Hybrid Penguins Search Optimization Algorithm and Genetic Algorithm Solving Traveling Salesman Problem

Ilyass Mzili[1(✉)], Mohammed Essaid Riffi[1], and Fatiha Benzekri[2]

[1] Department of Computer Science, Faculty of Sciences, Chouaïb Doukkali University, El Jadida, Morocco
dr.mzili.ilyass@gmail.com, saidriffi2@gmail.com
[2] Department of Mathematics, Faculty of Sciences, Chouaïb Doukkali University, El Jadida, Morocco
F.Benzekri@hotmail.fr

Abstract. This paper is to present a hybrid technique of two metaheuristic algorithm Penguins Search optimization Algorithm (PeSOA) and the genetic algorithm (GA) called HPeSOA, which was proposed to solve the combinatorial optimization problem NP-hard Traveling salesman problem. In this algorithm, we improve the population of the solutions by the integration of the genetic operators, namely the crossover and the mutation in the algorithm PeSOA. The experimental results of the application of HPeSOA algorithm on the instances TSPLIB are reported and compared, with the results of Penguins Search optimization Algorithm and the genetic algorithm.

Keywords: Metaheuristic · Combinatorial optimization · Genetic algorithm · Penguins search optimization algorithm · HPeSOA · Traveling salesman problem

1 Introduction

Travelling salesman problem (TSP) [1] is one of typical NP-hard problems in combinatorial optimization problem, which wishes to visit a certain number of cities, beginning and finishing its course in the same city by visiting each other's city one and only once. It wishes to select the best tour, which minimizes the traversed total distance.

The resolution of this problem is very important because of its application in various fields such as combinative data analysis, data-processing wiring, machine sequencing, the routing of the vehicles and scheduling, planning and logistics.

In the 19th century, several researchers used heuristic and metaheuristics methods for this problem: Local Search [2], Simulated Annealing [3], Tabu Search [4], Genetic Algorithm [5, 6], Ant Colony System [7], Particle Swarm Optimization [8], Bee Colony Optimization [9], and Penguins Search Optimization Algorithm [10]. Hybrid methods: A novel hybrid penguins search optimization algorithm [11], Hybrid genetic

M. Ezziyyani et al. (eds.), *Advanced Information Technology, Services and Systems*, Lecture Notes in Networks and Systems 25, https://doi.org/10.1007/978-3-319-69137-4_41

algorithm [12], Particle Swarm Optimization with Simulated Annealing [13] and Simulated Annealing Algorithm with Greedy Search [14].

In this paper, we have added GA operators such as selection, crossover and mutation to the PeSOA algorithm in order to find a new hybrid approach HPeSOA. This approach aims to solve the traveling salesman problem. The computational results show that the proposed HPeSOA algorithm has faster convergence, higher computational precision and is more efficient to solve the TSP.

The organization of this research is as follows: In Sect. 2, presentation of the traveling salesman problem; In Sect. 3, description of metheuristics.; In Sect. 4, the adaptation of Hybrid genetic algorithm and Penguins Search Optimization Algorithm to solve traveling salesman problem; In Sect. 5, the results of tests using TSPLIB instances and finally conclusions are summarized in Sect. 6.

2 The Travelling Salesman Problem

Travelling salesman problem (TSP) is one of the most ancient and widely studied problems in combinatorial optimization. Having N cities to visit wishes to establish a tour which permit it to pass exactly once by each city and to return to its starting point for lower costs, traversing the smallest possible distance.

The distances between the cities are known. It is necessary to find the way, which minimizes the distance from displacement. This problem is modeled in linear programming in integer numbers, associating to a binary variable x_{ij} the value 1 if the city i is immediately visited before the city j otherwise, this variable takes the value 0. The cost of this visitor is noted c_{ij}. We then obtain the following model:

$$\text{Min} \quad \sum_{i,j} c_{ij} x_{ij} \tag{1}$$

The constraints are:

$$\sum_{j} x_{ij=1} \quad \forall i \neq j \tag{2}$$

$$\sum_{i} x_{ij=1} \quad \forall j \neq i \tag{3}$$

$$0 \leq x_{ij} \leq 1 \quad \forall i, \forall j \tag{4}$$

The objective function (1) is expressed as a scalar product between a real vector c and a vector of binary variables x. This model is also composed of 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 Description of Metaheuristics

3.1 The Genetic Algorithm

Genetic algorithm [15], introduced by John Holland, is a search algorithm based on the mechanism of natural selection and genetics.

Genetic algorithm first generates an initial population randomly which consist of a set of individual solutions to the problem. The best individuals of the population are those, which have a better chance to reproduce and to transmit part of their genetic heritage to the next generation. A new population of generation is then created by combining the genes of the parents. We expects that certain individuals of the new generation have the best characteristics from both parents, so they will be better and be a better solution to the problem. The new group (the new generation) is then subjected to the same selection criteria, and roughly generates its own kids. This process is repeated several times, until all the individuals have the same genetic heritage. The members of this last generation, who are usually very different from their ancestors, have genetic information g which corresponds to the best solution to the problem. The basic genetic algorithm comprises two simple operations, which are not more complicated than algebraic operations: Crossover–Mutation.

The Pseudo code of this Genetic algorithm is as follows:

```
Algorithm 1. Pseudo code of Genetic algorithm (GA) .
Initialize population
Initial probabilities of crossover (pc) and mutation
  (pm)
do
  Generate new solution by crossover and mutation
      if PC >rand then
                Crossover;
          end if
      if PM >rand then
              Mutate;
          end if
  Accept the new solution if its fitness increases.
  Select the current best for the next generation.
While (the stop condition is not satisfied)
```

3.2 Penguins Search Optimization Algorithm

Penguins Search Algorithm optimization [16] is a new class of metaheuristics proposed in 2013 by the Gheraibia and Abdelouahab Moussaoui. This algorithm is inspired by the strategy of the hunting of Penguins, which is based on the concept of collaboration between the penguins to profit from their diving. This algorithm is based on the

construction of a population, it is divided into groups, each group starts looking for food in a random position, the oxygen reserve exhausted, the groups return to the earth, and choose the best group that consumed the largest amount of fish, the best individuals in the selected group immigrate to other groups to guide them to find rich positions. This technique is repeated several times in order to find one of the best positions. Each position of penguins in the search space is a solution to the problem correlated. Penguins cooperate to determine the best position (solution) in a specific hole and level according (solution space). The movement of penguins between the groups is carried by this equation:

$$D_{new} = D_{last} + Rand \otimes |X_{best} - X_{id}| \tag{5}$$

Where rand is a random number of the distribution; the current Solution (D_{last}), the best local solution (X_{best}), the last solution (X_{id}) and the new solution (D_{new}).

The Pseudo code of this Penguins Search Optimization algorithm (PeSOA) is as follows:

```
Algorithm 2.Pseudocode Penguins Search Optimization
    algorithm.
Initialize parameters
Generate random population P
Find the best in population
While (the stop condition is not satisfied) do
     For each i individual of  P do
         While (RO2> 0) to
              Adjusts a new position using the Eq5
              Choose the best Position.
            End while
 End for.
   Select the best group Gbest
 If (f (Gbest) <f (Xbest)) then
      Xbest = Gbest
        End If
End while
Return Xbest.
```

3.3 HPeSOA Algorithm

The HPeSOA hybrid method is a combination of two Metaheuristics PeSOA and GA. The objective of this method is to improve PeSOA research techniques, by integrating techniques of the genetic method to find solutions of good quality.

The genetic algorithm operator's crossover and mutation implemented to create a population of childhoods solutions from the current solutions and the best solution in

the previous population. The PeSOA mechanism, presents to improve this population by local search techniques in order to find the best solution.

The pseudo code for this HPeSOA algorithm is as follows:

```
Algorithm 3. Pseudo code of HPeSOA
Initialize parameters
Generate random population P
Sort the P in ascending order
While (stopping  critical  is  satisfied) do
   While (stopping  critical GA  is  satisfied) do
        Generate new solution by crossover and mutation
             If (PC >rand) then
                   Crossover;
                End If
             If (PM >rand) then
                    Mutate;
                 End if
   End wihle
     Xbest=min(P) ;
     For each i individual of  P do
           While (RO2> 0) to
             Adjusts a new position using the Eq4
             Choose the best Position.
           End while
     End for.
     Gbesti = Min (Gi)
   If (f (Gbesti) <f (Xbest)) then
      Xbest = Gbesti
   End If
End while
Return Xbest.
```

4 Adaptation of HPeSOA to Solve TSP

This section presents the HPeSOA method adaptation to solve a TSP.

The adaptation of HPeSOA consists in redefining algebraic operators of genetic algorithm (crossover - mutation), penguins search optimization algorithm and the HPeSOA algorithm structures and steps.

4.1 The GA Operators (Crossover - Mutation)

Suppose that the two positions randomly selected are 4 and 7, so we get the children q1 and q2, but q1 and q2 are not legitimate. We consider the duplicated cities in q1 as superfluous cities that are 2 and 10 in Fig. 1, and duplicate cities in q2 as being devoid of cities that are 3 and 9. Then exchange for the new law q1' and q2'.

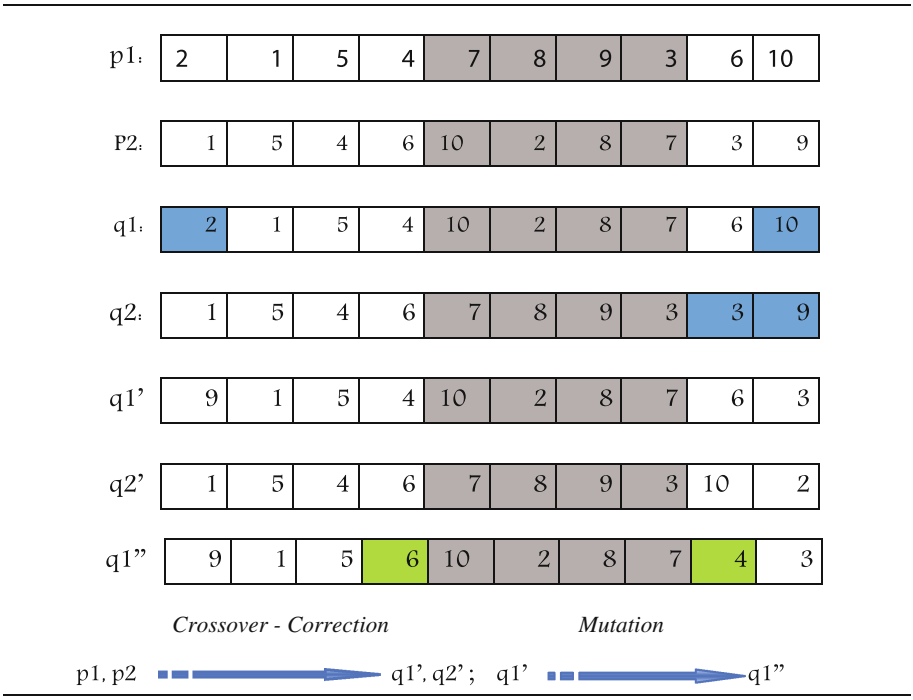

Fig. 1. An example of the GA operators (crossover – mutation).

The mutation operator creates random changes in the order of cities. By randomly selecting a solution and two positions, and then changing the first position city with the second one. In Fig. 1 an example of a mutation of a solution q1' with positions 4 and 8 gives the solution q1".

4.2 The PeSOA Operators (Substraction –Multiplication-Addition)

The new solution X_{new} is calculated by using Eq. (5), which is based on the operator's substraction, multiplication and addition: The substraction operation (−) is an operation between two solutions X_{best} and X_{id}, the result of this operation is a set of permutations Q, which can pass from the first solution to the second.

The multiplication operation (*) is an operation effected between a real k ϵ [0; 1] and a set of permutations Q, The result Q' is a part of the set Q according to the value of k.

The addition operation (+) is an operation effect between the solution X_{id} and the set of permutations Q', the result is a new solution X_{new}. This operation consists of applying permutations of Q' on X_{id} to obtain a new solution X_{new} (Fig. 2).

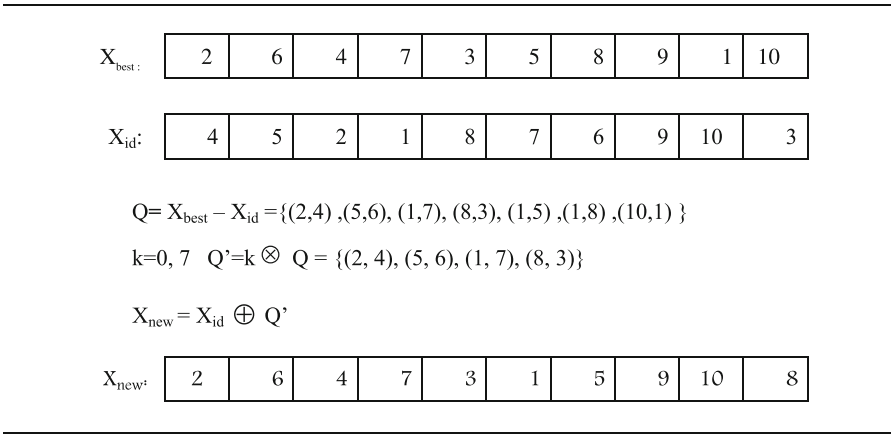

Fig. 2. A simple example of the PeSOA operators.

4.3 The Flowchart of the HPeSOA Algorithm

The Flowchart of the HPeSOA algorithm is shown in Fig. 3. The structures of two PeSOA and GA algorithms are present in the HPeSOA method to solve Travelling salesman problem.

- *1st step: Initialization.*

  The initialization of the parameters.

  Initialize a population of penguins with random positions on the dimensions D, a position represents a solution to our problem.

- *2nd step: Fitness.*

  Calculate the objective function of each solution in the population (the physical condition of each penguin in the population) $\{f(x_1), f(x_2), f(x_3), \ldots, f(x_M)\}\}$

- *3rd step: Selection.*

  Select the best solution (X_{best}) and the best group (g_{best}) of the population.

- *4th step: Crossover Operator.*

  Select solutions with PC probability of the population, and the crossover with the best solution g_{best}, then the best child of each crossover, are selected as a child solution to the second population.

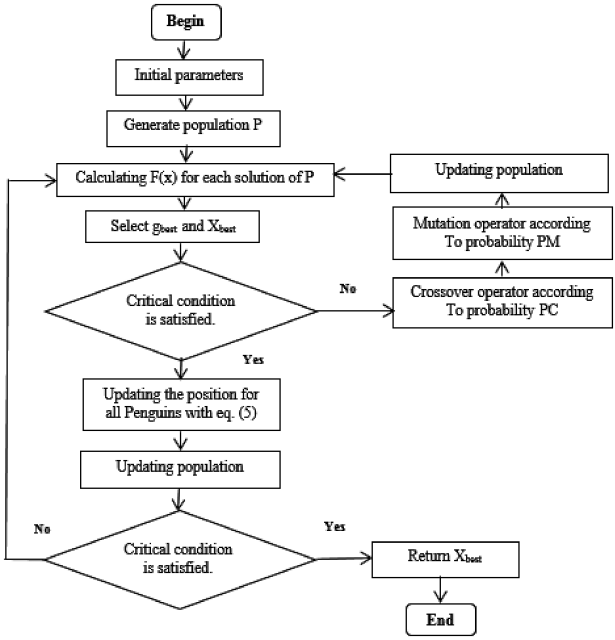

Fig. 3. Flowchart of the HPeSOA Algorithm

- *5th step: Shift Operator (mutation).*

Select solutions with PM probability of the population, and then generate solutions by exchanging two randomly selected positions within the solution.

- *6th step: update population/gbest*

Update the population by child solutions, which are generated by genetic operators and update the best group g_{best} of this population.

- *7th step: update population using PeSOA*

Update of the population, using Eq. (5) of PeSOA algorithm.

- *The last step: stopping critical*

Check the stop condition if the case leaves the loop and shows the best solution found, otherwise, return to step 2.

5 Experimental Results

We applied the HPeSOA method to the different instances of TSPLIB. This experiment implemented by language C in a PC with an Intel (R) Core (TM) 2 Duo processor 2.00 GHz M370@2.40GHZ 2.40GHZ and 4.00 GB of RAM. Each instance runs 10 times. The parameters of HPeSOA are Presented in the Table 1.

Table 1. Value of HPeSOA parameters

Parameters	Value
Gene	1000
M	80
RO2	10
PC	0.95
PM	0.1

Table 2. Numerical results obtained by HPeSOA applied to some TSP instances of TSPLIB

Instance	N	Optimum	Best	Worst	Err(%)	Tps(s)
Att48	48	33522	33522	33522	0	1.56
Berlin52	52	7542	7542	7542	0	0.39
Eil51	51	426	426	427	0.11	4.32
Eil76	76	538	538	538	0	28.28
Eil101	101	629	629	629	0	76.92
KroA100	100	21282	21282	21282	0	94.13
KroB100	100	22141	22141	22199	0.13	106.67
KroC100	100	20749	20749	20749	0	121.76
St70	70	675	675	675	0	18.53
Rd100	100	7910	7910	7910	0	84.59
Pr76	76	108159	108159	108159	0	21.33
Pr144	144	58537	58537	58537	0	159.63
Pr107	107	44303	44303	44303	0	174,82
Pr124	124	59030	59030	59030	0	104.65
Lin105	105	14379	14379	14379	0	28.96

The numerical results of this adaptation appearing in Table 2 presented as follows: The first column contains the names of the instances (Instance). The second column represents the number of cities (N). The third is the best result (Optimum) shown in the TSPLIB. The fourth and fifth columns represent the best (Best) and worst results (Worst) obtained by the HPeSOA method. The sixth is the percentage error (Err) and the last column contain the average time of execution (Tps).

$$Err = \frac{\frac{Best + Worst}{2} - Optimum}{Optimum} \times 100\% \tag{6}$$

Table 3 shows the comparison between the HPeSOA method, Penguins search optimization algorithm (PeSOA) and genetic algorithm (GA). This comparison is based on three criteria: Best solution value (Best), percentage of error (Err) of optimal solution value and average solution value of 10 runs, and the average time of execution (Tps) in second (Fig. 4).

This figure show that the error percentage of HPeSOA method is negligible with respect to GA and PeSOA (Fig. 5).

Table 3. Comparing the performance PeSOA, GA, with HPeSOA

Instance		PeSOA [10]			GA [6]			HPeSOA		
Name	Optimum	Best	Err (%)	Tps(s)	Best	Err (%)	Tps(s)	Best	Err (%)	Tps(s)
Berlin52	7542	7542	0	0.42	7542	0.24	5.10	7542	0	0.39
Eil51	426	426	0.11	2.5	426	0.63	4.59	426	0,11	4.32
Eil76	538	538	0	24.42	538	0.87	128.94	538	0	28.28
Eil101	629	629	0.07	209.54	634	1.12	226.42	629	0	76.92
KroA100	21282	21282	0	114.04	22142	4.37	48.75	21282	0	94. 13
KroC100	20749	20749	0	75.9	21122	2.77	123.76	20749	0	121.76
Pr76	108159	108159	0	20.63	108278	1.43	131.10	108159	0	21.33
Pr107	44303	44303	0.09	232.96	–	–	–	44303	0	174,82
Lin105	14379	14379	0	74.38	14741	2.67	185.90	14379	0	28.96

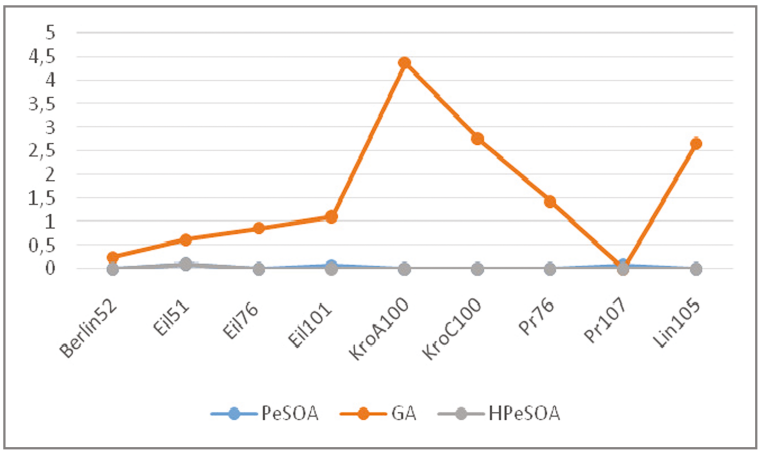

Fig. 4. Percentage of PeSOA, GA and HPeSOA methods for different instance.

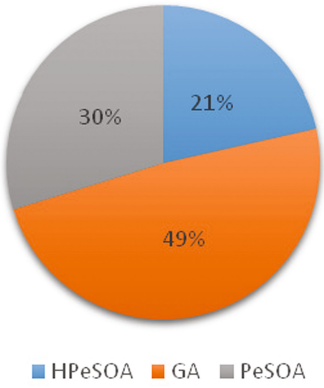

Fig. 5. The overall execution time of the different instances except Pr107 by methods PeSOA, GA and HPeSOA

This figure show that the HPeSOA method solve a set of instances at a small average time of execution, compared with PeSOA and GA methods.

6 Conclusion

In this paper, we have developed a hybrid algorithm HPeSOA that combines the advantage of the genetic algorithm and Penguins search optimization algorithm. This approach is an improvement of the two algorithms: the genetic algorithm so as not to be trapped in the local optimal and PeSOA algorithm by minimizing the time of search for the best solutions.

The result of HPeSOA shows its effectiveness to resolve some instance of TSP according PeSOA and GA, In the future; we plan to add more parameters to the proposed algorithm to make it more robust and flexible to solve the most difficult combinatorial optimization problems.

References

1. Lin, S.: Computer solutions of the traveling salesman problem. Bell Syst. Tech. J. **44**(10), 2245–2269 (1965)
2. Johnson, D.S.: Local optimization and the traveling salesman problem. In: International Colloquium on Automata, Languages, and Programming, pp. 446–461. Springer, Berlin (1990)
3. Bayram, H., Şahin, R.: A new simulated annealing approach for travelling salesman problem. Math. Comput. Appl. **18**(3), 313–322 (2013)
4. Fiechter, C.N.: A parallel tabu search algorithm for large traveling salesman problems. Discrete Appl. Math. **51**(3), 243–267 (1994)
5. Chatterjee, S., Carrera, C., Lynch, L.A.: Genetic algorithms and traveling salesman problems. Eur. J. Oper. Res. **93**(3), 490–510 (1996)
6. Ahmed, Z.H.: Genetic algorithm for the traveling salesman problem using sequential constructive crossover operator. Int. J. Biometrics Bioinf. (IJBB) **3**(6), 96 (2010)
7. Dorigo, M., Gambardella, L.M.: Ant colonies for the travelling salesman problem. Biosystems **43**(2), 73–81 (1997)
8. Wang, K.P., Huang, L., Zhou, C.G., Pang, W.: Particle swarm optimization for traveling salesman problem. In: The 2003 International Conference on Machine Learning and Cybernetics, pp. 1583–1585. IEEE Press, Xi'an(2003)
9. Wong, L.P., Low, M.Y.H., Chong, C.S.: A bee colony optimization algorithm for traveling salesman problem. In: 2008 Second Asia International Conference on Modelling & Simulation (AICMS), pp. 818–823. IEEE Press, Kuala Lumpur (2008)
10. Mzili, I., Riffi, M.E.: Discrete penguins search optimization algorithm to solve the traveling salesman problem. J. Theor. Appl. Inf. Technol. **72**(3), 331–336 (2015)
11. Mzili, I., Bouzidi, M., Riffi, M.E.: A novel hybrid penguins search optimization algorithm to solve travelling salesman problem. In: 2015 Third World Conference on Complex Systems (WCCS), pp. 1–5. IEEE Press, Marrakech (2015)
12. Yugay, O., Kim, I., Kim, B., Ko, F.I.: Hybrid genetic algorithm for solving traveling salesman problem with sorted population. In: Third International Conference on Convergence and Hybrid Information Technology(ICCIT 2008), pp. 1024–1028. IEEE Press, Busan (2008)
13. Fang, L., Chen, P., Liu, S.: Particle swarm optimization with simulated annealing for TSP. In: Proceedings of the 6th WSEAS International Conference on Artificial Intelligence, Knowledge Engineering and Data Bases (AIKED 2007), Corfu Island, Greece, pp. 206–210 (2007)
14. Geng, X., Chen, Z., Yang, W., Shi, D., Zhao, K.: Solving the traveling salesman problem based on an adaptive simulated annealing algorithm with greedy search. Appl. Soft Comput. **11**(4), 3680–3689 (2011)

15. Holland, J.H.: Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence. MIT Press, Cambridge (1992)
16. Gheraibia, Y., Moussaoui, A.: Penguins search optimization algorithm (PeSOA). In: International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, pp. 222–231. Springer, Heidelberg (2013)