) + T + 2 \times (I \times J \times T). \tag{18}$$

The number of decision variables x_{ijt} and y_{jt}, which defines the dimension of the optimization problem, is equal to,

$$M_v(I,J,T) = (J \times T) + (I \times J \times T). \tag{19}$$

Clearly, even for small values of I, J, and T, the corresponding problem constitutes a challenging mixed-integer optimization task due to the large number of variables and constraints.

3.2 Simplified Model

The decision parameters y_{jt} of the original model can be eliminated, thereby producing a simpler model of lower dimension. As we can infer from Eq. (13), there is dependence between y_{jt} and x_{ijt},

$$y_{jt} = \begin{cases} 1, & \text{if } x_{ijt} > 0 \text{ for at least one } i, \\ 0, & \text{otherwise.} \end{cases} \tag{20}$$

Table 1: Demand of the three products over the planning horizon of 4 periods.

	Time period			
Product	1	2	3	4
1	170	155	160	140
2	85	90	80	105
3	280	255	290	300

The variables y_{jt} are set to 1 if an order is placed on supplier j at time period t, otherwise they are set to 0. In the latter case, the quantities x_{ijt} of *all* products ordered from supplier j at period t, shall also remain equal to zero. In practice, we consider that x_{ijt} is equal to zero if it is smaller than a predefined threshold, i.e.,

$$0 < x_{ijt} \leqslant \varepsilon_z.$$

Based on these observations, we can replace the decision variables y_{jt} in Eq. (12) by their equivalent descriptions of Eq. (20), retaining only the real decision variables x_{ijt} in the model. This formulation will be henceforth called the *simplified model*. The objective function in the simplified model remains unaltered as long as we determine the parameters y_{jt} by using Eq. (20) whenever a function evaluation is conducted.

The simplified model is not a typical linear programming task, since the binary quantities are still preserved in the objective function. However, the immediate gain of excluding the binary variables y_{jt} is the remarkable reduction in the problem's dimension by a factor of $J \times T$. In both the original and the simplified model, the number and form of the constraints are preserved.

3.3 Constraints Handling

Constraints were handled through a multi-stage penalty function. Assume that,

$$\tilde{C}_s(i,j,t) = \begin{cases} |C_s(i,j,t)|, & \text{if constraint } C_s(i,j,t) \text{ is violated,} \\ \\ 0, & \text{otherwise,} \end{cases}$$

$$s \in \{I, II, III\}, \qquad i = 1,2,\ldots,I, \qquad j = 1,2,\ldots,J, \qquad t = 1,2,\ldots,T.$$

Also, let P_{pen} be a fixed positive value. Then, the employed penalty function is defined as,

$$\mathscr{F}\left(x_{ijt}, y_{jt}\right) = -f\left(x_{ijt}, y_{jt}\right) + \sum_{i,j,t,s} \tilde{C}_s(i,j,t) P_{\text{pen}}, \tag{21}$$

and depends on the degree of violation per violated constraint.

In order to avoid false penalization due to approximation errors, a violated constraint is penalized only if its value exceeds a predefined violation tolerance,

$$\tilde{C}_s(i,j,t) \geqslant \varepsilon_c > 0.$$

Also, Type IV constraints are not required to be included in the penalty function because they simply define the ranges of the variables and they can be explicitly

Table 2: Prices of products supplied by the three suppliers, and corresponding transaction costs.

	Price per supplier		
Products	1	2	3
1	25	27	24
2	30	32	33
3	54	50	49
Transaction cost	1000	900	1500

Table 3: Parameter sets for UPSO.

	UPSO parameter set		
	1	2	3
χ	0.729	0.6	0.721
c_1, c_2	2.05	2.83	1.654

handled by restricting the populations within the corresponding box constraints. If an individual violates such a constraint, it is either blocked on the violated limit or bounces back inside the search space.

4 Experimental Setting and Results

We considered the problem instance defined in [15] with $I = 3$ products, $J = 3$ suppliers, and $T = 4$ time periods. The demand and the prices of the products are reported in Tables 1 and 2, respectively. For the specific setting, the dimension of the corresponding mixed-integer original model is equal to $M_v(3,3,4) = 48$, while for the corresponding real-valued simplified model is equal to $M'_v(3,3,4) = 36$. In both cases, the total number of constraints is equal to $M_c(3,3,4) = 124$. Regarding the penalty function, the parameters $\varepsilon_c = 10^{-6}$ and $P_{\text{pen}} = 10^3$, were used as violation tolerance and fixed penalty, respectively. Also, the threshold $\varepsilon_z = 10^{-6}$ was used to identify a zero component in a solution vector.

Regarding the employed UPSO algorthm, different parameter settings were considered. Specifically, the unification factor assumed a wide range of values, i.e., $u = 0.0$ (lbest), 0.1, 0.5, 0.9, and 1.0 (gbest). The cases of $u = 0.1$, 0.5, and 0.9, were also equipped with mutation, according to Eqs. (10) and (11). Henceforth, we will denote the UPSO variants as UPSO_ℓ, $\text{UPSO}_{0.1}$, $\text{UPSO}^m_{0.1}$, $\text{UPSO}_{0.5}$, $\text{UPSO}^m_{0.5}$, $\text{UPSO}_{0.9}$, $\text{UPSO}^m_{0.9}$, and UPSO_g, where "m" denotes a variant with mutation. All mutated variants assumed a normally distributed mutation term $r_3 \sim \mathcal{N}(0,1)$. Regarding the parameters χ, c_1 and c_2, the three parameter sets reported in Table 3 were considered for all UPSO variants.

In order to tackle the binary decision variables of the original problem with the real-valued oriented UPSO, we let the corresponding particle components take real values in the range $[0,1]$, and rounded them to the nearest integer (either 0 or

Table 4: Results for all algorithms and parameter sets for the original 48-dimensional model.

Par. Set	Algor.	Suc.	Mean	St.D.	Min	Max	Mean Pen.	St.D. Pen.
1	$UPSO_{\ell}$	100.0	16537.5	2082.3	13522.9	21101.1	–	–
	$UPSO_{0.1}$	100.0	16690.2	2443.6	12796.2	21610.9	–	–
	$UPSO^{m}_{0.1}$	100.0	17362.0	2164.7	13869.9	22580.0	–	–
	$UPSO_{0.5}$	93.3	14387.1	2511.4	10675.0	19607.5	19600.0	10748.0
	$UPSO^{m}_{0.5}$	100.0	15851.5	3380.6	10920.5	23841.0	–	–
	$UPSO_{0.9}$	73.3	14129.7	2235.8	11453.9	21554.1	75287.5	66313.9
	$UPSO^{m}_{0.9}$	63.3	13831.7	1942.4	11327.6	18972.7	66445.5	73553.5
	$UPSO_{g}$	66.7	13870.6	1939.6	9679.7	17467.9	90810.0	74848.4
2	$UPSO_{\ell}$	100.0	17213.8	2405.0	12574.0	21623.8	–	–
	$UPSO_{0.1}$	100.0	17199.5	2596.6	13242.6	23383.9	–	–
	$UPSO^{m}_{0.1}$	100.0	17366.5	2518.8	13484.9	23375.9	–	–
	$UPSO_{0.5}$	100.0	14879.8	2705.9	11117.0	19945.3	–	–
	$UPSO^{m}_{0.5}$	100.0	15051.9	2152.4	12451.7	19241.0	–	–
	$UPSO_{0.9}$	86.7	14814.3	1783.2	12003.4	19445.6	158300.0	7200.0
	$UPSO^{m}_{0.9}$	70.0	15393.0	2371.4	11617.4	21042.8	80722.2	84941.4
	$UPSO_{g}$	70.0	15287.3	2216.7	12407.5	19987.9	94411.4	72475.1
3	$UPSO_{\ell}$	100.0	17540.6	2349.1	13397.4	22680.5	–	–
	$UPSO_{0.1}$	100.0	17036.8	2653.6	12866.4	23590.0	–	–
	$UPSO^{m}_{0.1}$	100.0	17561.7	2793.7	13209.9	23212.7	–	–
	$UPSO_{0.5}$	90.0	15046.6	2486.2	10036.5	20550.8	11665.9	576.7
	$UPSO^{m}_{0.5}$	100.0	15001.7	2876.9	10745.7	23334.8	–	–
	$UPSO_{0.9}$	80.0	13524.3	2237.6	10437.8	19253.1	83583.3	76711.1
	$UPSO^{m}_{0.9}$	66.7	14762.0	2226.5	10859.0	19023.8	52169.6	65860.8
	$UPSO_{g}$	73.3	13660.6	1753.1	9295.4	16265.2	71800.0	82803.4

1) during function evaluation. In the simplified model, such assumptions are not required since all the independent decision parameters are real-valued.

The swarm was randomly initialized in the search space, based on the ranges proposed in [15]. In the original model, uniform initialization within the ranges is adequate. However, in the simplified model a crucial initialization issue arises. Specifically, in the original model a binary parameter has equal probability of being initialized either to 0 or 1, since the algorithms uniformly sample real numbers within $[0,1]$. On the other hand, the simplified model samples only within the ranges of the real parameters x_{ijt}, and then computes the corresponding y_{jt} by using Eq. (20) and the relaxation parameter ε_z. Yet, this initialization almost surely assigns values $x_{ijt} > \varepsilon_z$, which correspond to $y_{jt} = 1$ because the volume (Lebesque measure) of the portion of the search space that corresponds to $x_{ijt} < \varepsilon_z$ for all i (and hence $y_{jt} = 0$) is a very small fraction of the whole search space.

Therefore, a completely random initialization in the simplified model would be heavily biased towards the values $y_{jt} = 1$ that correspond to solutions where all suppliers are getting product orders. In order to alleviate this deficiency, initializa-

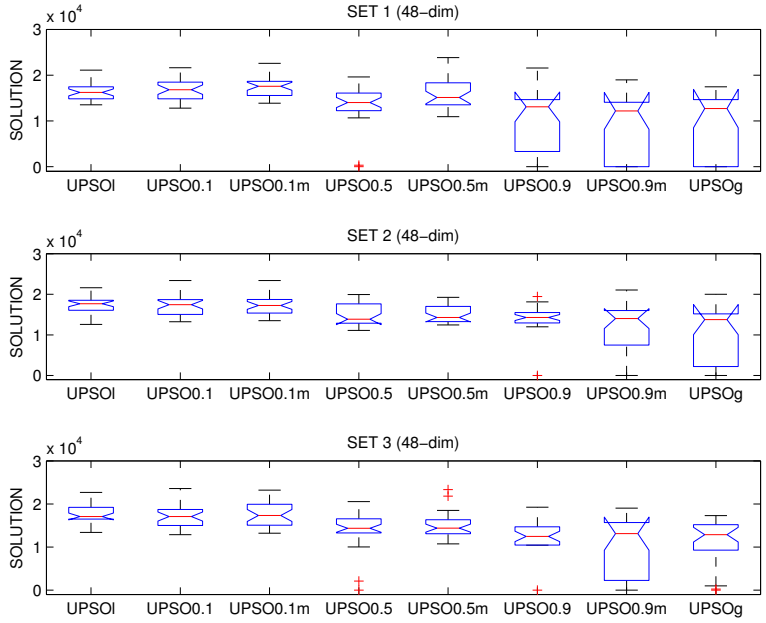

Figure 1: Boxplot of solution values per algorithm and parameter set for the 48-dimensional simplified model.

tion in the simplified model was conducted as follows,

$$x_{ijt} = \begin{cases} r_{ijt}, & \text{if } \mathscr{R}_{ijt} > 0.5, \\ 0, & \text{otherwise}, \end{cases} \qquad \forall i, j, t,$$

where $\mathscr{R}_{ijt}$ is a random value uniformly distributed in $[0,1]$. Thus, each component of the initial population had equal probability of being initialized to zero or a non-zero value.

The performance of all UPSO variants under all parameter sets was assessed on both the original and the simplified model. For each algorithm instance, 30 independent experiments were performed. At each experiment, the algorithm was allowed to perform 2×10^3 iterations using swarm size $N = 80$. The best solution detected per experiment and algorithm was recorded along with its feasibility. If a solution was infeasible, the corresponding penalty term was recorded. All experiments were conducted using the data provided in [15].

4.1 Results for the Original Model

The results for the original (48-dimensional) model are reported in Table 4 (recall that higher function values correspond to better solutions). The first two columns of the table stand for the parameter set and algorithm. The next column (labeled as "Suc.") reports the percentage of success in detecting a feasible solution in 100 experiments. The succeeding four columns report the mean, standard deviation,

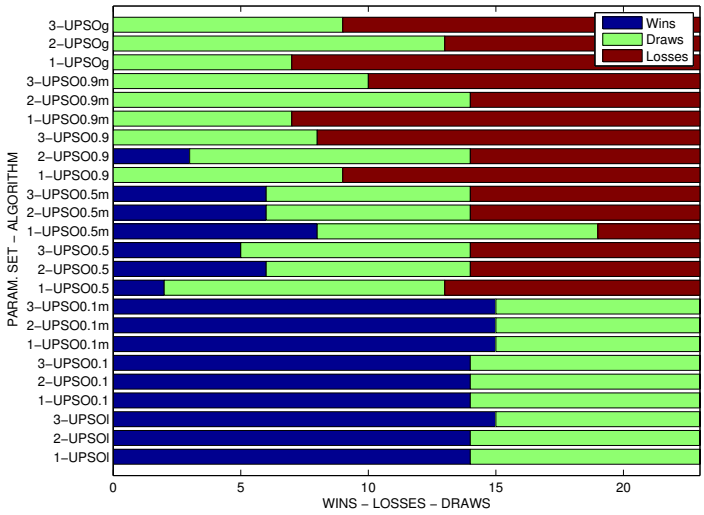

Figure 2: Number of wins, draws, and losses per algorithm for the original 48-dimensional model.

minimum, and maximum solution objective value *only for the successful runs*. For the cases where the final solution was infeasible, the last two columns report the mean and standard deviation of the corresponding penalties of the final solutions.

In order to facilitate visual comparisons between the algorithms, boxplots of the obtained solutions' objective values per algorithm and parameter set are illustrated in Fig. 1. Moreover, pairwise Wilcoxon tests were performed between the algorithms to ensure statistical significance of the observed differences. Specifically, each was parameter set-algorithm combination was compared to the rest with confidence level 95%, counting wins, draws, and losses. The corresponding results are illustrated in Fig. 2.

The reported results reveal clear trends of performance for the algorithms. Evidently, we observe superiority of exploration-oriented UPSO variants for all parameter sets. UPSO$_\ell$, UPSO$_{0.1}$, and its mutated counterpart UPSO$^m_{0.1}$, exhibited remarkable consistency, regardless of the selected parameter set. The UPSO$^m_{0.1}$ variant was also promising but with clearly inferior average performance than the previous ones. Finally, no clear correlation of specific parameter set and best performing UPSO variant was revealed by the reported values.

As a final comment for the original model, we shall mention that the best feasible solution obtained with the GA approach in [15] has objective value 15266.8. In the next section, the results for the simplified model are presented.

4.2 Results for the Simplified Model

The results for the simplified (36-dimensional) model are reported in Table 5, following the presentation motif of the previous section. A first inspection of the table offers some immediate insight. Specifically, the successful UPSO variants remarkably improved their performance, and UPSO$_{0.1}$ as well as UPSO$_\ell$ maintained their

Table 5: Results for all algorithms and parameter sets for the original 36-dimensional model.

Par. Set	Algor.	Suc.	Mean	St.D.	Min	Max	Mean Pen.	St.D. Pen.
1	$UPSO_{\ell}$	100.0	19740.2	2377.4	15377.0	23826.7	–	–
	$UPSO_{0.1}$	100.0	20275.9	1093.6	16826.7	21728.8	–	–
	$UPSO^{m}_{0.1}$	100.0	18656.1	2288.7	14235.1	21667.4	–	–
	$UPSO_{0.5}$	86.7	16974.1	3178.7	11823.5	22011.0	10878.7	1008.5
	$UPSO^{m}_{0.5}$	90.0	16910.8	2832.2	10642.3	20866.3	3092.5	1909.4
	$UPSO_{0.9}$	63.3	15648.7	3043.8	10187.5	23759.3	26509.2	33157.7
	$UPSO^{m}_{0.9}$	66.7	16359.3	3532.7	11256.8	23252.1	23805.0	39090.7
	$UPSO_{g}$	36.7	14598.8	3547.2	10104.6	20044.5	59059.8	64486.2
2	$UPSO_{\ell}$	100.0	20317.6	1672.6	17645.4	23404.3	–	–
	$UPSO_{0.1}$	100.0	20318.2	842.1	16749.2	21371.2	–	–
	$UPSO^{m}_{0.1}$	100.0	18826.2	1561.1	15360.5	21056.7	–	–
	$UPSO_{0.5}$	93.3	16723.3	3568.9	8819.5	22519.8	6.4	8.4
	$UPSO^{m}_{0.5}$	96.7	17413.8	1960.3	11744.1	20333.5	3333.6	0.0
	$UPSO_{0.9}$	86.7	17621.2	2584.2	14240.7	24970.4	26732.6	30802.5
	$UPSO^{m}_{0.9}$	83.3	15712.5	2959.3	9954.7	20984.7	46280.4	76447.3
	$UPSO_{g}$	63.3	14911.2	2357.0	10940.5	19533.3	74697.3	71821.4
3	$UPSO_{\ell}$	100.0	19658.8	1826.0	15276.6	23011.8	–	–
	$UPSO_{0.1}$	100.0	20133.3	1326.8	15799.7	21622.8	–	–
	$UPSO^{m}_{0.1}$	100.0	19961.6	934.9	16989.2	21590.2	–	–
	$UPSO_{0.5}$	86.7	15034.6	3069.5	8039.6	20023.5	9505.3	1979.8
	$UPSO^{m}_{0.5}$	76.7	13726.4	2407.2	10011.6	18131.2	20819.1	16451.2
	$UPSO_{0.9}$	50.0	16309.2	3117.9	10771.3	20199.4	48299.8	64722.9
	$UPSO^{m}_{0.9}$	66.7	15047.7	3363.0	9538.4	23043.7	35021.1	27113.3
	$UPSO_{g}$	60.0	16821.2	4249.6	8014.9	25351.7	62186.1	87439.2

top-ranking positions in terms of solution quality. Obviously, the reduced problem dimensionality along with the adequate search diversification attained by UPSO, offers properly balanced search capability to the algorithm.

Similarly to the analysis in the previous section, boxplots of the obtained solutions' objective values per algorithm and parameter set are shown in Fig. 3, while wins, draws, and losses from Wilcoxon tests are illustrated in Fig. 4. The figure reveals that, in the simplified model, exploration oriented UPSO variants achieved better scores than in the original model. This improvement can be attributed to the reduced dimension of the simplified problem. Another interesting observation is the reduced number of draws for most algorithms. This is an indication that the simplified model tends to bias the performance of different UPSO variants, rendering it more sensitive with respect to the specific problem instance.

Finally, Table 6 contains an overall ranking of all algorithms for both models with respect to their number of wins, draws, and losses in pairwise Wilcoxon ranksum tests among them based on solution quality. The ranking verifies our previous findings. The simplified model boosts the detection of high quality solutions, which are typically detected by exploration-oriented UPSO variants. The effect of the parameter set is problem-dependent. Mutated variants seem to be competi-

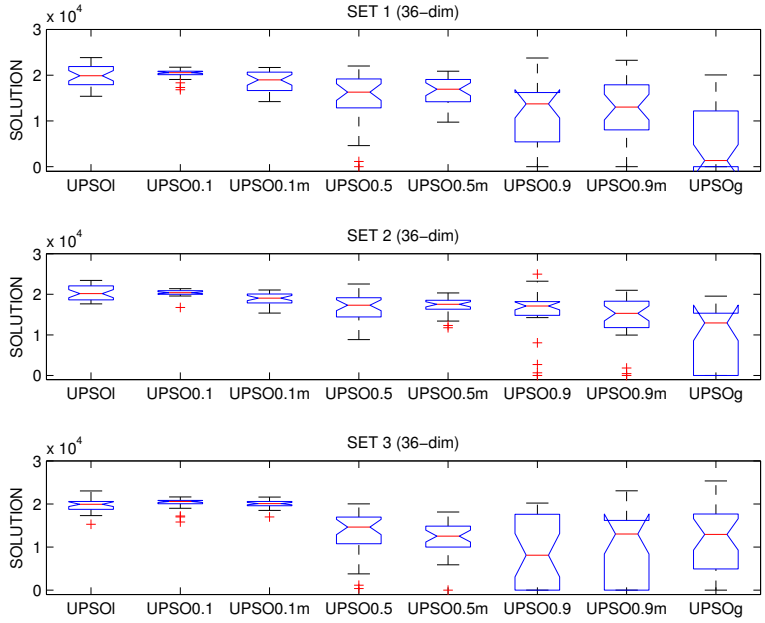

Figure 3: Boxplot of solution values per algorithm and parameter set for the 36-dimensional simplified model.

tively efficient to non-mutated ones, occupying almost half of the top 10 positions in the ranking. On the other hand, exploitation-oriented variants exhibited declining performance because of their tendency for search stagnation. This is partially due to the rounding procedure, which biases velocity towards zero in swarms that lose diversity by concentrating all particles in small regions of the search space.

5 Conclusion

We offered an experimental investigation of UPSO on a recently proposed model for supply chain with multiple items and suppliers, where the goal is the determination of an optimal procurement strategy given the demand for a finite planning horizon. The original model corresponds to a highly-constrained mixed-integer optimization task. We proposed a simplified model that reduces the problem to a real-valued optimization task. On both models, different variants of UPSO were applied and analyzed.

The obtained results suggest that the simplified model can be significantly advantageous for the successful algorithms than the original one. UPSO appeared to be highly competitive to existing GA-based approaches reported in the literature. Moreover, it was shown to be robust and less affected by the parameter set. Ongoing work includes the extension of the results in enhanced versions of the problem (e.g., including backlogging).

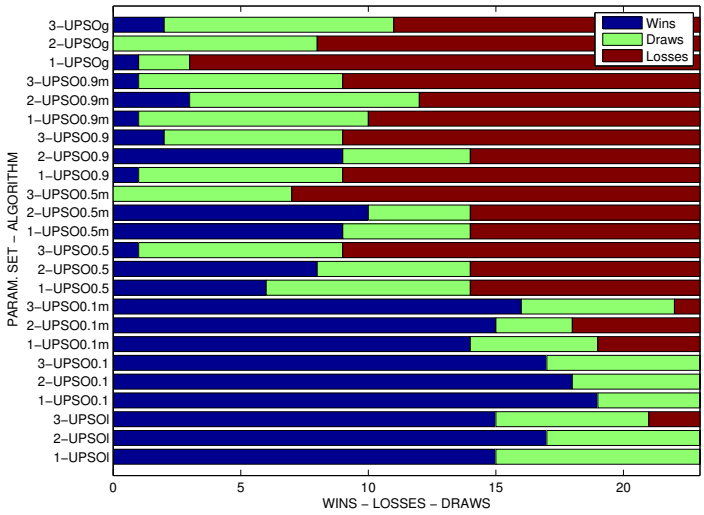

Figure 4: Number of wins, draws, and losses per algorithm for the 36-dimensional simplified model.

Acknowledgement

This research has been co-financed by the European Union (European Social Fund – ESF) and Greek national funds through the Operational Program "Education and Lifelong Learning" of the National Strategic Reference Framework (NSRF) – Research Funding Program: ARCHIMEDES III. Investing in knowledge society through the European Social Fund.

References

[1] C. Basnet and J. M. Y. Leung. Inventory lot–sizing with supplier selection. *Computers & Operations Research*, 32(1):1–14, 2005.

[2] M. Ben-Daya. Multi–stage lot sizing models with imperfect processes and inspection errors. *Production Planning and Control*, 10:118–126, 1989.

[3] H. Y. Benson. Optimal pricing and procurement strategies in a supply chain with multiple capacitated suppliers. *Computers and Operations Research*, 32:1–14, 2005.

[4] M. Clerc. Stagnation analysis in particle swarm optimization or what happens when nothing happens. Technical report, http://hal.archives-ouvertes.fr/hal-00122031, 2006.

[5] M. Clerc and J. Kennedy. The particle swarm–explosion, stability, and convergence in a multidimensional complex space. *IEEE Trans. Evol. Comput.*, 6(1):58–73, 2002.

Table 6: Overall ranking of all algorithms.

Rank	Model	Algorithm	Param. Set
1	simp	$\text{UPSO}_{0.1}$	1
2	simp	$\text{UPSO}_{0.1}$	2
3	simp	$\text{UPSO}_{0.1}$	3
4	simp	UPSO_{ℓ}	2
5	simp	$\text{UPSO}^{m}_{0.1}$	3
6	simp	UPSO_{ℓ}	1
7	simp	UPSO_{ℓ}	3
8	simp	$\text{UPSO}^{m}_{0.1}$	2
9	simp	$\text{UPSO}^{m}_{0.1}$	1
10	orig	$\text{UPSO}^{m}_{0.1}$	3
11	orig	UPSO_{ℓ}	3
12	simp	$\text{UPSO}^{m}_{0.5}$	2
13	orig	$\text{UPSO}^{m}_{0.1}$	2
14	orig	$\text{UPSO}^{m}_{0.1}$	1
15	orig	$\text{UPSO}_{0.1}$	3
16	orig	$\text{UPSO}_{0.1}$	2
17	orig	UPSO_{ℓ}	2
18	simp	$\text{UPSO}_{0.5}$	2
19	orig	$\text{UPSO}_{0.1}$	1
20	orig	UPSO_{ℓ}	1
21	simp	$\text{UPSO}^{m}_{0.5}$	1
22	simp	$\text{UPSO}_{0.9}$	2
23	simp	$\text{UPSO}_{0.5}$	1
24	orig	$\text{UPSO}^{m}_{0.5}$	1
25	orig	$\text{UPSO}^{m}_{0.5}$	2
26	orig	$\text{UPSO}^{m}_{0.5}$	3
27	orig	$\text{UPSO}_{0.5}$	2
28	simp	$\text{UPSO}^{m}_{0.9}$	2
29	simp	$\text{UPSO}_{0.5}$	3
30	orig	$\text{UPSO}_{0.5}$	3
31	orig	$\text{UPSO}_{0.9}$	2
32	orig	$\text{UPSO}_{0.5}$	1
33	simp	$\text{UPSO}^{m}_{0.5}$	3
34	orig	$\text{UPSO}_{0.9}$	3
35	simp	$\text{UPSO}^{m}_{0.9}$	1
36	simp	UPSO_{g}	3
37	simp	$\text{UPSO}_{0.9}$	1
38	orig	$\text{UPSO}^{m}_{0.9}$	2
39	orig	UPSO_{g}	2
40	simp	$\text{UPSO}^{m}_{0.9}$	3
41	orig	$\text{UPSO}^{m}_{0.9}$	3
42	orig	UPSO_{g}	3
43	orig	$\text{UPSO}_{0.9}$	1
44	simp	UPSO_{g}	2
45	simp	$\text{UPSO}_{0.9}$	3
46	orig	UPSO_{g}	1
47	orig	$\text{UPSO}^{m}_{0.9}$	1
48	simp	UPSO_{g}	1

[6] K.-N. Francis Leung. A generalized geometric–programming solution to an economic production quantity model with flexibility and reliability considerations. *European Journal of Operational Research*, 176(1):240–251, 2007.

[7] E. Hassini. Order lot sizing with multiple capacitated suppliers offering leadtime–dependent capacity reservation and unit price discounts. *Production Planning and Control*, 19(2):142–149, 2008.

[8] P. A. Hayek and M. K. Salameh. Production lot sizing with the reworking of imperfect quality items produced. *Production Planning and Control*, 12:584–590, 2001.

[9] J. Kennedy and R. C. Eberhart. Particle swarm optimization. In *Proc. IEEE Int. Conf. Neural Networks*, volume IV, pages 1942–1948, Piscataway, NJ, 1995. IEEE Service Center.

[10] S. Papachristos and I. Konstantaras. Economic ordering quantity models for items with imperfect quality. *International Journal of Production Economics*, 100(1):148–154, 2006.

[11] K. Parsopoulos, K. Skouri, and M. Vrahatis. Particle swarm optimization for tackling continuous review inventory models. In *Lecture Notes in Computer Science (LNCS)*, volume 4974, pages 103–112. Springer, 2008.

[12] K. E. Parsopoulos and M. N. Vrahatis. UPSO: A unified particle swarm optimization scheme. In *Lecture Series on Computer and Computational Sciences, Vol. 1, Proceedings of the International Conference of Computational Methods in Sciences and Engineering (ICCMSE 2004)*, pages 868–873. VSP International Science Publishers, Zeist, The Netherlands, 2004.

[13] K. E. Parsopoulos and M. N. Vrahatis. Parameter selection and adaptation in unified particle swarm optimization. *Mathematical and Computer Modelling*, 46(1–2):198–213, 2007.

[14] K. E. Parsopoulos and M. N. Vrahatis. *Particle Swarm Optimization and Intelligence: Advances and Applications*. Information Science Publishing (IGI Global), 2010.

[15] J. Rezaei and M. Davoodi. A deterministic, multi–item inventory model with supplier selection and imperfect quality. *Applied Mathematical Modelling*, 32(10):2106–2116, 2008.

[16] M. J. Rosenblat and H. L. Lee. Economic production cycles with imperfect production processes. *IIE Transactions*, 18:48–55, 1986.

[17] M. K. Salameh and M. Y. Jaber. Economic production quantity model for items with imperfect quality. *International Journal of Production Economics*, 64(1–3):59–64, 2000.

[18] I. C. Trelea. The particle swarm optimization algorithm: Convergence analysis and parameter selection. *Information Processing Letters*, 85:317–325, 2003.