XI Balkan Conference on Operational Research
BALCOR 2013
Belgrade, Zlatibor, 07-10 September 2013
UNIVERSITY OF BELGRADE
FACULTY OF ORGANISATIONAL SCIENCES

A FIRST STUDY OF PARTICLE SWARM OPTIMIZATION ON THE DYNAMIC LOT SIZING PROBLEM WITH PRODUCT RETURNS

Eirini Moustaki[1], Konstantinos E. Parsopoulos[1], Ioannis Konstantaras[2], Konstantina Skouri[3], Ioannis Ganas[4]

[1] Department of Computer Science & Engineering, University of Ioannina, GR-45110 Ioannina, Greece, {emoustak, kostasp}@cs.uoi.gr

[2] Department of Business Administration, University of Macedonia, 156 Egnatia Str., GR-54006 Thessaloniki, Greece, ikonst@uom.gr

[3] Department of Mathematics, University of Ioannina, GR-45110 Ioannina, Greece, kskouri@uoi.gr

[4] Department of Accounting, Technological Educational Institute of Epirus, Psathaki, PO Box 169, GR-48100 Preveza, Greece, ganas@teiep.gr

Abstract: *We study the behavior of a popular metaheuristic optimization algorithm, namely Particle Swarm Optimization (PSO), on the single-item dynamic lot sizing problem with returns and remanufacturing. The most suitable variants of the algorithm are identified and applied after the necessary modifications. The performance of the algorithm is assessed on an extensive test suite employed in previous studies. Its performance is compared with that of the adapted Silver-Meal algorithm as well as with its recently enhanced versions. The results suggest that PSO is very competitive and can be considered as a promising alternative for solving the considered problems.*

Keywords: *Lot sizing, Inventory, Remanufacturing, Particle Swarm Optimization*

1. INTRODUCTION

In recent years, manufacturers have paid growing attention to reuse activities that provide material waste reduction via the recovery of some content of used products. Motivation behind these product recovery activities is two-fold: growing environmental concerns and potential economical benefits. In several countries environmental regulations are in place, rendering manufacturers responsible for the whole life cycle of the product they produce. A common example of these regulations is take-back obligations after usage [7]. Even in the absence of such regulations, the expectations of environmentally conscious consumers put pressure on companies to consider environmental issues in their manufacturing process. Nowadays, a green image, which can be obtained by implementing recoverable manufacturing systems, has become a powerful marketing tool and provides a significant competitive advantage to companies that seek to have a place in the global market.

Remanufacturing is a typical example for economically attractive reuse activities by transforming used products into like-new ones. After disassembly, modules and parts are extensively inspected. Problematic parts are repaired or, if not possible, replaced with new ones. These operations allow a considerable amount of value to be incorporated in the used product to be regained. Remanufactured products have usually the same quality as the new products and are sold for the same price, but they are less costly. Examples of remanufacturable products include mostly high-value components such as aircraft or automobile engines, aviation equipment, medical equipment, office furniture, machine tools, copiers, computers, electronics equipment, toner cartridges, cellular telephones, and single-use cameras, among others [9, 21].

Dynamic or *Economic Lot Sizing* (ELS), i.e., planning manufacturing/production orders over a number of future periods in which demand is dynamic and deterministic, is one of the most extensively researched topics in production and inventory control. However, the ELS problem with Remanufacturing options (ELSR), as an alternative for manufacturing, has received quite a bit of attention in the reverse logistics literature. The ELSR problem can be described as follows: in every period over a finite, discrete time horizon, a retailer faces a deterministic and dynamic demand for a single type of product, and receives a deterministic amount of returned used items. In order to meet demand, the retailer can either place an order for newly manufactured items or send some of its returned used product to be remanufactured. The retailer maintains separate inventories for the serviceable product and the returned one. When ordering newly manufactured product or remanufactured product, the retailer incurs a fixed setup cost. In addition, in each period the retailer incurs holding costs for storing the serviceable and the returned product in inventory.

Various different variants of the ELSR problem have been studied. In [17] the classical Wagner-Whitin model [23] is extended by introducing a remanufacturing process. The authors showed that there exists an

optimal solution that is a zero-inventory policy, and gave a dynamic programming algorithm to determine the periods where products are manufactured and remanufactured. Golany et al. [8] considered a variant of ELSR with disposal of returned used products at a cost, and showed that this problem is NP complete under general concave production and holding costs. Pineyro and Viera [14] studied the ELSR problem with fixed production and disposal setup costs, as well as with linear production, holding, and disposal costs. Also, they proposed a Tabu Search procedure with the aim of finding a near-optimal solution.

Teunter et al. [20] studied ELSR with separate as well as joint setup for manufacturing and remanufacturing. For the case of joint setup cost, they provided an exact polynomial-time dynamic programming algorithm. They also studied and compared the computational performance of modified versions of three well-known heuristics, namely *Silver-Meal* (SM), Least Unit Cost, and Part Period Balancing, for the separate and joint setup cost cases. Schulz [18] proposed a generalization of the SM-based heuristic introduced by Teunter et al. [20] for the separate setup cost case. The enhanced SM variants exhibited significantly better performance in terms of the average percentage error to the optimal solution.

Recent works on the Wagner-Whitin and relevant inventory optimization problems [15, 16], have shown the potential of effectively solving these problems by using modern population-based optimization algorithms. Although the studied algorithms are primarily designed for real-valued optimization, proper modifications of their operation as well as of the problem's formulation can render them applicable also on integer and mixed-integer problems, as the one under consideration.

The reported nice performance triggered our interest in studying their behavior also on the ELSR problem. Specifically, we selected the highly promising and popular *Particle Swarm Optimization* (PSO) algorithm and assessed its performance on the test suite used by Schulz [18], comparing its performance with the reported one for the SM-based variants. The aim of the study was to probe the potential of PSO to serve as promising alternative for tackling the ELSR problem, enriching the algorithmic artillery for this type of problems.

The rest of the paper is organized as follows: Section 2 constitutes the basic formulation of the problem, while the PSO algorithm is described in Section 3. Section 4 reports the obtained results and the paper concludes with Section 5.

2. ORIGINAL MODEL FORMULATION

The original problem considered in our study consists of the dynamic lot sizing model with both remanufacturing and manufacturing setup costs, as it was introduced by Teunter et al. [20] and studied by Schulz [18]. This problem emerged as an extension of the original Wagner-Whitin problem [23] and considers a manufacturer that produces a single product over a finite planning horizon. At each time period, there is a known demand for the product as well as a number of returned items that can be completely remanufactured and sold as new. If the remanufactured items that are stored in inventory are inadequate to satisfy the demand, an additional number of items is manufactured. The aim is to determine the exact number of remanufactured and manufactured items per time period, in order to minimize the total holding and setup cost under various operational constraints.

In order to formally describe the considered model, we will henceforth use the following notation that closely follows the presentation of Schulz [18]:

- t: time period, $t = 1, 2, \ldots, T$.
- D_t: demand for time period t.
- R_t: number of returned items in period t that can be completely remanufactured and sold as new.
- h^R: holding cost for the recoverable items per unit time.
- h^M: holding cost for the manufactured items per unit time.
- z_t^R: number of items that are eventually remanufactured in period t.
- z_t^M: number of manufactured items in period t.
- K^R: remanufacturing setup cost.
- K^M: manufacturing setup cost.
- y_t^R: inventory level of items that can be remanufactured in period t.
- y_t^M: inventory level of ready-to-ship items in period t.

Now we can define the main cost optimization problem as follows [18]:

$$\min \quad C = \sum_{t=1}^{T} \left(K^R \gamma_t^R + K^M \gamma_t^M + h^R y_t^R + h^M y_t^M \right), \tag{1}$$

where:

$$\gamma_t^R = \begin{cases} 1, & \text{if } z_t^R > 0, \\ 0, & \text{otherwise}, \end{cases} \qquad \gamma_t^M = \begin{cases} 1, & \text{if } z_t^M > 0, \\ 0, & \text{otherwise}, \end{cases} \tag{2}$$

are binary decision variables denoting the initiation of a remanufacturing or manufacturing lot, respectively. Naturally, the model is accompanied by a number of constraints [18]:

$$y_t^R = y_{t-1}^R + R_t - z_t^R, \qquad t = 1,2,\ldots,T, \tag{3}$$

$$y_t^M = y_{t-1}^M + z_t^R + z_t^M - D_t, \qquad t = 1,2,\ldots,T, \tag{4}$$

$$z_t^R \leqslant Q\gamma_t^R, \qquad t = 1,2,\ldots,T, \tag{5}$$

$$z_t^M \leqslant Q\gamma_t^M, \qquad t = 1,2,\ldots,T, \tag{6}$$

$$y_0^R = y_0^M = 0, \tag{7}$$

$$\gamma_t^R, \gamma_t^M \in \{0,1\}, \qquad y_t^R, y_t^M, z_t^R, z_t^M \geqslant 0, \qquad t = 1,2,\ldots,T. \tag{8}$$

The constraints defined in Eqs. (3) and (4) guarantee the inventory balance taking into consideration the incoming and outcoming items. Equations (5) and (6) guarantee that fixed costs are introduced whenever a new lot is initiated. The parameter Q is a sufficiently large number; Schulz [18] suggests the use of the total demand during the planning horizon. Finally, Eqs. (7) and (8) ensure that the inventories are initially empty, as well as that all parameters assume only reasonable values.

Teunter et al. [20] have identified some interesting properties of the considered model. Firstly, there is a possibility of attaining optimal solutions that do not adhere to the zero-inventory property. Secondly, although theoretically the (mixed-integer) problem can be solved to optimality, there is strong evidence that it is NP hard. This conjecture was stated by Teunter et al. [20] and strengthened by the findings in [18], offering motivation for the use of heuristic algorithms such as the adapted Silver-Meal and its improvements proposed in these sources. Also, it triggered our interest in tackling the problem with the metaheuristic algorithm described in the following section.

3. PARTICLE SWARM OPTIMIZATION

Particle Swarm Optimization (PSO) is a stochastic, population-based metaheuristic algorithm. It was introduced by Eberhart and Kennedy [5] as an alternative to the dominant Evolutionary Algorithms (EAs) [1] for solving numerical optimization problems. Since its development, PSO has gained increasing scientific interest. This is attributed to its verified efficiency in a plethora of challenging optimization problems, as well as to its easy implementation. Today, PSO is placed in a salient position among the state-of-the-art of metaheuristic optimization algorithms, counting a large number of applications in diverse scientific fields [2, 12] as well as an extensive bibliography [6, 3, 10, 13].

The main search engine of PSO consists of a group of cooperative search agents that iteratively probe the search space. Putting it formally, consider the n-dimensional global optimization problem:

$$\min_{x \in X \subset \mathbb{R}^n} C(x).$$

and let the set of indices $I = \{1,2,\ldots,N\}$. A *swarm* S of *size* N, is a set of search points defined as:

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

where each search point, also called a *particle*, is an n-dimensional vector defined as:

$$x_i = (x_{i1}, x_{i2}, \ldots, x_{in})^T \in X, \qquad i \in I.$$

Each search point is allowed to move within the search space. For this purpose, an adaptable position shift, also called the *velocity* of the particle, is used and defined as follows:

$$v_i = (v_{i1}, v_{i2}, \ldots, v_{in})^T, \qquad i \in I.$$

Moreover, each particle retains in memory the *best position* it has ever visited in the search space, i.e., the position with the lowest objective value:

$$p_i = (p_{i1}, p_{i2}, \ldots, p_{in})^T, \qquad i \in I.$$

If t denotes the algorithm's iteration counter (it shall not be confused with the time period counter used in Section 2), then it holds that:

$$p_i(t) = x_i(\tau),$$

where:

$$\tau = \arg \min_{\kappa \in \{0,1,\ldots,t\}} \Big\{ C\big(x_i(\kappa)\big) \Big\}.$$

The best positions constitute a sort of *experience* for the particles. Sharing this experience produces cooperation among the particles, guiding their search towards the most promising regions of the search space.

The information-sharing is based on the concept of *neighborhood* [19]. A neighborhood of the i-th particle is defined as a set of the indices of all the particles with which, it exchanges information:

$$NB_{i,s} = \{j_1, j_2, \ldots, j_s\} \subseteq I,$$

while it holds that $i \in NB_{i,s}$. Thus, the neighborhood's best position, p_{g_i}, where:

$$g_i = \arg \min_{j \in NB_{i,s}} \Big\{ C(p_j) \Big\},$$

is used along with p_i to update the i-th particle at each iteration.

The user-defined parameter s, also called the *neighborhood size*, has direct impact on the magnitude of information-sharing among the particles. Hence, it can affect the exploration/exploitation properties of the algorithm. In the special case where $s = N$, the whole swarm constitutes the neighborhood of each particle. This case defines the so called *global PSO model*, usually denoted as *gbest*. On the other hand, strictly smaller neighborhoods define *local PSO models*, denoted as *lbest*.

The particles that will constitute each neighborhood are usually derived from specific abstract schemes that assume a spatial organization of the particles' indices. Such schemes are called *neighborhood topologies* and they can have crucial impact on PSO's performance because they determine the information flow among the particles. A widely used neighborhood topology is the *ring* [11], where all particles' indices are assumed to be organized on a ring in ascending order, assuming that indices recycle after N. According to this topology, each particle assumes as immediate neighbors the particles with its adjacent indices.

Up-to-date there are various PSO variants proposed in the literature. One of the most popular is the *constriction coefficient PSO* introduced by Clerc and Kennedy [4]. According to it, the particles' motion is dictated by the following equations:

$$v_{ij}(t+1) = \chi\Big[v_{ij}(t) + c_1 \mathcal{R}_1 \Big(p_{ij}(t) - x_{ij}(t)\Big) + c_2 \mathcal{R}_2 \Big(p_{g_i,j}(t) - x_{ij}(t)\Big)\Big], \tag{9}$$

$$x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t+1), \tag{10}$$

where, $i = 1,2,\ldots,N$; $j = 1,2,\ldots,n$; the parameter χ is the *constriction coefficient*; c_1 and c_2 are constants also called the *cognitive* and *social* parameter, respectively; and $\mathcal{R}_1$, $\mathcal{R}_2$, are random variables uniformly distributed

Table 1: Parameter values for the test problems provided by Schulz [18].

Parameter Description	Value(s)
Setup costs	$K^M, K^R \in \{200, 500, 2000\}$
Holding cost for ready-to-ship products	$h^M = 1$
Holding cost for ready-to-ship products	$h^R \in \{0.2, 0.5, 0.8\}$
Demand for time period t	$D_t \sim \mathcal{N}\left(\mu_D, \sigma_D^2\right)$, $\mu_D = 100$, $\sigma_D^2 = 10\%, 20\%$ of μ_D (10% small variance, 20% large variance)
Returns for time period t	$R_t \sim \mathcal{N}\left(\mu_R, \sigma_R^2\right)$, $\mu_R = 30\%, 50\%, 70\%$ of μ_D, $\sigma_R^2 = 10\%, 20\%$ of μ_R (10% small variance, 20% large variance)

Table 2: Parameter values for the PSO algorithm.

Parameter Description	Value(s)
Number of experiments per problem instance	30
Maximum number of function evaluations	10^8
PSO model and neighborhood topology	lbest model with ring topology of radius $r = 1$ (gbest model also tested but with inferior results)
PSO restart frequency	10^4 iterations

in the range $[0, 1]$. It shall be noted that a different value of $\mathcal{R}_1$ and $\mathcal{R}_2$ is sampled for each i and j in Eq. (9) per iteration.

The best positions of the particles are also updated at each iteration, as follows:

$$p_i(t+1) = \begin{cases} x_i(t+1), & \text{if } C\Big(x_i(t+1)\Big) < C\Big(p_i(t)\Big), \\ p_i(t), & \text{otherwise,} \end{cases} \qquad i \in I. \tag{11}$$

This variant of PSO is supported by thorough theoretical analysis [4]. The analysis implied the default general-purpose parameter values $\chi = 0.729$, $c_1 = c_2 = 2.05$. This is considered to be a satisfactory setting that produces balanced convergence speed for the algorithm. Nevertheless, alternative settings have also been studied by Trelea [22].

Although PSO was initially proposed for continuous optimization problems, it has been applied also on integer and mixed-integer problems. The simplest yet frequently adequate modification that is required is the rounding of the real values to the closest integer for the integer variables of the problem. Also, the absolute minimum value of each velocity component of the particles can be set to 0.5 for these variables. This promotes the exploration properties of the particles, since absolute velocity values lower than 0.5 result in rounding at the same integer value. Further applications of PSO on integer and discrete optimization problems can be found in [13].

4. EXPERIMENTAL RESULTS

The PSO algorithm was applied on the test suite used by Schulz in [18], which is an extended version of the one in [20]. It consists of a full factorial study of various problem instances with a common planning horizon of $T = 12$ time periods. The setup costs, K^M and K^R, as well as the holding cost for the recoverable items, h^R, assume three different values each. The demands and returns are drawn from normal distributions with both large and small deviations. The mean of the returns' distribution assumes also three different values (return ratios). The exact configuration of the test problems is reported in Table 1.

For each specific combination of parameter values, 20 different problem instances were produced by Schulz [18]. The specific test suite was selected in our study because it contained a large number of 6480 different problem instances. Also, it facilitated comparisons with the results reported in [18] for the adapted SM algorithm and its enhanced versions, in order to gain better insight of the PSO's performance.

PSO was applied with the default parameter set and the lbest model, although the gbest model was also considered but exhibited inferior performance. It shall be noticed that the selected PSO variant promotes exploration rather than exploitation. This is a premature yet clear indication that success is related to the exploration property of the algorithm. All the parameter values of PSO are summarized in Table 2.

Table 3: Percentage cost error for all instances as well as for different variance of demand, returns, and return ratios.

		Algorithm	Average	St. Dev.	Maximum
All Instances		SM_2	7.5%	7.9%	49.2%
		SM_4	6.1%	7.6%	47.3%
		SM_2^+	6.9%	7.9%	49.2%
		SM_4^+	2.2%	2.9%	24.3%
		PSO	4.3%	4.5%	49.8%
Demand	Small	SM_2	7.2%	7.9%	43.6%
	Variance	SM_4	6.0%	7.6%	47.3%
		SM_2^+	6.6%	7.9%	43.5%
		SM_4^+	2.1%	2.8%	18.9%
		PSO	4.4%	4.6%	49.8%
	Large	SM_2	7.8%	8.0%	49.2%
	Variance	SM_4	6.1%	7.5%	43.9%
		SM_2^+	7.2%	8.0%	49.2%
		SM_4^+	2.4%	3.0%	24.3%
		PSO	4.1%	4.5%	48.3%
Returns	Small	SM_2	7.3%	7.8%	47.2%
	Variance	SM_4	6.1%	7.6%	47.3%
		SM_2^+	6.8%	7.8%	47.2%
		SM_4^+	2.2%	2.9%	21.1%
		PSO	4.3%	4.6%	46.7%
	Large	SM_2	7.7%	8.0%	49.2%
	Variance	SM_4	6.1%	7.5%	46.3%
		SM_2^+	7.1%	8.0%	49.2%
		SM_4^+	2.3%	2.9%	24.3%
		PSO	4.2%	4.5%	49.8%
Return Ratio	30%	SM_2	5.5%	5.5%	31.3%
		SM_4	3.7%	4.5%	28.5%
		SM_2^+	4.9%	5.4%	31.3%
		SM_4^+	1.2%	1.8%	12.1%
		PSO	3.5%	3.1%	45.5%
	50%	SM_2	8.5%	9.4%	40.1%
		SM_4	7.3%	8.2%	41.8%
		SM_2^+	8.0%	9.3%	39.8%
		SM_4^+	2.3%	2.7%	16.2%
		PSO	4.1%	4.0%	34.0%
	70%	SM_2	8.4%	8.0%	49.2%
		SM_4	7.2%	8.7%	47.3%
		SM_2^+	8.0%	8.0%	49.2%
		SM_4^+	3.3%	3.5%	24.3%
		PSO	5.1%	5.9%	49.8%

For each problem instance, 30 independent experiments of PSO were conducted resulting in a total number of $6480 \times 30 = 194400$ independent experiments. The optimal value per problem instance was known and, naturally, the main optimization goal was to achieve the lowest possible percentage error from the optimal solution. The algorithm was always initialized with uniformly distributed random swarms (populations). Each experiment was terminated as soon as the optimal solution was found or a fixed computational budget (maximum number of function evaluations) was reached. Then, the final solution's percentage error was recorded and, after the end of the 30 experiments, the sample of these values was statistically analyzed with respect to its mean, standard deviation, and maximum value.

The obtained statistics were compared to the corresponding values reported in the thorough analysis of Schulz [18] for four versions of the adapted SM heuristic. The first version refers to SM with the options of (a) manufacture only or (b) remanufacture (and manufacture if necessary), henceforth denoted as SM2.The second version refers to SM2 with the additional options of (c) manufacture first and remanufacture later or (d) remanufacture first and manufacture later, henceforth denoted as SM4. Moreover, comparisons included the enhanced versions SM2+ and SM4+ that are derived from the previous ones by additionally checking if their solutions admit one of the following improvements: (i) two consecutive time windows can be combined or (ii) a remanufacturing lot can be increased [18].

All numerical results are reported in Tables 3-6. Specifically, Table 3 reports the average, standard deviation,

Table 4: Percentage cost error for different levels of manufacturing setup cost.

	Algorithm	Average	St. Dev.	Maximum
K^M=200	SM_2	4.3%	4.5%	20.2%
	SM_4	3.4%	3.6%	17.6%
	SM_2^+	3.5%	4.0%	20.2%
	SM_4^+	2.3%	2.6%	13.5%
	PSO	4.0%	3.1%	45.5%
K^M=500	SM_2	5.4%	5.2%	25.1%
	SM_4	3.9%	3.9%	19.3%
	SM_2^+	4.8%	4.9%	23.7%
	SM_4^+	2.1%	2.5%	12.8%
	PSO	4.5%	4.1%	27.5%
K^M=2000	SM_2	12.8%	9.9%	49.2%
	SM_4	10.9%	10.4%	47.3%
	SM_2^+	12.6%	9.9%	49.2%
	SM_4^+	2.3%	3.4%	24.3%
	PSO	4.4%	5.9%	49.8%

Table 5: Percentage cost error for different levels of remanufacturing setup cost.

	Algorithm	Average	St. Dev.	Maximum
K^R=200	SM_2	10.9%	9.1%	49.2%
	SM_4	6.6%	7.8%	40.2%
	SM_2^+	10.0%	9.4%	49.2%
	SM_4^+	1.9%	2.1%	11.8%
	PSO	5.7%	5.5%	49.8%
K^R=500	SM_2	7.9%	6.6%	34.7%
	SM_4	8.1%	8.2%	47.3%
	SM_2^+	7.3%	6.6%	34.7%
	SM_4^+	3.4%	3.2%	19.1%
	PSO	3.8%	4.1%	37.4%
K^R=2000	SM_2	3.7%	6.0%	29.4%
	SM_4	3.5%	5.7%	25.7%
	SM_2^+	3.6%	5.9%	29.4%
	SM_4^+	1.4%	2.9%	24.3%
	PSO	3.3%	3.5%	45.5%

and maximum values of the percentage errors for the aforementioned variants of the SM algorithm as well as for PSO. The first block of the table refers to all problem instances, followed by three blocks that refer to the special cases of small and large variance for demand and returns, as well as to different return ratios. Tables 4-6 complement the results for different values of the manufacturing and remanufacturing setup cost, as well as for different holding costs of the recoverable items.

A first inspection of the results clearly reveals that PSO is very competitive to the SM variants with respect to their average percentage errors and standard deviations. Indeed, in all cases reported in Table 3, PSO attained lower averages than the SM variants, with the exception of the SM4+ approach. Undoubtedly, the results offer strong evidence that PSO can be considered as promising alternatives for solving the considered problems.

Regarding the different test cases, the values of K^M and K^R seemed to affect the relative performance of the algorithms more than the rest of the parameters. In fact, PSO exhibited their largest standard deviations for $K^M = 2000$, as we can see in Tables 4 and 5. On the other hand, the variance of demand and returns, the different return ratios as well as the value of the holding cost h^R, were all accompanied by identical relative ordering of the algorithm's performance.

Regarding the time-complexity of the proposed algorithm, it ranged from a few seconds (in the cases where the optimal solution was attained) up to 2 to 3 minutes for the cases where the number of function evaluations was exceeded. The reported times are indicative, since they heavily depend on the implementation, the hardware, and the machine's load at the time of execution. In our case, no effort was paid to optimize the running time of the algorithm. All times refer to execution on Ubuntu linux servers with Intel® Core™ i7 processors, 8GB RAM, occupying all the available cores.

Table 6: Percentage cost error for different levels of holding cost.

	Algorithm	Average	St. Dev.	Maximum
h^R=0.2	SM_2	5.9%	8.0%	42.9%
	SM_4	5.3%	8.0%	47.3%
	SM_2^+	5.8%	8.0%	42.9%
	SM_4^+	1.7%	2.5%	21.1%
	PSO	4.5%	5.2%	49.8%
h^R=0.5	SM_2	7.5%	7.7%	49.2%
	SM_4	6.5%	7.6%	42.4%
	SM_2^+	7.0%	7.7%	49.2%
	SM_4^+	2.3%	3.0%	24.3%
	PSO	4.3%	4.5%	45.5%
h^R=0.8	SM_2	9.1%	7.7%	44.4%
	SM_4	6.3%	7.0%	40.3%
	SM_2^+	8.1%	7.8%	44.4%
	SM_4^+	2.8%	3.0%	20.6%
	PSO	4.0%	3.9%	42.9%

5. CONCLUSIONS

We studied the behavior of a popular metaheuristic optimization algorithm, namely PSO, on the single-item dynamic lot sizing problem with returns and remanufacturing. The algorithm was tested on a large number of problems previously used in relevant studies. Its performance was compared with two variants of the adapted SM algorithm, namely SM2 and SM4, as well as with their recently proposed enhanced versions, SM2+ and SM4+.

Preliminary experiments identified the PSO model with the most effective parameter set. The selected approach was thoroughly analyzed on the test problems after introducing the necessary modifications in the formulation of the corresponding optimization problem and its structure. The results suggested that PSO can be considered as promising alternative for solving such problems, verifying its nice performance reported in previous works for similar lot sizing problems.

Future work will contribute towards the direction of developing more refined versions of PSO in order to further enhance its performance on the specific problem type. Specialized operators may offer the desirable performance boost. Also, different metaheuristics will be considered for the specific type of problems.

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.
The authors would like to express their deep appreciation to Dr. T. Schulz for providing the complete test set that was used in the experimental part of the paper.

REFERENCES

[**1**] Back, T., Fogel, D., and Michalewicz, Z., (1997). Handbook of Evolutionary Computation. New York: IOP Publishing and Oxford University Press.

[**2**] Banks, A., Vincent, J., and Anyakoha, C., (2008). A review of particle swarm optimization. Part II: hybridisation, combinatorial, multicriteria and constrained optimization, and indicative applications. Natural Computing, 7 (1), 109–124.

[**3**] Clerc, M., (2006). Particle Swarm Optimization. ISTE Ltd.

[**4**] Clerc, M. and Kennedy, J., (2002). The Particle Swarm-Explosion, Stability, and Convergence in a Multidimensional Complex Space. IEEE Trans. Evol. Comput., 6 (1), 58–73.

[**5**] Eberhart, R.C. and Kennedy, J., (1995). A New Optimizer Using Particle Swarm Theory. *In*:Proceedings Sixth Symposium on Micro Machine and Human Science, Piscataway, NJ: IEEE Service Center, 39–43.

[**6**] Engelbrecht, A.P., (2006). Fundamentals of Computational Swarm Intelligence. Wiley.

[**7**] Fleischmann, M., et al., (1997). Quantitative models for reverse logistics: A review. European Journal of Operational Research, 103 (1), 1–17.
[**8**] Golany, B., Yang, J., and Yu, G., (2001). Economic lot-sizing with remanufacturing options. IIE Transactions, 33 (11), 995–1003.
[**9**] Guide Jr, V.D.R., Jayaraman, V., and Srivastava, R., (1999). Production planning and control for remanufacturing: a state-of-the-art survey. Robotics and Computer Integrated Manufacturing, 15 (3), 221–230.
[**10**] Kennedy, J. and Eberhart, R.C., (2001). Swarm Intelligence. Morgan Kaufmann Publishers.
[**11**] Li, X., (2010). Niching Without Niching Parameters: Particle Swarm Optimization Using a Ring Topology. IEEE Transactions on Evolutionary Computation, 14 (1), 150–169.
[**12**] Parsopoulos, K.E. and Vrahatis, M.N., (2002). Recent Approaches to Global Optimization Problems Through Particle Swarm Optimization. Natural Computing, 1 (2-3), 235–306.
[**13**] Parsopoulos, K.E. and Vrahatis, M.N., (2010). Particle Swarm Optimization and Intelligence: Advances and Applications. Information Science Publishing (IGI Global).
[**14**] Pineyro, P. and Viera, O., (2009). Inventory policies for the economic lot-sizing problem with remanufacturing and final disposal options. Journal of Industrial and Management Optimization, 5 (2), 217–238.
[**15**] Piperagkas, G.S., et al., (2012). Solving the Stochastic Dynamic Lot-Sizing Problem Through Nature-Inspired Heuristics. Computers & Operations Research, 39 (7), 1555–1565.
[**16**] Piperagkas, G.S., et al., (2011). Applying PSO and DE on Multi-Item Inventory Problem with Supplier Selection. In: The 9th Metaheuristics International Conference (MIC 2011), Udine, Italy, 359–368.
[**17**] Richter, K. and Sombrutzki, M., (2000). Remanufacturing planning for the reverse Wagner/Whitin models. European Journal of Operational Research, 121 (2), 304–315.
[**18**] Schulz, T., (2011). A New Silver-Meal Based Heuristic for the Single-Item Dynamic Lot Sizing Problem with Returns and Remanufacturing. International Journal of Production Research, 49 (9), 2519–2533.
[**19**] Suganthan, P.N., (1999). Particle Swarm Optimizer with Neighborhood Operator. In: Proc. IEEE Congr. Evol. Comput., Washington, D.C., USA, 1958–1961.
[**20**] Teunter, R.H., Bayindir, Z.P., and Van den Heuvel, W., (2006). Dynamic Lot Sizing with Product Returns and Remanufacturing. International Journal of Production Research, 44 (20), 4377–4400.
[**21**] Thierry, M.C., et al., (1995). Strategic issues in product recovery management. California Management Review, 37 (2), 114–135.
[**22**] Trelea, I.C., (2003). The Particle Swarm Optimization Algorithm: Convergence Analysis and Parameter Selection. Information Processing Letters, 85, 317–325.
[**23**] Wagner, H.M. and Whitin, T.M., (1958). Dynamic version of the economic lot size model. Management Science, 5 (1), 88–96.