# Discrete penguins search optimization algorithm to solve flow shop scheduling problem

**Ilyass Mzili[1], Mohammed Essaid Riffi[2], Fatiha Benzakri[3]**
[1,2]Department of Computer Science, Faculty of Sciences, Chouaïb Doukkali University, Morocco
[3]Department of Mathematics, Faculty of Science, Chouaïb Doukkali University, Morocco

**Article Info**

***Article history:***

Received Jun 30, 2019
Revised Dec 26, 2019
Accepted Feb 24, 2020

***Keywords:***

Combinatorial optimization
Flow shop scheduling problem
PeSOA
Makespan
Metaheuristics
NP-Hard

**ABSTRACT**

Flow shop scheduling problem is one of the most classical NP-hard optimization problem. Which aims to find the best planning that minimizes the makespan (total completion time) of a set of tasks in a set of machines with certain constraints. In this paper, we propose a new nature inspired metaheuristic to solve the flow shop scheduling problem (FSSP), called penguins search optimization algorithm (PeSOA) based on collaborative hunting strategy of penguins.The operators and parameter values of PeSOA redefined to solve this problem. The performance of the penguins search optimization algorithm is tested on a set of benchmarks instances of FSSP from OR-Library, The results of the tests show that PeSOA is superior to some other metaheuristics algorithms, in terms of the quality of the solutions found and the execution time.



***Corresponding Author:***

Ilyass Mzili,
Department of Computer Science, Faculty of Sciences,
Chouaib Doukkali University,
El Jadida, Morocco.
Email: dr.mzili.ilyass@gmail.com

## 1. INTRODUCTION

The Scheduling is a branch of this operational research in production management that aims to improve the efficiency of a company in terms of production costs and delivery times. Scheduling problems appear in all areas of the economy: computers, construction (project management), industry (workshops problems, production management), administration (schedule) .The flow shop problem scheduling [1] is one of the most difficult combinatorial optimization problem belonging to NP-hard problem [2] family, it is widely known in the industry. The solution to the problem in finding an order for execution of tasks on machines subject to many constraints in an optimal time.

Over the last few decades, nature has been a source of inspiration for many metaheuristics, which has been introduced to solve optimization problems. A set of these metaheuristics has been tested to solve discrete problems. The results of these tests are not unique, the quality of solutions varies according to the characteristics and category of method. Generally, methods based on particle population swarm intelligence algorithms providing solutions of good quality, for example: bat algorithm (BA) [3, 4], particle swarm optimization (PSO) [5], artificial bee colony (ABC) [6], cat swarm optimization (CSO) [7, 8], hunting search algorithm (HuS) [9], elephant herding optimization (EHO [10], swallow swarm optimization (SSO) [11], golden ball algorithm (GBA) [12, 13], cuckoo search (CS) [14], chicken swarm algorithm (CSA) [15-16] and flower pollination algorithm (FPA) [17]. In this context we proposed a new metaheuristic of swarm intelligence inspired by nature nominate PeSOA, to solve the flow shop scheduling problem one of the NP-hard problem in combinatorial optimization. PeSOA is an optimization technique inspired by the hunting strategy of penguins, which was developed to deal with optimization problems in the continuous case. Recently, this algorithm has been used to solve discrete problems [18-20] because of the satisfied results in the continuous case.

In this paper, we propose an adaptation of PeSOA algorithm to solve FSSP. In this one, we adapt the parameters values and operators PeSOA to find good-quality solutions. The performance of PeSOA algorithm evaluated by a test on a set of benchmarks of FSSP from the OR-Library of various sizes and a comparison with other métaheuristics. The organizational structure of this paper as follows: The second section is contains a short description of flow shop scheduling problem.In the third section PeSOA algorithm are presented.In the fourth section,adaptation of PeSOA to solve the FSSP. The experimental results and discussion in the fifth section. Finally, a conclusion and perspectives.

## 2. FLOW SHOP SCHEDULING PROBLEM SEARCH

### 2.1. Presentation

The FSSP is an important issue in production scheduling. This problem was first proposed in 1954 by Johnson [21] and classified as a NP-hard problem. The FSSP is described by a set n jobs J = {1,..., n} not related, must be treated in m machines in M = {1,..., m}. Each job j contains a sequence of operations O = {Oji, j ∈ {1,..., n} and i ∈ {1,..., m}} which have been executed in a given order. Following this order, each operation Oij must be performed on a specified machine k for a specified time Pijk. Each machine can only process one job at a time, and each job has to go through each machine once and only once. The FSSP consists in finding a schedule to perform operations on the machines that minimizes the Cmax (makespan), ie the time required to perform all jobs.

### 2.2. Formulation

FSSP are often designated by the $m$, n, $\pi$ and $C$max symbols where $n$ represents the number of jobs; $m$ is the number of machines, $\pi = (j_1, j_2 ... j_n)$ is a planning permutation of all jobs and $C$max is the makespan. Let t $(i, j)$ $(1 \le i \le n, 1 \le j \le$ m) the processing times of job $i$ on the machine $j$, assuming that the preparation time for each job is zero or included in the working time treatment, ; $\pi = (j_1, j_2, ..., j_n)$ is a planning permutation of all jobs. Π is the overall planning of permutation. $C(j_i, k)$ is the completion time of the job $j_i$ on the machine $k$.The completion time of the flow shop problem planning $\pi = (j_1, j_2, ..., j_n)$ is illustrated as follows :

$$C(j_1, 1) = t(j_1, 1), \tag{1}$$

$$C(j_i, 1) = C(j_{i-1}, 1) + t(j_i, 1), \quad i = 2, 3, \ldots, n, \tag{2}$$

$$C(j_1, k) = C(j_1, k-1) + t(j_1, k), \quad k = 2, 3, \ldots, m, \tag{3}$$

$$C(j_i, k) = \max\{C(j_{i-1}, k), C(j_i, k-1)\} + t(j_1, k), \quad i = 2, 3, n, k = 2, 3, \ldots, m, \tag{4}$$

$$\pi* = \arg \{C\max(\pi) = C(jn, m)\} \rightarrow \min, \forall \pi \in \Pi, \tag{5}$$

where π * is the most appropriate arrangement which is the purpose of the permutation of flow shop problem to find $C$max (π *) as the minimal component.

We consider the flow shop scheduling problem: in a car paint factory. We calculated the duration of painting of two cars (jobs), knowing that car painting is a sequence of two operations, degreasing and painting (machines). The Figure 1 presents the time required to degrease and paint the two cars 1 and 2. The problem data is two jobs, two machine and the completion time of each job on each machine. The optimal problem solution is car2 - car1, the completion time (Cmax) is 12. The Figure 2 shows the use of a gantt chart to calculate the completion time and the planning the implementation of the solution.

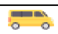
| | Degreasing | Painting |
|---|---|---|
| Car1  | 5 | 3 |
| Car2  | 4 | 4 |

Figure 1. The processing time of the cars paint

| TASK NAME | TIME | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| Degreasing | Car2 | | | | Car1 | | | | | | | |
| Painting | | | | | Car2 | | | | | Car1 | | |

Figure 2. Gantt chart

## 3. PeSOA ALGORITHM

The PeSOA algorithm is a new metaheuristic [22] nature-inspired based on the collaborative hunting behavior of penguins. The hunting strategy of penguins base on collaborate their efforts and synchronize their dives to optimize the global energy in the process of collective hunting and nutrition. Penguins divide into groups to start hunting. As penguins eat fish, they are diving to harvest food until oxygen reserves are depleted.

The hunting strategy of penguins; each group of penguins starts searching in a specific position and random levels, the penguin of each group looks for foods in random way and individually in its group, and after an approximate number of dives, the penguins get back on the ice to share with its affiliates. If the amount of food in a position is not enough for the group, part of the group migrates to another position. The group that ate the most fish gives us the location of the rich food represented by the hole and the level.

The PeSOA algorithm is performed through a set of penguin population (random initial solutions). In the next step, the amount of fish consumed is the objective function related to each member of the population. The optimal value of the objective function presented the location where there is a large amount of food, which was granted by the groups. To find a location of enough food (best solutions), penguins move to a new location identified. This movement expressed this formula follows ;

$$X_{new} = X_{id} + rand \times |X_{bestlocal} - X_{id}| \tag{6}$$

where rand is a random number for distribution; and we have three solution, $X_{bestlocal}$ the best local solution, $X_{id}$ the last solution and the $X_{new}$ is a new solution.The pseudo code of the PeSOA algorithm:

**Algorithm 1** : Pseudo-code of PeSOA

```
1.  Generate random population P of the initial solutions
    (penguins);
2.  Initialize the probability of existence of fish in the
    holes-levels;
3.  While (the stop condition is not satisfied) do
4.  For each i individual of P do
5.  While (RO2> 0) do
6.   - Take a random step
7.   - Improve the penguin position using Eq (6);
8.   - Update quantities of fish eaten for this penguin.
9.  End while
10. End for
11. Update quantities of eaten fish in the holes-levels.
12. Update the best group.
13. Recalculate the probability of existence of fish in the
    holes-levels
14. Update best- solution
15. End while
16. Return best solution.
```

## 4. APPLY PeSOA TO FLOW SHOP SCHEDULING PROBLEM

We applied PeSOA on FSSP to improve the results and to show its performance against other metaheuristics. The PeSOA as all metaheuristics introduced initially for continuous optimization, PeSOA standard continuous coding scheme cannot be used directly to solve discrete problems like flow shop problem, we adapt the algorithm parameters and the FSSP solution form in the space and the transition from the current solution to another one.

### 4.1. FSSP solution

The position of penguin presents a solution. Therefore, PeSOA considered a penguin in a population as a solution in the search space. Considering an instance of FSSP, for example Table 1 that contains 4 jobs and 3 machines, and each job operation is associated with its machine and processing time.

Table 1. FSSP instance (4×3)

| | M1 | M2 | M3 |
|---|---|---|---|
| *J1* | 6 | 1 | 4 |
| *J2* | 3 | 6 | 2 |
| *J3* | 1 | 2 | 1 |

A solution of FSSP instance is a permutation of jobs that can be presented by a series of n integers (Jobs) S = {1, 2, 3... n}, each integer i in the series is designated the index of job and the order of i in S is the order of processing of these jobs in each machine. In our example, consider an initial solution of instance (4.3) 2-1-4-3 which represents the order of processing of the 4 jobs on each machine. For example, the machines starts processing jobs 2 and ends with jobs 3. Considering Table 1 and the initial solution, we can design the Gantt chart as shown in Figure 3 of the initial solution (2-1-4-3) to calculate the objective function (Cmax) of proposed solution. The makespan of the instance solution is 21.

<table>
<tr><td rowspan="2">Machines</td><td colspan="21">Time</td></tr>
<tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td>15</td><td>16</td><td>17</td><td>18</td><td>19</td><td>20</td><td>21</td></tr>
<tr><td>M1</td><td colspan="3">2</td><td colspan="6">1</td><td colspan="2">4</td><td>3</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>M2</td><td></td><td></td><td></td><td colspan="6">2</td><td>1</td><td></td><td>4</td><td colspan="2">3</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>M3</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td colspan="2">2</td><td colspan="4">1</td><td colspan="5">4</td><td>3</td></tr>
</table>

Figure 3. Gantt chart of initial solution 2-1-4-3 (S= (2-1-4-3))

### 4.2. Position updating of PeSOA

The transition from one solution to another in the search space is an operation based on the notion of length and topology. The length is designated as the cost of solution, the topology of passage is a technique performed by the operators of (7)

$$S_{new} = S_{id} \oplus rand \otimes (S_{best} - S_{id}) \quad (7)$$

where rand is a random number for distribution; and we have three solution, the best local solution ($S_{best}$), the current solution ($S_{id}$) and the new solution ($S_{new}$).
The operations in (7) defined as follows:

a) Substruction operation $-$

An operation between two solutions(scheduling) $S_1$ and $S_2$, which gives a displacement vector Q. in this way, extracts the permutations applied to the two solutions $S_2$ to obtain $S_1$, for example:

Whether $S_1 = \{j_1, j_2, j_3, j_4 \dots j_{n-1}, j_n\}$ and $S_2 = \{j_2, j_3, j_1, j_4 \dots j_n, j_{n-1}\}$
$Q = S_1 - S_2$ then $Q = \{(j_1, j_2), (j_2, j_3), \dots, (j_n, j_{n-1})\}$

In other words $S_1 - S_2 = Q \rightarrow S_1 \oplus Q = S_2$

b) Addition operation $\oplus$

An operation between two different variables, a solution $S_2$ and a vector Q, give a solution $S_{new}$. Is an application of the permutations of the vector Q to the solution $S_2$ for obtain a new solution $S_{new}$, for example:

Whether $S_2 = \{j_1, j_2, j_3, j_4, j_5 \dots, j_n\}$ and $Q = \{(j_3, j_1), (j_4, j_3), (j_2, j_5)\}$
$S_{new} = S_2 \oplus Q$ then $S_{new} = \{j_4, j_5, j_1, j_3, j2, \dots, j_n\}$

c) Multiplication operation $\otimes$

An operation between a real number r ($r \epsilon [0,1]$) and displacement vector Q. The main role of this operation is to reduce the number of permutations of the vector Q according to the value of r. We consider a displacement vector Q of n couple of permutation

$Q = \{(c_1, c_2), (c_3, c_4), (c_5, c_6) \dots, (c_{2n-1}, x_{2n})\}$ and real number $0 \le r \le 1$.
$Q' = r \times Q$ Then $m = r \times n \le n$, $Q' = \{(c_1, c_2), (c_3, c_4) \dots, (c_{2m-1}, c_{2m})\}$

The Figure 4 illustrates the use of the operators of (7) to move it from the current solution $S_{Id}$ to a new solution $S_{new}$.

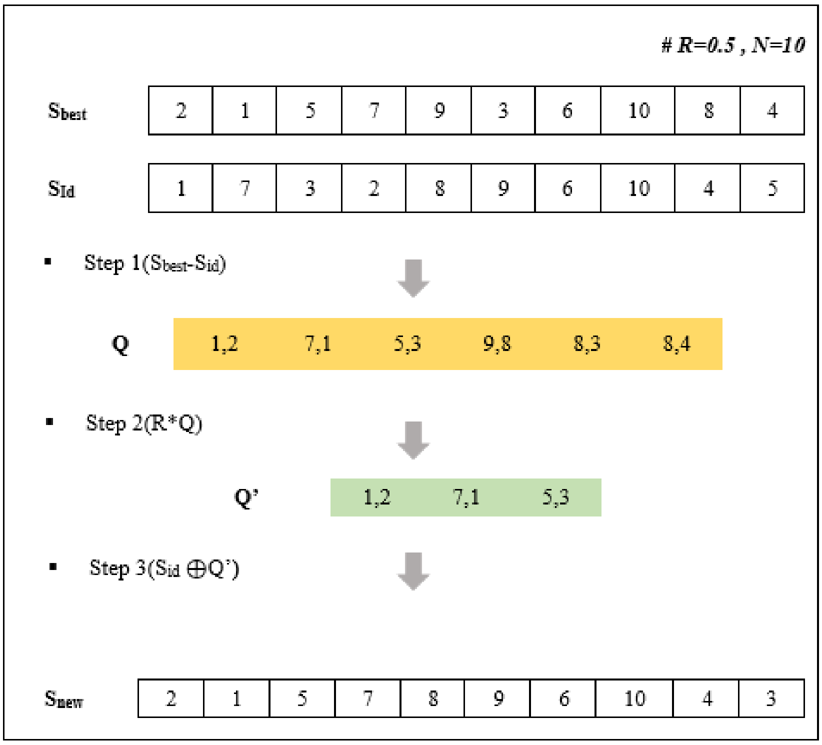



Figure 4. Illustration of PeSOA operators function.

### 4.3. The PeSOA_ FSSP Algorithm

As any algorithm, the first step in the PeSOA algorithm is to initialize some necessary parameters for good and efficient operation of the algorithm. In the proposed algorithm, a few parameters are to be set. Steps of the proposed algorithm (PeSOA) are presented below.

Step 1: Initialization
- Initialize a population size p
- Generate random population P of the initial solutions S (schedules).
- Initialize Ro2 and the global solution $S_{gbest}$

Step 2: Calculate completion time (Cmax) for all solutions in P
Step 3: Calculate $S_{best}$ the best schedule in p
Step 4: For each solution (schedules), calculate her new version using (7).
Step 5: Improving of population solutions using descent method.
Step 6: Update $S_{best}$ and $S_{gbest}$ as follows:

If (Cmax(Sid) < Cmax ($S_{best}$)) Then
$S_{best}$=Jid.

If (Cmax ($S_{best}$) < Cmax ($S_{gbest}$)) Then
$S_{gbest}$ = $S_{best}$;

Step 7: Stop iterations if stop condition is satisfed.
Return to Step 4, otherwise.

The solution of the problem is the last $S_{gbest}$. Figure 5 shows the flowchart of the proposed algorithm.

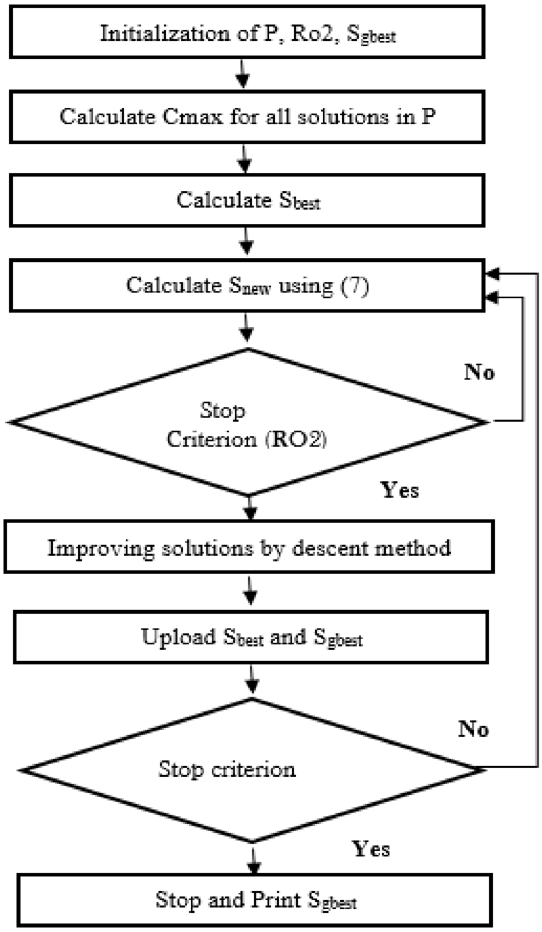



Figure 5. Flowchart of the proposed algorithm (PeSOA_FSSP)

## 5. EXPERIMENTAL RESULTS AND DISCUSSION

In this section, we evaluated the PeSOA by 31 benchmarks instances from the OR-Library [23], using c langage. The results of this experiment are extracted from a computer with Intel (R) Core (TM) 2 processor Dual CPU 2.00GHz M370@2.40GHZ 2.40 GHz and 4.00 GB of RAM. Each instances is tested 10 times with the program. Experimental results and analysis are divided into three parts. First, in section 1, we evaluate the effect of important parameters on the proposed algorithm. In section 2, we present the computational results of the PeSOA algorithm when applied to some instances of the OR-library and final section; we compare our results with other metaheuristics to study the performance and the efficiency of PeSOA.

### 5.1. Parameters of PeSOA_FSSP

The algorithm performance appear in the good setting of the parameters. So, at first we make a regulation of the values of the key parameters for the obtaining an effective adaptation. The Figure 6 shows the results of use Hel1 (100 x 10) instance of OR-library to test the effect of parameters PeSOA. This instance contains 100 jobs for 10-machine.We increase the population size of the algorithm ranging from 10 to 60, and marked the quality of the solution found. The optimal solution appeared in point 40. Therefore, we set the population size at 40 for this experiment.

In Figures 7 and 8 we used two instance Hel1(100 x 10) and ReC7 (20 x 10), to find the good value of parameter RO2. We vary the values of RO2 from 0 to 30 and marks the deviation of the length of the solution found over the length of the best solution, and time executions.At point 10, we found good solutions quality in reasonable time. So, the value of Ro2 must be fixed it 10 for this experience.

$$\text{Cost} = \text{length of the solution found} - \text{length of the optimal solution} \quad (8)$$

The values of the parameters used in this experiment are shown in Table 2.

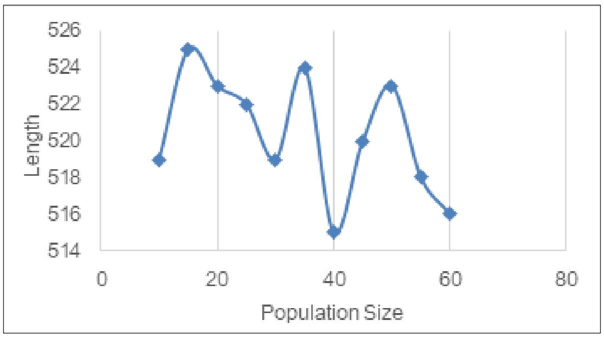



Figure 6. The impact of increasing the PeSOA population size (M)

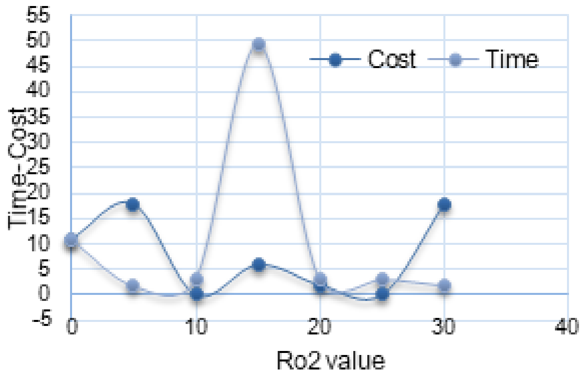



Figure 7. The impact of changing the value of $R_{O2}$ for solutions Hel1 instance

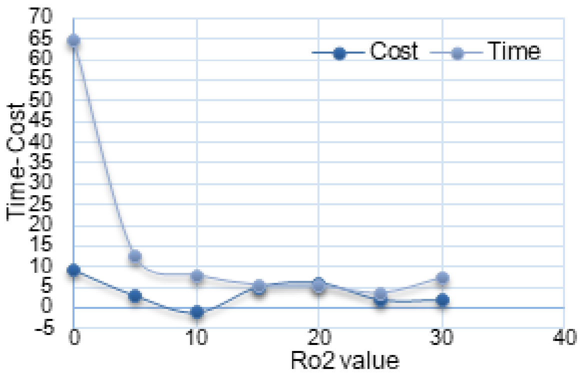



Figure 8. The impact of changing the value of RO2 for solutions ReC7 instance

Table 2. The Parameters of PeSOA

| Parameters | Meaning | Value |
|---|---|---|
| Gene | Maximum number of iterations | 1000 |
| M | Population size | 40 |
| RO2 | oxygen reserve | 10 |

**5.2. Computational results**

The numerical results of this adaptation are presented in Table 3. The first column contains the name of the instance ‘Instance’, the column ‘n×m’ represents n job for m machine. The column ‘BKS’ show the best result proposed by the other algorithms. The column ‘Best’ and ‘Worst’ presents the best results and the worst results obtained by the application of the PeSOA method for the selected instance. The columns ‘Average’ contain information on average, the column ‘PDav(%)’ denotes the percentage deviation of the average of length of solutions found over the length of the optimal solution in 10 runs and the last column ‘Time’ shows the average execution time in seconds for the 10 runs .

$$PDav(\%) = \frac{BKS - Average}{BKS} \times 100\% \quad (9)$$

Table 3 presents the computational results of PeSOA algorithm on 31 instances of OR-library. The column ‘PDav(%)’ which takes the value 0.00 ndicated in bold when all the solutions found in 10 tests are equal with the length of the best known solution, and the value less than 0.00 in bold-blue, if the average of solutions found on all the tests is less than the length of the best known solution, in this case the solutions found are better than the best known solution. the rest; 26% of the values, are less than 0.5%, which means that the best solution found, on the 10 tests, approximates less than 0.5% of the best-known solution .the 16% of the values, are more 0.5% , which means that the PeSOA does not reach the best-known solution or near solution to this solution in the 10 test. The numerical values in Table 3 show that PeSOA is of great ability to provide high quality solutions in reasonable time.

Table 3. Numericals results obtain

| Instance | m x n | BKS | Best | Worst | Average | PDav(%) | Time(S) |
|---|---|---|---|---|---|---|---|
| Car1 | 11 x 5 | 7038 | 7038 | 7038 | 7038 | **0.00** | 0.11 |
| Car2 | 13 x4 | 7166 | 7166 | 7166 | 7166 | **0.00** | 0.07 |
| Car3 | 12 x 5 | 7312 | 7312 | 7312 | 7312 | **0.00** | 0.08 |
| Car4 | 14 x 4 | 8003 | 8003 | 8003 | 8003 | **0.00** | 0.08 |
| Car5 | 10 x 6 | 7720 | 7720 | 7720 | 7720 | **0.00** | 0.07 |
| Car6 | 8 x 9 | 8505 | 8505 | 8505 | 8505 | **0.00** | 0.07 |
| Car7 | 7 x 7 | 6590 | 6590 | 6590 | 6590 | **0.00** | 0.04 |
| Car8 | 8 x 8 | 8366 | 8366 | 8366 | 8366 | **0.00** | 0.06 |
| Hel1 | 100 x 10 | 516 | 515 | 516 | 515,5 | **-0.09** | 40.58 |
| Hel2 | 20 x 10 | 136 | 135 | 136 | 135,5 | **-0.36** | 7.21 |
| ReC1 | 20 x 5 | 1247 | 1247 | 1247 | 1247 | **0.00** | 0.64 |
| ReC3 | 20 x 5 | 1109 | 1109 | 1109 | 1109 | **0.00** | 1.17 |
| ReC5 | 20 x 5 | 1242 | 1245 | 1245 | 1245 | 0.24 | 0.14 |
| ReC7 | 20 x 10 | 1566 | 1566 | 1566 | 1566 | **0.00** | 2.26 |
| ReC9 | 20 x 10 | 1537 | 1537 | 1537 | 1537 | **0.00** | 2.19 |
| ReC11 | 20 x 10 | 1431 | 1431 | 1431 | 1431 | **0.00** | 1.97 |
| ReC13 | 20 x 15 | 1930 | 1930 | 1936 | 1933 | 0.15 | 9.45 |
| ReC15 | 20 x 15 | 1950 | 1950 | 1950 | 1950 | **0.00** | 71.58 |
| ReC17 | 30 x 15 | 1902 | 1902 | 1902 | 1902 | **0.00** | 5.17 |
| ReC19 | 30 x 10 | 2093 | 2099 | 2099 | 2099 | 0.29 | 17.89 |
| ReC21 | 30 x 10 | 2017 | 2046 | 2046 | 2046 | 1.44 | 15.45 |
| ReC23 | 30 x 10 | 2011 | 2020 | 2020 | 2020 | 0.45 | 88.46 |
| ReC25 | 30 x 15 | 2513 | 2525 | 2530 | 2527,5 | 0.57 | 325.23 |
| ReC27 | 30 x 15 | 2373 | 2379 | 2384 | 2381,5 | 0.36 | 80.30 |
| ReC29 | 30 x 15 | 2287 | 2298 | 2298 | 2298 | 0.48 | 439.38 |
| ReC31 | 50 x 10 | 3045 | 3053 | 3061 | 3057 | 0.39 | 86.88 |
| ReC33 | 50 x 10 | 3114 | 3118 | 3118 | 3118 | 0.13 | 140.56 |
| ReC35 | 50 x 10 | 3277 | 3277 | 3277 | 3277 | **0.00** | 04.58 |
| ReC37 | 75 x 20 | 4951 | 5062 | 5080 | 5071 | 2.42 | 468.70 |
| ReC39 | 75 x 20 | 5087 | 5180 | 5184 | 5182 | 1.86 | 159.33 |
| ReC41 | 75 x 20 | 4960 | 5079 | 5118 | 5098.5 | 2.79 | 236.54 |

### 5.3. Comparison with other metaheuristic

To further present the performance of the proposed PeSOA, a comparison was made between the proposed algorithm with results of EM, and GA listed in the work of Yuan, Henequin, Wang, and Gao (2006) [24] and PSO-EDA in the work of Hongcheng Liu et al. (2011) [25] and GBA in the work F. Sayoti, M.E.Riffi ) (2016) [26], using different instances of OR-library. The results are listed in Table 4.

Table 4. Comparison of PeSOA with different algorithms: GA, EM, GBA and PSO-EDA

| | ***Problem*** | | ***PDav (%)*** | | | | |
|---|---|---|---|---|---|---|---|
| Instance | n × m | BKS | PeSOA | EM | GA | PSO-EDA | GBA |
| Car1 | 11 x 5 | 7038 | 0 | 0 | 0 | 0 | 0 |
| Car2 | 13 x4 | 7166 | 0 | 0 | 0 | 0 | 0 |
| Car3 | 12 x 5 | 7312 | 0 | 0 | 1.19 | 0 | 0 |
| Car4 | 14 x 4 | 8003 | 0 | 0 | 0 | 0 | 0 |
| Car5 | 10 x 6 | 7720 | 0 | 0 | 0 | 0 | 0 |
| Car6 | 8 x 9 | 8505 | 0 | 0 | 0 | 0 | 0 |
| Car7 | 7 x 7 | 6590 | 0 | 0 | 0 | 0 | 0 |
| Car8 | 8 x 8 | 8366 | 0 | 0 | 0 | 0 | 0 |
| Hel1 | 100 x 10 | 516 | -0.09 | - | - | - | 2.9263 |
| Hel2 | 20 x 10 | 136 | -0.36 | - | - | - | 1.7647 |
| Rec1 | 20 x 5 | 1247 | 0 | 4.41 | 6.96 | 0.096 | 0.1764 |
| Rec3 | 20 x 5 | 1109 | 0 | 1.98 | 4.45 | 0.036 | 0.252 |
| Rec5 | 20 x 5 | 1242 | 0.24 | 2.01 | 3.82 | 0.242 | 0.3059 |
| Rec7 | 20x10 | 1566 | 0 | 3.70 | 5.31 | 0 | 0.8812 |
| Rec9 | 20x10 | 1537 | 0 | 3.97 | 4.73 | 0.202 | 2.0104 |
| Rec11 | 20x10 | 1431 | 0 | 4.05 | 7.39 | 0.126 | 2.0614 |
| Rec13 | 20 x15 | 1930 | 0.15 | 4.87 | 5.97 | 0.223 | 2.0362 |
| Rec15 | 20 x 15 | 1950 | 0 | 3.79 | 4.29 | 0.303 | 2.3128 |
| Rec17 | 30 x 15 | 1902 | 0 | 5.57 | 6.08 | 0.289 | 2.3554 |
| Rec19 | 30 x 10 | 2093 | 0.29 | 5.97 | 6.07 | 0.612 | 2.8858 |
| Rec21 | 30 x 10 | 2017 | 1.44 | 3.22 | 6.07 | 1.408 | 2.6425 |
| Rec23 | 30 x 10 | 2011 | 0.45 | 5.97 | 7.46 | 0.597 | 3.0283 |

This comparison is based on the percentage deviation, i.e. the quality of solution found compared to best-known solution according in OR-library. Which is confirmed by Figure 9 indicating the lower curve associated with PeSOA. Hence, PeSOA is more appropriate to be adapted to solve FSSP and it can give good results.

In Table 4, the experimental results of PeSOA algorithm are compared with the results of the EM, GA, PSO-EDA and GBA methods. these results clearly show that PeSOA outperforms the other EM, GA, PSO-EDA and GBA algorithms in resolution of all twenty-two tested instances .The proposed PeSOA algorithm gets $\frac{17}{22}$ Best solutions while EM / GA / PSO-EDA / GBA only gets $\frac{8}{20}/\frac{7}{20}/\frac{9}{20}/\frac{8}{22}$ of the best solutions among 22 or 20 instances. Furthermore, we find that the average of PDav (%) for the EM / GA / PSO-EDA / GBA algorithms is equal to 2,475 / 3,489 /0,206 / 1,165, while our algorithm PeSOA is equal to 0,096. Figure 9 shows the study of the PDav (%) variation between the PeSOA algorithm and the EM, GA, PSO-EDA and GBA algorithms for the twelve instances of different sizes. In Figure 9, the lower curve associated with the PeSOA algorithm is better, in terms of the quality of the solution. This can explained the performance and efficiency of FSSJ resolution PeSOA algorithm in terms of the quality of the solutions found.

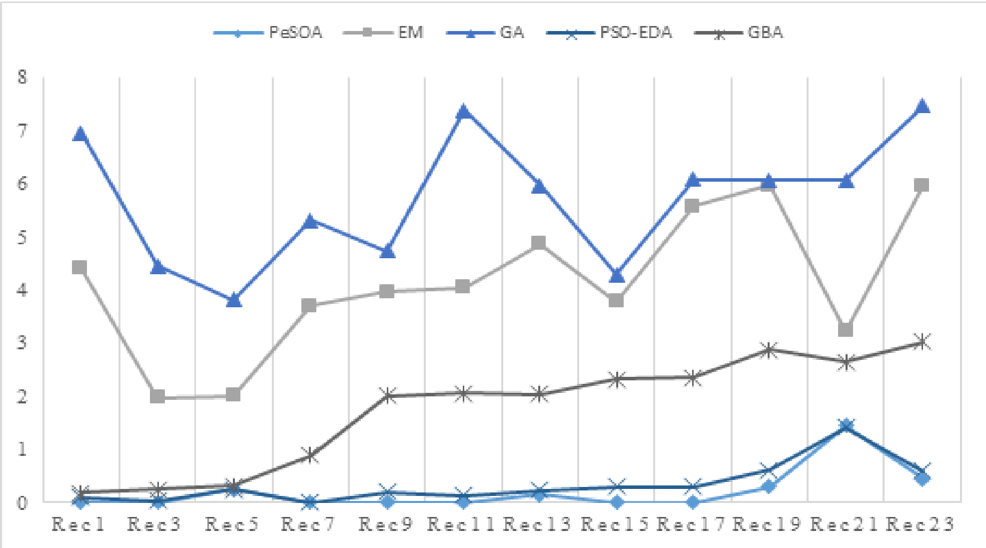



Figure 9. The percentage deviation for the PeSOA and other metaheuristics

## 6. CONCLUSION

In this paper, we proposed a swarm-intelligence algorithm, so-called PeSOA to solve flow shop scheduling problem. This adaptation consists of adding a set of changes to the standard version of PeSOA regarding position representation and its update equation, as well as the operators of this equation. Besides, a local search mechanism was used to improve research strategy. The PeSOA algorithm was applied on a set of benchmark instances of FSSP from the OR-Library. It has been shown that the effectiveness as well as performance of PeSOA, has reached that of the other techniques such as GA, EM, GBA and PSO-EDA in terms of the quality of solutions and the computing time. Future work is to extend the PeSOA application for other combinatorial optimization problems such as other scheduling models, logistic network models, vehicle routing models, etc. We will also try to improve the PeSOA algorithm in order to obtain even better results by hybridization with other methods or by modifications of the parameters or the algorithm operators.

## REFERENCES

[1] N. Gupta, et al., "Flowshop scheduling research after five decades," *European Journal of Operational Research*, vol. 169, no. 3, pp. 699-711, 2006.

[2] M. R. Garey, et al., "The complexity of flowshop and jobshop scheduling," *Mathematics of operations research*, vol. 1, no. 2, pp. 117-129, 1976.

[3] Y. Saji, et al., "Discrete bat-inspired algorithm for travelling salesman problem," *In Second World Conference on Complex Systems, 2014. WCCS 2014, IEEE,* pp. 28-31, 2014.

[4] M.E. Riffi, et al., "Incorporating a modified uniform crossover and 2-exchange neighborhood mechanism in a discrete bat algorithm to solve the quadratic assignment problem," *Egyptian Informatics Journal*, vol. 18, pp. 221–232, 2017.

[5] X. H. Shi, et al., "Particle swarm optimization-based algorithms for TSP and generalized TSP," *Information processing letters*, vol. 103, no. 5, pp. 169-176, 2007.

[6] D. Karaboga and B. Gorkemli, " A combinatorial artificial bee colony algorithm for traveling salesman problem," *In International Symposium on Innovations in Intelligent Systems and Applications, 2011. IEEE*, pp. 50-53, 2011.

[7] A. Bouzidi, and M. E. Riffi, "Improved CSO to Solve the TSP," *In International Conference on Advanced Intelligent Systems for Sustainable Development, 2018. Springer, Cham*, pp. 252-260, 2018.

[8] A. Bouzidi, et al., "Cat swarm optimization for solving the open shop scheduling problem," *Journal of Industrial Engineering International*, vol. 15, no. 2, pp. 367-378, 2019.

[9] A. Agharghor, et al., "Improved hunting search algorithm for the quadratic assignment problem," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 14, no. 1, pp. 143-154, 2019.

[10] A. Hossam, et al., "Elephants herding optimization for solving the travelling salesman problem," *In International Conference on Advanced Intelligent Systems for Sustainable Development, Springer, Cham*, pp. 122-130, 2018.

[11] S. Bouzidi, and M. E. Riffi, "Discrete swallow swarm optimization algorithm for travelling salesman problem," *In Proceedings of the 2017 International Conference on Smart Digital Environment*, ACM, pp. 80-84, 2017.

[12] F. Sayoti, et al., "Optimization of makespan in job shop scheduling problem by golden ball algorithm," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 4, no. 3, pp. 542-547, 2016.

[13] K. Ruttanateerawichien, et al., "An improved golden ball algorithm for the capacitated vehicle routing problem," *In Bio-Inspired Computing-Theories and Applications. Springer, Berlin, Heidelberg*, pp. 341-356, 2014.

[14] A. Ouaarab, et al., "Discrete cuckoo search algorithm for the travelling salesman Problem," *Neural Computing and Applications*, vol. 24, no. 7-8, pp. 1659-1669, 2014.

[15] S.C.B. Semlali, et al., "Parallel hybrid chicken swarm optimization for solving the quadratic assignment problem," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 9, no. 3, pp. 2064-2074, 2019.

[16] H. Ye, et al., "An Improved Chicken Swarm Optimization for TSP," in *International Conference on Applications and Techniques in Cyber Security and Intelligence. Springer, Cham*, pp. 211-220, 2019.

[17] D. P. Samdean, et al., "Flower Pollination Algorithm (FPA) to Solve Quadratic Assignment Problem (QAP) ," *Contemporary Mathematics and Applications*, vol. 1, no. 2, pp. 58-67, 2019.

[18] I. Mzili and M.E. Riffi, "Discrete penguins search optimization algorithm to solve the traveling salesman problem," *Journal of Theoretical & Applied Information Technology*, vol. 72, no. 3, 2015.

[19] I. Mzili, et al., "Hybrid Penguins Search Optimization Algorithm and Genetic Algorithm Solving Traveling Salesman Problem," *In International Conference on Advanced Information Technology, Services and Systems, Springer, Cham*, pp. 461-473, 2017.

[20] I. Mzili, et al., "Penguins search optimization algorithm to solve quadratic assignment problem," *In Proceedings of the 2nd international Conference on Big Data, Cloud and Applications, 2017, BDCA'17. ACM*, pp. 1-6, 2017.

[21] S. M. Johnson, "Optimal two-and three-stage production schedules with setup times included," *Naval research logistics quarterly*, vol. 1, pp. 61-68, 1954.

[22] Y. Gheraibia, and A. Moussaoui, "Penguins search optimization algorithm (PeSOA)," *In International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, 2013,* Berlin 2013, Springer Heidelberg, pp. 222-231, 2013.

[23] J.E. Beasley, "OR-Library: distributing test problems by electronic mail," *Journal of the operational research society*, vol. 41, pp. 1069-1072, 1990.

[24] K.Yuan, et al., "A new heuristic-EM for permutation flowshop scheduling," *IFAC Proceedings Volumes,* vol. 39, pp. 33-38, 2006.

[25] H. Liu, et al., "A hybrid particle swarm optimization with estimation of distribution algorithm for solving permutation flowshop scheduling problem," *Expert Systems with Applications,* vol. 38, pp. 4348-4360, 2011.

[26] F. Sayoti, and M.E. Riffi, "Golden ball algorithm for solving flow shop scheduling problem," *The International Journal of Interactive Multimedia and Artificial Intelligence*, vol. 4, no. 1, pp. 15-18, 2016.