DISCRETE PENGUINS SEARCH OPTIMIZATION ALGORITHM TO SOLVE THE TRAVELING SALESMAN PROBLEM

1ILYASS MZILI, 2MOHAMMED ESSAID RIFFI

1,2Computer Science Dept Faculty of Sciences, Chouaib Doukkali University El Jadida, Morocco.
E-mail: 1 dr.mzili.ilyass@gmail.com, 2 sad5005@yahoo.fr

ABSTRACT

The traveling salesman problem (TSP) is an NP-hard combinatorial optimization problem that can be solved by finding optimal solutions. In this work we present an adaptation of the metaheuristic “Penguins Search Optimization Algorithm” (PeSOA) depending on the Penguins hunting strategies. The results show that the proposed adaptation which is favorably compared with the methods proposed in the literature, gives very good results in a reasonable time.

Keywords: Metaheuristic, Traveling Salesman Problem, PeSOA, Combinatorial Optimization.

1. INTRODUCTION

The traveling salesman problem (TSP) ([1]) is one of the most ancient and extensively treated problems in the operational research. This problem can be formulated as follows: A commercial traveler should visit a number of cities; he must start and end his journey in the same city of departure but visiting each of the other cities only once. The objective is to find the tour that has the minimal total distance traveled.

The importance of this problem appears in many application areas such as telecommunications ([2]), transport and logistics ([3]) or in the industry ([4]). Since the 19th century, the traveling salesman problem (TSP) has been ranked among the most difficult problems. But now we can resolve it with the heuristic appearance, for example, Local search ([5-6]), simulated annealing ([7-8]), and tabu search ([9-10]) and by the appearance of metaheuristic, Such as, genetic algorithm([11-13]), Ant Colony System ([14-15]), particle swarm optimization([16-18]), bee colony optimization([19-21]), cat swarm optimization([22]) and harmony search algorithm([23]). In this paper, we propose to solve this problem using the algorithm of PeSOA ([24]). In this research, the first part will be devoted to the description of TSP, the second provides an overview of the algorithm of PeSOA, the third part is an adaptation of the algorithm of PeSOA or TSP, The fourth discusses the results of tests using instances of TSPLIB and finally a general conclusion.

2. THE TRAVELING SALESMAN PROBLEM (TSP).

The traveling salesman problem (TSP) is a classic optimization problem which was mentioned for the first time in 1930 by the Viennese mathematician Karl Menger, it’s about finding the shortest way passing only once by a limited number of cities N, And returning to the starting point.

The traveling salesman problem (TSP) is composed of a set of variables:

\[ C_{ij} : \text{The distance between two cities i and j.} \]

\[ N: \text{Number of cities} \]

\[ X_{ij} : \text{Binary variable that takes the value 1 if the city i is visited immediately before the city j else, this variable takes the value 0.} \]

The problem is to minimize the length of the Hamiltonian cycle. The objective function is:

\[ \text{Minimize} : \quad Z = \sum_{i=1}^{n} \sum_{j=1}^{n} C_{ij} X_{ij} \quad (1) \]

The constraints are:

\[ \sum_{j=1}^{n} X_{ij} = 1 \quad i = 1, \ldots, n \quad (2) \]
\[ \sum_{j=1}^{n} x_{ij} = 1 \quad j = 1, \ldots, n \] (3)

\[ \sum_{i=j}^{s} a_{ij} \geq 1 \quad s = 1, \ldots, n \] (4)

\[ x_{ij} \in \{0, 1\} \quad \forall i, \forall j \]

The constraints (2) and (3) allow to visit all cities exactly one and only once. The third constraint (4) prohibits solutions which will form sub-rounds; it is generally called removal sub-rounds constraint.

3. THE PESOA ALGORITHM.

PeSOA is a method for optimizing nonlinear systems developed by Youcef Gheraibia and Abdelouahab Moussaoui in 2013 to solve optimization problems ([21]), this is a simplified model of social relations, which is the hunting of the Penguins. The hunting strategy of the Penguins is a collaborative work of effort and timing, they benefit of their dives by optimizing the overall energy in the process of collective hunting and nutrition.

In the PeSOA algorithm each penguin is represented by its position and the number of fish consumed. The distribution of penguins is based on probabilities of existence of fish in a position. The penguins are divided into groups and begin research in random positions, after a fixed number of dives, penguins return to the ice and begin to share information such as position and the amount of nutrients found.

Penguins of one or more groups that have some food follow in the next dive penguins that chased a lot of fish. Each penguin that represents solution is based on probabilities of existence of fish in a position. The penguins are divided into groups and begin research in random positions, after a fixed number of dives, penguins return to the ice and begin to share information such as position and the amount of nutrients found.

4. ADAPTATION OF THE PENGUINS SEARCH OPTIMIZATION ALGORITHM TO SOLVING THE TRAVELLING SALESMAN PROBLEM.

To adapt the method PeSOA for the traveling salesman problem, with preservation of the philosophy of the method, it requires firstly to do a parameter adaptation of PeSOA with TSP and secondly rewrite the equation database system of PeSOA(equation (5)) to improve the position of the penguin to find optimal solutions, which makes the equation(5) as follows :

\[ X_{new} = X_{id} \oplus \text{rand}() \otimes (X_{best} - X_{id}) \] (6).

As:

\textbf{Position}: This is the solution represented by the Hamiltonian path.

\[ X_{new} \] : The new solution.

\[ X_{id} \] : The current solution.

\[ X_{best} \] : The best solution in the population.

4.1 Algebra of Pesoa

Before we start to define the operations, we define variables X and X' as two positions / path of the salesman, and Q as a set of links (x,y) to perform all permutations that may be appointed: displacement.

Example: \( Q = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\} \) with \( x_i \neq y_j \)
The addition $\oplus$:
This applies between one position and displacement and returns to a position. The result is obtained by successive permutations of the elements X taking into account those of Q. It is only used during the update of the position.
Example: $X = \{1 2 3 4 5 6\}$ and $Q = \{(1, 2) (4, 5)\}$
$X' = X \oplus Q$  Return $X' = \{2 1 3 5 4 6\}$.

The multiplication $\otimes$:
This is done between a real k and displacement Q. Different cases are considered according to the real.

- If $0 < k < 1$ then: $K \otimes Q = E(k \otimes |V|)$
  where $|V|$ is the number of elements V and $E(x)$ is the entire upper part of $x$.
- If $k = 1$ then we separate integer from decimal where $k = n + x$ whether $n$ and $x$ correspond respectively to the integer and the decimal parts of $k$. We thereby reduce each party to the previous cases.
- If $k < 0$ then: $K \otimes Q = |k| \otimes Q$ returns to the previous case of a value of $|k|$.

Example: $k = 0.5$ and $Q = \{(1, 2) (4, 5)\}$
$K \otimes Q$  return $Q = \{(1, 2)\}$.

The subtraction $-$:
The subtraction is performed between two positions and returns a set of permutations that can go from the first to the second position.
Example: whether $X_1 = \{1 2 3 4 5\}$ and $X_2 = \{1 2 4 3 5\}$ then $Q = X_1 - X_2$
Return $Q = \{(2, 3) (4, 5)\}$.

4.2 Algorithm PeSOA to solve TSP.
The following pseudo code describes how the algorithm PeSOA was applied to solve our problem TSP.

Algorithm (2) : pseudo code PeSOA for TSP

Initialize the population $P$.
Initialization of the parameters:
- Gene: number of generations.
- RO2: oxygen reserves.
- BestSol: best solution.
Calculate best individual of population BestGp
While (iteration $<$ gene) do
  For each individual do
    While (RO2 $>$ 0) do
      Calculate new solutions $x_i$ using equation (6).
      Choose the best solutions of $x_i$.
    End while
  End for
  //Update BestGp and BestSol
  If ($f(x_id) < f(\text{BestGp})$) Then
    BestGp = xid.
  End If
  If ($f(\text{BestGp}) < f(\text{BestSol})$) Then
    BestSol = BestGp;
  End If
End if
End while.

- Number of generations (Gene): number of iterations needed to find the optimal solution.
- The population (P): a set of solutions to the problem. The size of the initial population is left to the programmer, it should be between 80 and 100 individuals and will often be ample, both for the quality of the solutions and for the running time of our algorithm.
- The variable oxygen supply (RO2): an integer shows the number of updating a solution by using the equation (6), its value must be greater than 0. The best value for this variable is 5 (figure 1).

5. EXPERIMENTAL RESULTS.
The implementation of the algorithm PeSOA in C and the run on multiple instances of TSP using a pc with Intel (R) Core (TM) 2 Duo CPU 2.00GHz M370@2.40GHZ 2.40GHZ and 4.00 Go of RAM. Each instance runs 10 times.
The first feature of our program is to read the TSP files and calculate distances between cities then storing these data in a distance matrix. The time of these operations is not included in the execution time of our algorithm. Therefore our program generates a random population of initial solutions. There are three keys parameters that influenced on the results of the performance of our algorithm: the size of the population (NP), generation number (gene) and reserve oxygen (RO2). The parameter values are obtained after completing 10 experiments on each variable in order to find the best value for a very good running time of our algorithm. The following table shows the best values found and used for these parameters:

Table 1: Value Of Uses Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>RO2</td>
<td>5</td>
</tr>
<tr>
<td>Gene</td>
<td>800000</td>
</tr>
<tr>
<td>NP</td>
<td>80</td>
</tr>
</tbody>
</table>
The figure below illustrates the supply of oxygen with time of PeSOA algorithm for instance, the TSPLIB library Berlin52 and Att48. This figure shows that when we increase the value RO2 the time increases.

![Figure1: The Relationship between the Average Execution Time and the value of RO2.](image)

Table 2 presents the numerical results. Where the first column contains the name of the proceeding (Instance), the second shows the number of node (Nbr node), the third shows the best result found by the documentation TSPLIB (Optimum), the fourth presents the best result (BestR) obtained by the application of the algorithm PeSOA, the fifth contains the worst results (WorstR), the sixth the percentage error (Err) and the last one is the average execution time (Tps).

![Image1](image)

Table 2: Numericals Results Obtained

<table>
<thead>
<tr>
<th>Instance</th>
<th>Nbr node</th>
<th>Optimum</th>
<th>BestR</th>
<th>WorstR</th>
<th>Err(%)</th>
<th>Tps(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>att48</td>
<td>48</td>
<td>33522</td>
<td>33522</td>
<td>33522</td>
<td>0</td>
<td>4,07</td>
</tr>
<tr>
<td>berlin52</td>
<td>52</td>
<td>7542</td>
<td>7542</td>
<td>7542</td>
<td>0</td>
<td>0,42</td>
</tr>
<tr>
<td>bier127</td>
<td>127</td>
<td>118282</td>
<td>118282</td>
<td>119693</td>
<td>0,59</td>
<td>550,28</td>
</tr>
<tr>
<td>eil51</td>
<td>51</td>
<td>426</td>
<td>426</td>
<td>427</td>
<td>0,11</td>
<td>2,5</td>
</tr>
<tr>
<td>eil76</td>
<td>76</td>
<td>538</td>
<td>538</td>
<td>538</td>
<td>0</td>
<td>24,42</td>
</tr>
<tr>
<td>eil101</td>
<td>101</td>
<td>629</td>
<td>629</td>
<td>630</td>
<td>0,07</td>
<td>209,54</td>
</tr>
<tr>
<td>ch130</td>
<td>130</td>
<td>6110</td>
<td>6110</td>
<td>6172</td>
<td>0,50</td>
<td>928,16</td>
</tr>
<tr>
<td>ch150</td>
<td>150</td>
<td>6528</td>
<td>6528</td>
<td>6573</td>
<td>0,34</td>
<td>1968</td>
</tr>
<tr>
<td>pr76</td>
<td>76</td>
<td>108159</td>
<td>108159</td>
<td>108159</td>
<td>0</td>
<td>20,63</td>
</tr>
<tr>
<td>pr107</td>
<td>107</td>
<td>44303</td>
<td>44303</td>
<td>44387</td>
<td>0,09</td>
<td>232,96</td>
</tr>
<tr>
<td>pr144</td>
<td>144</td>
<td>58537</td>
<td>58537</td>
<td>58537</td>
<td>0</td>
<td>130,78</td>
</tr>
<tr>
<td>pr124</td>
<td>124</td>
<td>59030</td>
<td>59030</td>
<td>59030</td>
<td>0</td>
<td>153,92</td>
</tr>
<tr>
<td>kroa100</td>
<td>100</td>
<td>21282</td>
<td>21282</td>
<td>21282</td>
<td>0</td>
<td>114,04</td>
</tr>
<tr>
<td>krob100</td>
<td>100</td>
<td>22141</td>
<td>22141</td>
<td>22199</td>
<td>0,13</td>
<td>113,46</td>
</tr>
<tr>
<td>kroc100</td>
<td>100</td>
<td>20749</td>
<td>20749</td>
<td>21010</td>
<td>0,62</td>
<td>75,9</td>
</tr>
<tr>
<td>krod100</td>
<td>100</td>
<td>21294</td>
<td>21294</td>
<td>21389</td>
<td>0,22</td>
<td>581,05</td>
</tr>
<tr>
<td>kroe100</td>
<td>100</td>
<td>22068</td>
<td>22068</td>
<td>22106</td>
<td>0,08</td>
<td>311,39</td>
</tr>
<tr>
<td>rat99</td>
<td>99</td>
<td>1211</td>
<td>1211</td>
<td>1211</td>
<td>0</td>
<td>322,9</td>
</tr>
<tr>
<td>st70</td>
<td>70</td>
<td>675</td>
<td>675</td>
<td>675</td>
<td>0</td>
<td>13,26</td>
</tr>
<tr>
<td>lin105</td>
<td>105</td>
<td>14379</td>
<td>14379</td>
<td>14379</td>
<td>0</td>
<td>74,38</td>
</tr>
<tr>
<td>u159</td>
<td>159</td>
<td>42080</td>
<td>42080</td>
<td>42396</td>
<td>0,3</td>
<td>564,95</td>
</tr>
<tr>
<td>dantzig42</td>
<td>42</td>
<td>699</td>
<td>699</td>
<td>699</td>
<td>0</td>
<td>0,72</td>
</tr>
<tr>
<td>rd100</td>
<td>100</td>
<td>7910</td>
<td>7910</td>
<td>7910</td>
<td>0</td>
<td>190,41</td>
</tr>
<tr>
<td>swiss42</td>
<td>42</td>
<td>1273</td>
<td>1273</td>
<td>1273</td>
<td>0</td>
<td>16,43</td>
</tr>
<tr>
<td>ulysses16</td>
<td>16</td>
<td>6859</td>
<td>6859</td>
<td>6859</td>
<td>0</td>
<td>73</td>
</tr>
<tr>
<td>ulysses22</td>
<td>22</td>
<td>7013</td>
<td>7013</td>
<td>7013</td>
<td>0</td>
<td>95</td>
</tr>
</tbody>
</table>
After applying the results of our algorithm on a set of instances of the problem contained in the above table we compare the performance of our algorithm to other existing algorithms: Simulated annealing (SA), Genetic Algorithm (GA), ant colony system (ACS), particle swarm optimization (PSO), Bee colony optimization, Cat swarm optimization (CSO) and Harmony search algorithm (HS).

**Table 3: Results Obtained By Different Metaheuristics**

<table>
<thead>
<tr>
<th>Method</th>
<th>Instance (optimal solution)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Pr76 (108159)</td>
</tr>
<tr>
<td>PeSOA</td>
<td>108159</td>
</tr>
<tr>
<td>SA(8]</td>
<td>109769</td>
</tr>
<tr>
<td>GA ([18])</td>
<td>117979.83</td>
</tr>
<tr>
<td>ACS([15])</td>
<td>---</td>
</tr>
<tr>
<td>PSO([18])</td>
<td>108280</td>
</tr>
<tr>
<td>BCO([21)])</td>
<td>---</td>
</tr>
<tr>
<td>CSO([22])</td>
<td>---</td>
</tr>
<tr>
<td>HS([23])</td>
<td>108159</td>
</tr>
</tbody>
</table>

**6. CONCLUSION:**

This paper has proposed an adaptation of Penguins Search Optimization algorithm to solve a combinatorial optimization problem, by redefining operators and operations. The performances of this method were proved in the experimental results applying PeSOA algorithm to solve some instances of TSP problem. The results were compared with the best optimal solutions found by other methods such as SA, HS, GA, ACO, BCO, PSO, and CSO. Future research should focus on applying the presented method to solve other real applications based on the traveling salesman problem.

**REFERENCES:**


