A Binary Particle Swarm Optimization (BPSO) algorithm to solve a reverse logistics supply chain problem

Ernesto Del R. Santibanez-Gonzalez, Jairo R. Montoya-Torres, Luisa Huaccho Huatuco

1Department of Computer Science, Federal University of Ouro Preto, Ouro Preto, MG, Brazil, * corresponding author: santibanez.ernesto@gmail.com

2Escuela Internacional de Ciencias Económicas y Administrativas, Universidad de La Sabana, Km 7 autopista norte de Bogotá, D.C., Chía (Cundinamarca), Colombia

3Leeds University Business School, University of Leeds, Woodhouse Lane, Leeds, LS2 9JT, UK

Abstract The design of a reverse logistics supply chain is a complex problem and is still relatively unexplored and underdeveloped. We propose a binary particle swarm optimization (BPSO)-based scheme for solving a NP-hard reverse logistics supply chain design problem. The algorithm combines a traditional stochastic search with an optimal solution method for solving to optimality a relaxed linear programming (LP) model. We divide the swarm into two elementary groups. The first swarm group guides the search for the best location of remanufacturing facilities, while the second swarm group defines the optimal flows between the facilities. We solve to optimality, a relaxed linear problem is obtained from the original problem and then we project the solution into the swarm space. This approach was validated using different data sets (instances) randomly generated with up to 350 sourcing facilities, 100 candidate sites for locating reprocessing facilities and 40 remanufacturing facilities. The results regarding experimental approximability of proposed heuristic versus the optimal solution are very promising, as well as their computation times.

Keywords sustainable supply chain; reverse logistics; remanufacturing; particle swarm optimization; integer programming.

1. Introduction

In this paper, we focus on the remanufacturing supply chain design problem and then on reverse logistics supply chain. Here, remanufacturing is defined as one of the recovery methods by which worn-out products or parts are recovered to produce a unit equivalent in quality and performance to the original new product. As a consequence, these remanufactured products or parts can be resold as new products or parts. Because remanufacturing activities are often implemented by the
original producer, such a supply chain is likely to be a closed-loop system. Remanufacturing activities are recognized as a main option of recovery in terms of its feasibility and benefits. It provides firms a way to master the disposal of their used products, to reduce effectively the costs of production and save raw materials. When designing a reverse logistics supply chain for remanufactured products, several well-known supply chain design problems arise. For instance, Facility location is one of the strategic problems being part of the supply chain planning process. The problem of locating facilities and allocating customers is not new to the Operations Research community and covers the key aspects of supply chain design [5]. This problem is one of “the most comprehensive strategic decision problems that need to be optimized for long-term efficient operation of the whole supply chain” [2]. As observed by Farahani and Hekmatfar [9], some small changes to classical facility location models turn these problems quite hard to solve.

Particle Swarm Optimization (PSO) is one of the meta-heuristic algorithms that have been successfully applied in many areas and is a suitable approach for several optimization problems. As pointed out by some authors, this technique has succeeded in many continuous problems, however, in discrete or binary version there are still some difficulties [8]. In this paper, we adapt the binary PSO algorithm to solve the reverse logistics supply chain design problem and analyze its performance, in terms of computational times and quality of the solution obtained, over a set of randomly generated instances (data sets).

This paper makes two primary contributions. Firstly, it proposes a PSO-based scheme, combining an optimization method inside of a PSO algorithm scheme for solving a reverse distribution problem. Secondly, in contrast with traditionally studied approaches that focus only on small sized problems, the current paper presents a set of computational experiments carried out on medium to large sized data sets (instances).

2. Literature Review
In the last few years, mathematical modeling and solution methods for the efficient management of return flows (and/or integrated with forward flows) has been studied in the context of reverse logistics, closed-loop supply chain and sustainable supply chain management. Dekker et al. [6] classified the research on reverse logistics (and closed-loop supply chain) into three functional areas: Distribution, Inventory and Production and Supply Chain Scope. Traditional distribution decisions involve the design of supply chain and the location of forward and reverse facilities for the distribution of products and for collecting and reprocessing returned products. These authors focused on quantitative models for designing closed-loop supply chains, and were concerned with the decisions regarding the
topological structure of the supply chain, the number of facilities to locate, their locations and capacities and the allocation of product flows between the facilities. Rubio et al. [23] carried out a compilation of research published on reverse logistics within the period 1995-2005. They studied the topic based on the classification proposed by Dekker et al. [6].

Reverse logistics models have been discussed previously (e.g. [7],[10],[12],[21],[24],[25],[27]). Almost all the authors proposed MILP models. The majority of solution methods are based on standard commercial packages. Closed-loop supply chain models have also been taken into account (e.g. [1],[3],[4],[13],[18],[20],[28]. Stochastic models in combination with multi-objective functions were presented by Pishvaee and Torabi [22], while the work of Govindan et al. [11] presented a multi-criteria decision making approach based on the Analytical Network Process (ANP) to solve the problem of selecting a reverse logistics provider.

Because of the computational complexity of this problem, its solution using meta-heuristic algorithms has also been of interest by some researchers. Procedures such as Simulated Annealing [15],[21], Genetic Algorithms [16],[19],[28], Memetic Algorithms [22] and Tabu Search [15] have been proposed. Notice that Lee et al. [16] solved problems of up to 15 returning centers, 10 disassembly centers and 14 processing centers. They did not report computational results regarding time or the quality of the solution obtained, measured using the gap (approximation) to the optimal solution. Lee and Dong [15] solved problems with supply chains of up to 100x40x30 with a maximum computing time of 2,939 seconds and a maximum gap (compared to a lower bound) of 15%.

3. Mathematical model for designing a remanufacturing supply chain

In this section, we present a mixed-integer linear programming (MILP) model for the problem of designing a remanufacturing supply chain. This problem can be categorized as a single product, static, three-echelon, capacitated location model with known demand. The remanufacturing supply chain consists of three types of members: sourcing facilities (e.g. a retail store), collection sites and remanufacturing facilities. At the customer level, there are product demands and used products ready to be recovered such as, for example, cell phones. We suppose that customers return products to origination sites like a retail store. At the second layer of the supply chain, there are reprocessing centers (collection sites) used only in the reverse channel and they are responsible for activities, such as cleaning, disassembly, checking and sorting, before the returned products are sent back to remanufacturing facilities. At the third layer, remanufacturing facilities accept the checked returns from intermediate facilities and they are responsible for the process of remanufacturing. In this paper we address the backward flow of
returns coming from sourcing facilities and going to remanufacturing facilities through reprocessing facilities properly located at pre-defined sites. In such a supply chain, the reverse flow, from customers through collection sites to remanufacturing facilities is formed by used products, while the other (“forward” flow) from remanufacturing facilities directly to point of sales consists of “new” products. Figure 1 shows a reverse logistics supply chain with six sourcing facilities, eight candidate sites for locating reprocessing facilities (three out of eight facilities are opened) and two remanufacturing facilities.

![Figure 1. Schematic representation of the reverse logistics supply chain under study](image)

**Remanufacturing supply chain design problem (RSCP) Model**

Our model assumes that product demands (new ones) and available quantities of used products at the customers are known and deterministic. We introduce the following inputs and sets:

- $I$ = the set of sourcing facilities at the first layer, indexed by $i$
- $J$ = the set of remanufacturing nodes at the third layer indexed by $j$
- $K$ = the set of candidate reprocessing facility locations at the mid layer, indexed by $k$

- $a_i$ = supply quantity at source location $i \in I$
- $b_j$ = demand quantity at remanufacturing location $j \in J$
- $f_k$ = fixed cost of locating a mid-layer reprocessing facility at candidate site $k \in K$
- $g_k$ = management cost at a mid-layer reprocessing facility at candidate site $k \in K$
- $c_{ik}$ = is the unit cost of delivering products at $k \in K$ from a source facility located in $i \in I$
- $d_{kj}$ = is the unit cost of supplying demand $j \in J$ from a mid-layer facility located in $k \in K$
- $u_k$ = capacity at reprocessing facility location $k \in K$
We consider the following decision variables:
\[ w_k = \begin{cases} 
1 & \text{if we locate a reprocessing facility at candidate site } k \in K, \\
0 & \text{otherwise} 
\end{cases} \]
\[ x_{ik} = \text{flow from source facility } i \in I \text{ to reprocessing facility located at } k \in K \]
\[ y_{kj} = \text{flow from remanufacturing facility located at } k \in K \text{ to facility } j \in J \]

Following the model proposed by Li [17], the remanufacturing supply chain design problem (RSCP) is defined by:

\[
\text{Minimize } \sum_{k \in K} f_k w_k + \sum_{i \in I} \sum_{k \in K} c_{ik} x_{ik} + \sum_{k \in K} \sum_{j \in J} g_k x_{jk} + \sum_{k \in K} \sum_{j \in J} d_{kj} y_{kj} \quad (1)
\]

\[
\sum_{i \in I} x_{ik} \leq u_k w_k \quad \forall k \in K \quad (2)
\]

\[
\sum_{k \in K} x_{ik} \leq a_i \quad \forall i \in I \quad (3)
\]

\[
\sum_{j \in J} y_{kj} \geq b_j \quad \forall j \in J \quad (4)
\]

\[
\sum_{i \in I} x_{ik} = \sum_{j \in J} y_{kj} \quad \forall k \in K \quad (5)
\]

\[
x_{ik}, y_{kj} \geq 0, \quad \forall i \in I, \forall j \in J, \forall k \in K \quad (6)
\]

\[
w_k \in \{0,1\} \quad \forall k \in K \quad (7)
\]

The objective function (1) minimizes the sum of the installation reprocessing facility costs plus delivering costs from sourcing facilities to reprocessing facilities and from these to remanufacturing facilities. Constraint (2) ensures that supplying at facility \( k \in K \) is delivered to a mid-layer reprocessing facility already opened. Constraint (3) ensures that all return products from \( i \in I \) go backward to facility \( k \in K \). Constraint (4) ensures that demand at remanufacturing facility \( j \in J \) is satisfied by reprocessing facilities \( k \in K \). Constraint (5) ensures that all return products arriving to facility \( k \) are also delivered to remanufacturing facilities. Constraints (6) and (7) are respectively standard positive and binary constraints.

4. Solution approach based on PSO

In our solution approach, we use a BPSO algorithm for guiding the whole process of seeking an optimal solution for the RSCP problem. However, part of the particle is obtained by solving to optimality the RSCP problem. In this way, we use a decomposed BPSO algorithm, based into two groups of swarms, one of them guide the overall search for an optimal solution.
Let $L$ be the size of the swarm, each particle $i$ has the position $u_i \in \{0,1\}^n$ in the search space and a velocity $v_i \in \mathbb{R}^n$. Let $p_i$ be the best known position for particle $i$ and $p_g$ the best known position for the swarm. The fitness function $f(.)$ is the objective function value for a particle. The pseudo-code of our proposed Binary PSO Algorithm is shown in Table 1.

**Table 1. Pseudo-code of proposed BPSO algorithm**

1. **Generate** the swarm
   a. **For** each particle in the swarm $i = 1, \ldots, L$ **do:**
   i. Initialize the position of particle $i$ using a random uniform distribution: $u_i \sim U(0,1)$;
   c. **Analyze** feasibility for each particle $i$. Whenever necessary apply the feasibility procedure;
   d. **Solve** to optimality the $RSCP_R$ problem and complete the dimension of particle $i$;
   e. **Initialize** the best position of particle $i$ as its initial position: $p_i \leftarrow u_i$;
   f. **If** ($f(p_i) < f(p_g)$) **update** the best swarm position: $p_g \leftarrow p_i$;
   g. **Initialize** velocity of particle: $v_i^1 \sim U(-4,4)$

2. **While** stopping criteria **do**:
   a. **Update** value of $t$;
      i. **For** each particle $i = 1, \ldots, L$ **do**:
         1. **For** each dimension $d = 1, \ldots, n$ **do**:
             a. Select random numbers: $r_1, r_2 \sim U(0,1)$
         2. **Update** velocity of particle using (8)
         3. **Update** position of particle using (10)
         4. **Solve** to optimality the $RSCP_R$ problem and complete the dimension of particle;
         5. **If** ($f(p_i) < f(p_g)$) **do**:
            a. **update** the best position of particle: $p_i \leftarrow x_i$
            b. **If** ($f(p_i) < f(p_g)$) update the best swarm position: $p_g \leftarrow p_i$
   b. $p_g$ contains the best solution found so far.
   c. **EndWhile**

**4.2 The particle coding method**

To find a good coding method corresponding to the optimization problem is the most critical problem [14]. In this paper, the particle’s $d$-dimensional space is divided into two sets $u^1$ and $u^2$. The length of each set (vector) respectively corresponds to the number of candidate sites to locate reprocessing facilities $|K|$, the second part represents the solution to the relaxed LP associated.
In the first group of particles, every component of each vector can take only 1 or 0. If the $k$-th component of $u^1$ is equal to 1, then the reprocessing facility at candidate site $k$ must be open, 0 otherwise. The second group of particles represents the flows of return products between the sourcing facility and the reprocessing facilities and between the reprocessing facilities and the remanufacturing facilities. As described earlier in this paper, this part is obtained solving to optimality the $RSCP_R$ problem.

4.3 Generating the initial swarm

For step (1) of the algorithm, the initial swarm of size $L$, with the position of each particle $i$ of the swarm is generated as follows:

**Steps (1b) and (1c)**

For particles belonging to the first swarm group, the position is generated randomly. First, a uniform distribution between [0,1] is used to generate a value $\text{rand}()$. For each velocity vector $v^t_{id}$, $t=0$, the position $d$ is generated randomly within the interval [-4,4] as is usually done in the continuous PSO algorithm. Equation (10) is used to obtain the initial position of a particle $i$. Alternatively, a binary uniform distribution can be used to generate the initial position $u_i$. A $k$-dimensional vector is used to represent the initial position $u_i$ of each particle, where $|K|$ is the number of candidate sites to locate a reprocessing facility. This procedure can generate unfeasible solutions to the problem.

**Step (1d)**

For each particle of the first group corresponds to one particle of the second group. The position of a particle in the second group is generated by solving the associated LP model, which in turn considers the position of a particle in the first group. For instance, let us suppose that for particle $u_3$ is solved the problem $RSCP_R$. The position for particle $u_3$ of this group can be the following:

$$(100;0;150;0;150;0;100;250;0;0;250).$$

For this instance problem with 4 sourcing facilities, 2 reprocessing facilities and 2 remanufacturing facilities, the position of $u_3$ indicates that 100 units of products flow from the first source facility to the first reprocessing facility opened, 150 units of products flow from the second sourcing facility to the first reprocessing facility opened, 150 units of products flow from the third sourcing facility to the second reprocessing facility opened and 100 units of products flow from the fourth sourcing facility to the second reprocessing facility opened. Then, 250 units of products are transported from the first reprocessing facility opened to the first remanufacturing facility opened and the same volume is transported from the second reprocessing facility opened to the second remanufacturing facility opened.
5. Numerical experiments

In this section we discuss and compare the computational results obtained by the proposed evolutionary algorithm. Our proposal is to analyze the performance of the BPSO algorithm regarding both the quality of the upper bound (solution) obtained and computational time required to obtain this solution.

We implemented the MILP models in GAMS (General Algebraic Modeling System) and used CPLEX solver as a subroutine for solving to optimality the RSCP problem. All the experiments were developed on a PC with 4Gb RAM and 2.3GHz. In the literature, there are not large data sets available for our problem. We hence randomly generated 10 test problems following similar methodologies used for well-known related supply chain problems (e.g. [18],[26]). These test problems are data sets corresponding to supply chains of up to 350 origination sites, 100 candidate sites for locating reprocessing facilities and 40 remanufacturing facilities. The data sets for the test problems are given in Table 2.

<table>
<thead>
<tr>
<th>#</th>
<th>Problem size</th>
<th>Origination sites</th>
<th>Candidates reprocessing facilities</th>
<th>Remanufacturing facilities</th>
<th>Fixed costs</th>
<th>a_j</th>
<th>m_k</th>
<th>b_l</th>
</tr>
</thead>
<tbody>
<tr>
<td>I1</td>
<td>40x20x15</td>
<td>40</td>
<td>20</td>
<td>15</td>
<td>300</td>
<td>150</td>
<td>400</td>
<td>400</td>
</tr>
<tr>
<td>I2</td>
<td>100x40x20</td>
<td>100</td>
<td>40</td>
<td>20</td>
<td>500</td>
<td>150</td>
<td>400</td>
<td>750</td>
</tr>
<tr>
<td>I3</td>
<td>150x40x20</td>
<td>150</td>
<td>40</td>
<td>20</td>
<td>1000</td>
<td>200</td>
<td>800</td>
<td>1500</td>
</tr>
<tr>
<td>I4</td>
<td>200x80x20</td>
<td>200</td>
<td>80</td>
<td>20</td>
<td>1000</td>
<td>300</td>
<td>800</td>
<td>3000</td>
</tr>
<tr>
<td>I5</td>
<td>300x80x40</td>
<td>300</td>
<td>80</td>
<td>40</td>
<td>2000</td>
<td>200</td>
<td>800</td>
<td>1500</td>
</tr>
<tr>
<td>I6</td>
<td>350x100x40</td>
<td>350</td>
<td>100</td>
<td>40</td>
<td>2000</td>
<td>200</td>
<td>800</td>
<td>1750</td>
</tr>
<tr>
<td>I7</td>
<td>200x80x20</td>
<td>200</td>
<td>80</td>
<td>20</td>
<td>10000</td>
<td>300</td>
<td>800</td>
<td>3000</td>
</tr>
<tr>
<td>I8</td>
<td>300x80x40</td>
<td>300</td>
<td>80</td>
<td>40</td>
<td>20000</td>
<td>200</td>
<td>800</td>
<td>1500</td>
</tr>
<tr>
<td>I9</td>
<td>200x80x20</td>
<td>200</td>
<td>80</td>
<td>20</td>
<td>10000</td>
<td>300</td>
<td>1200</td>
<td>3000</td>
</tr>
<tr>
<td>I10</td>
<td>300x80x40</td>
<td>300</td>
<td>80</td>
<td>40</td>
<td>20000</td>
<td>200</td>
<td>1200</td>
<td>1500</td>
</tr>
</tbody>
</table>

We run 5 trials for each test instance. Table 3 presents the results obtained using the algorithm.

<table>
<thead>
<tr>
<th># instance</th>
<th>z_max</th>
<th>z_min</th>
<th>%dev (OPT)</th>
<th>%imp(z_max/z_min)</th>
</tr>
</thead>
<tbody>
<tr>
<td>I1</td>
<td>Minimum 148500</td>
<td>146900</td>
<td>0.00</td>
<td>1.08</td>
</tr>
<tr>
<td>I1</td>
<td>Maximum 150050</td>
<td>149000</td>
<td>1.43</td>
<td>0.70</td>
</tr>
<tr>
<td>I1</td>
<td>Average 149310</td>
<td>148000</td>
<td>0.75</td>
<td>0.88</td>
</tr>
<tr>
<td>I2</td>
<td>Minimum 520300</td>
<td>520250</td>
<td>0.03</td>
<td>0.01</td>
</tr>
<tr>
<td>I2</td>
<td>Average 520750</td>
<td>520290</td>
<td>0.04</td>
<td>0.09</td>
</tr>
<tr>
<td>I3</td>
<td>Minimum 1065400</td>
<td>1065200</td>
<td>0.23</td>
<td>0.02</td>
</tr>
</tbody>
</table>
A Binary Particle Swarm Optimization (BPSO) algorithm to solve a reverse logistics supply chain problem
Ernesto Santibanez-Gonzalez, Jairo Montoya-Torres, Luisa Huaccho Huatuco

<table>
<thead>
<tr>
<th>Instance</th>
<th>Maximum</th>
<th>Average</th>
<th>Minimum</th>
<th>Maximum</th>
<th>Average</th>
<th>Minimum</th>
<th>Maximum</th>
<th>Average</th>
<th>Minimum</th>
<th>Maximum</th>
<th>Average</th>
<th>Minimum</th>
</tr>
</thead>
<tbody>
<tr>
<td>I4</td>
<td>1066600</td>
<td>1065800</td>
<td>0.28</td>
<td>0.08</td>
<td>2090500</td>
<td>2089000</td>
<td>0.30</td>
<td>0.07</td>
<td>2130800</td>
<td>2130800</td>
<td>0.38</td>
<td>0.00</td>
</tr>
<tr>
<td>I5</td>
<td>1065960</td>
<td>1065440</td>
<td>0.25</td>
<td>0.04</td>
<td>2090740</td>
<td>2089800</td>
<td>0.34</td>
<td>0.04</td>
<td>2131780</td>
<td>2130920</td>
<td>0.39</td>
<td>0.04</td>
</tr>
<tr>
<td>I6</td>
<td>2091400</td>
<td>2090500</td>
<td>0.37</td>
<td>0.04</td>
<td>2468750</td>
<td>2467350</td>
<td>0.35</td>
<td>0.06</td>
<td>2132400</td>
<td>2131100</td>
<td>0.40</td>
<td>0.06</td>
</tr>
<tr>
<td>I7</td>
<td>2765960</td>
<td>2765440</td>
<td>0.25</td>
<td>0.04</td>
<td>2767920</td>
<td>2764100</td>
<td>0.23</td>
<td>0.05</td>
<td>2766400</td>
<td>2765120</td>
<td>0.27</td>
<td>0.05</td>
</tr>
<tr>
<td>I8</td>
<td>3481300</td>
<td>3480200</td>
<td>0.22</td>
<td>0.03</td>
<td>3483000</td>
<td>3480200</td>
<td>0.22</td>
<td>0.03</td>
<td>3481300</td>
<td>3480200</td>
<td>0.22</td>
<td>0.03</td>
</tr>
<tr>
<td>I9</td>
<td>2471900</td>
<td>2469550</td>
<td>0.44</td>
<td>0.10</td>
<td>2470920</td>
<td>2468400</td>
<td>0.40</td>
<td>0.10</td>
<td>2765500</td>
<td>2764100</td>
<td>0.23</td>
<td>0.05</td>
</tr>
<tr>
<td>I10</td>
<td>2130800</td>
<td>2130800</td>
<td>0.38</td>
<td>0.00</td>
<td>2132400</td>
<td>2131100</td>
<td>0.39</td>
<td>0.04</td>
<td>2765960</td>
<td>2765440</td>
<td>0.25</td>
<td>0.04</td>
</tr>
<tr>
<td></td>
<td>2471900</td>
<td>2469550</td>
<td>0.44</td>
<td>0.10</td>
<td>2470920</td>
<td>2468400</td>
<td>0.40</td>
<td>0.10</td>
<td>2765500</td>
<td>2764100</td>
<td>0.23</td>
<td>0.05</td>
</tr>
</tbody>
</table>

For each instance problem, the table provides solution obtained at the initial random phase of the algorithm ($z_{ran}$), the final value obtained by BPSO algorithm ($z_p$). The last two columns of this table present respectively the percentage deviation, denoted as $\%dev(OPT)$, between the final solution $z_p$ against the optimal value ($z^*$) obtained using the MILP model, and the percentage of improvement of the objective function value (e.g., $z_{ran}$ versus $z_p$). These two metrics are computed using Equations (11) and (12) respectively:

$$\%dev(OPT) = \frac{z^* - z_p}{z^*} \times 100\%$$

$$%imp(z_{ran}, z_p) = \frac{z_{ran} - z_p}{z_{ran}} \times 100\%$$

All values of $\%dev$ and $\%imp$ were rounded to two decimals. Observe that, for all the test instances, the maximum average $\%dev$ is 1.58% (instance #9) and the minimum average $\%dev$ is 0.04% (instance #2). Note also that BPSO procedure obtained the optimal solution for instance I9. From the results presented in this table, we observe that the proposed BPSO performs very well since percentage deviations from optimal values are never higher than 1.70% (for the maximum values) and no greater than 0.54% in average. In addition, according to values obtained for metric $%imp$, we observe that the improvement of the initial solution is about 0.35%, in average, which means that the BPSO algorithm shows very fast convergence. It is to note that the highest value of this improvement is observed for instance #9, with maximum and averages values of 7.56% and 2.02%,
respectively. This means that, in one of the trials, the initial random solution employed by BPSO had to be improved in a 7.56%. When regarding the maximum value of the final deviation of the BPSO solution against the optimum for this data set (instance #9), we notice that again BPSO algorithm performs very well because the final solution is only 1.69% higher than the optimal value given by MILP. Finally, looking at computational times, we notice that all test instances were solved in less than 58 seconds, as shown in Table 4.

<table>
<thead>
<tr>
<th># instance</th>
<th>Minimum</th>
<th>Maximum</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>I1</td>
<td>13.40</td>
<td>13.87</td>
<td>13.59</td>
</tr>
<tr>
<td>I2</td>
<td>15.73</td>
<td>15.98</td>
<td>15.88</td>
</tr>
<tr>
<td>I3</td>
<td>19.01</td>
<td>20.10</td>
<td>19.75</td>
</tr>
<tr>
<td>I4</td>
<td>32.40</td>
<td>34.85</td>
<td>33.45</td>
</tr>
<tr>
<td>I5</td>
<td>54.80</td>
<td>58.32</td>
<td>56.35</td>
</tr>
<tr>
<td>I6</td>
<td>53.82</td>
<td>54.70</td>
<td>54.29</td>
</tr>
<tr>
<td>I7</td>
<td>35.75</td>
<td>40.01</td>
<td>36.71</td>
</tr>
<tr>
<td>I8</td>
<td>54.33</td>
<td>56.52</td>
<td>55.28</td>
</tr>
<tr>
<td>I9</td>
<td>35.46</td>
<td>36.79</td>
<td>36.27</td>
</tr>
<tr>
<td>I10</td>
<td>38.97</td>
<td>40.01</td>
<td>40.68</td>
</tr>
<tr>
<td>Average</td>
<td>35.367</td>
<td>37.915</td>
<td>36.725</td>
</tr>
</tbody>
</table>

6. Conclusions
This paper considered the problem of designing a reverse logistics supply chain. In the first instance, a mixed-integer linear programming (MILP) model was proposed in order to formalize the relation between the different elements of the reverse logistics supply chain. However, as this problem is classified as NP-hard, we proposed a solution scheme based on Binary Particle Swarm Optimization (BPSO). The algorithm guides the search for a solution by combining, on the one hand, a random solution search with an optimal solution, on the other hand, obtained by solving an associated relaxed problem. The algorithm generates two groups of swarms: (1) particles representing whether the facility is opened or closed, and (2) a set of chromosome when solving to optimality the associated relaxed problem. Computational experiments were carried out using random generated data sets. The aim of these experiments was to analyze the performance of the proposed BPSO algorithm on large sized data sets. Since approximability of solutions obtained by BPSO was very close to the optimum values in very short computation time, the proposed procedure looks to be very promising for actual implementation in real-life for the design of reverse logistics supply chains.

In addition, several lines for further research can be proposed. In the first instance, the model under study in this paper considered economic metrics (costs), especially in the objective function. Other dimensions of sustainable supply chain
management (e.g., social / societal issues) can be included in the model as part of the constraints or in the objective function. Another opportunity for further research may be the inclusion of stochastic issue into the model. Finally, a trend in recent years regarding the implementation of intelligent algorithms is the hybridization of traditional meta-heuristics with other procedures. Hence, although the proposed BPSO algorithm incorporates the use of mixed-integer linear programming during the resolution stages, it could be interesting to analyze the benefits that could be achieved by incorporating other intensification or diversification procedures as additional steps in the procedure.

Acknowledgements
The work of Jairo R. Montoya-Torres was supported by a Marie Curie International Incoming Fellowship within the 7th European Community Framework Programme (project “DISRUPT”, grant No. ESR-299255).

References
A Binary Particle Swarm Optimization (BPSO) algorithm to solve a reverse logistics supply chain problem
Ernesto Santibanez-Gonzalez, Jairo Montoya-Torres, Luisa Huaccho Huatuco