Elephants Herding Optimization for Solving the Travelling Salesman Problem

Chapter · January 2019

DOI: 10.1007/978-3-030-12065-8_12

Anass Hossam
2 PUBLICATIONS **15** CITATIONS
SEE PROFILE

Abdelhamid Bouzidi
Université Chouaib Doukkali
25 PUBLICATIONS **211** CITATIONS
SEE PROFILE

Mohammed Essaid Riffi
Université Chouaib Doukkali
65 PUBLICATIONS **597** CITATIONS
SEE PROFILE

Elephants Herding Optimization for Solving the Travelling Salesman Problem

Anass Hossam(✉), Abdelhamid Bouzidi, and Mohammed Essaid Riffi

Laboratory LAROSERI, Department of Computer Science,
Faculty of Science, Chouaib Doukkali University, EI Jadida, Morocco
anass.hossam@gmail.com,
mr.abdelhamid.bouzidi@gmail.com, said@riffi.fr

Abstract. This paper proposes a novel metaheuristic called Elephant Herding Optimization (EHO) to solve the Travelling Salesman Problem (TSP), which is a combinatorial optimization problem classified as NP-Hard. The EHO algorithm is bio-inspired from the natural herding behavior of elephants groups, which proved its efficiency to solve continued optimization problems. To extend the application of this algorithm, we had proposed a novel adaptation of the EHO by respecting the natural herding behavior of elephants. To test the efficiency of our proposal adaptation, we applied the adapted EHO algorithm on some benchmark instances of TSPLIB. The obtained results shows the excellent performance of the proposed method.

Keywords: Travelling Salesman Problem · Elephants Herding Optimization · Combinatorial optimization · Metaheuristic · Nature-inspired

1 Introduction

The Travelling Salesman Problem (TSP) [1] is a classic problem in combinatorial optimization that can be described as follow, given a list of cities and the distances between each pair of cities, the travelling salesman should find the shortest possible route that visits each city by starting and returns to the first city. This problem classified as NP-hard [2] problem who's the computational complexity increases exponentially with the number of cities.

The importance of the TSP appears in many applications in real world such as electronics, telecommunications, transportation, industry, logistics, and astronomy. No exact algorithm exists for solving the TSP. The need to find an efficient solution to this problem, in a reasonable execution time, has led researchers to propose a various approximation algorithms like heuristics such as Local Search [3], Simulated Annealing [4], Tabu Search [5] etc., and metaheuristics such as Genetic Algorithm (GA) [6], Ant Colony Optimization (ACO) [7], Particle Swarm Optimization (PSO) [8], Bee Colony Optimization (BCO) [9], Discrete novel hybrid PSO [10], Discrete Cat Swarm Optimization [11], Discrete Penguins Search Optimization [12], Discrete Swallow Swarm Optimization [13] etc.

The objective of this work is to adapt the EHO algorithm [14], introduced in 2016 by Gai-Ge Wang et al., to solve the TSP.

The rest of this paper is organized as follows: Sect. 2 a presentation of the EHO algorithm introduced to solve continued optimization problems. Section 3 description of the TSP. Section 4 describes the proposed adaptation of EHO to solve the TSP. Section 5 presents in detail the results of numerical experiments on a set of benchmarks from the TSPLIB library [15]. Finally, comes the conclusion in the last section.

2 EHO Algorithm

Gai-Ge Wang et al. introduced the EHO in 2016 [14] to solve continued optimization problems. This method, as indicated by its name, was inspired from the natural herding behavior of elephants, and it was modeled as EHO algorithm, the authors of this algorithm preferred to simplify it into the following idealized rules:

- The elephant population is composed of some clans, and each clan has exactly the same number of elephant.
- At the start of each generation, a number of male elephants will leave their family group.
- In each clan, elephants live together under the leadership of a matriarch. For optimization problem, a matriarch is the fittest elephant in this clan.

2.1 Clan Updating Operator

As mentioned before, the elephants in each clan live under the leadership of a matriarch. Therefore, the next position of each elephant in clan ci is influenced by the position of the matriarch in clan ci. Then, elephant positions are updating as follows:

$$x_{new,ci,j} = x_{ci,j} + \alpha \times \left(x_{best,ci} - x_{ci,j}\right) \times r \tag{1}$$

$x_{new,ci,j}$ is the new position of elephant j in clan ci, $x_{ci,j}$ is the old position of elephant j in clan ci. $x_{best,ci}$ is the matriarch position in clan ci. $\alpha \in [0, 1]$ is a factor that determines the influence of $x_{best,ci}$ on $x_{ci,j}$. $r \in [0, 1]$ is a random number from uniform distribution.

The matriarch position is updated as:

$$x_{new,ci,j} = \beta \times x_{center,ci} \tag{2}$$

$x_{new,ci,j}$ is the new position of the matriarch in clan ci, β is a factor in the range [0,1] that controls the influence of the $x_{center,ci}$ on $x_{new,ci,j}$. $x_{center,ci}$ is the center position in clan ci, and it can be defined as:

$$x_{center,ci} = \frac{1}{n_{ci}} \times \sum\nolimits_{j=1}^{n_{ci}} x_{ci,j} \tag{3}$$

Where n_{ci} is the number of elephants in clan ci.

2.2 Separating Operator

When mature, male elephants will leave their clans. At each generation, the elephant with the worst fitness in clan ci will be replaced via the separating operator as follow:

$$x_{worst,ci} = x_{min} + (x_{max} - x_{min}) \times \text{ rand} \tag{4}$$

x_{max} and x_{min} represent respectively upper position and lower position of the elephant population, $x_{worst,ci}$ is the elephant with the worst fitness in clan ci, rand $\in$ [0,1] is a random number from uniform distribution in the range [0, 1].

2.3 Elephant Herding Optimization Algorithm

```
Begin
  Step 1: Initialization. Set the generation counter t=1;
  initialize the population, set the maximum generation
  MaxGen, the scale factor $\alpha$ and $\beta$, the number of clan
  nClan and the number of elephants for each clan $n_{ci}$.
  Step 2: Fitness evaluation. Evaluate each elephant in-
  dividual according to its position.
  Step 3: while t < MaxGen do
  Sort all the elephant according to their fitness.
  Implement clan updating operator:
    for ci=1 to nClan do
      for j=1 to $n_{ci}$ do
        Update $x_{ci,j}$ and generate $x_{new,ci,j}$ by equation(1).
        if $x_{ci,j}=x_{best,ci}$ then
          Update $x_{ci,j}$ and generate $x_{new,ci,j}$ by equation(2).
        end if
      end for
    end for
  Implement separating operator:
  for ci=1 to nClan do
    Replace the elephant with the worst fitness in clan
  ci by equation(4).
  end for
  Evaluate the population according to the newly updated
  positions.
  Update the generation counter, t=t+1.
  Step 4: end while
  Step 5: Return the best solution found.
End.
```

3 The Travelling Salesman Problem

The TSP is defined by the number of cities (N) and the distances between all pairs of cities (dij for i, j ∈ [1, N]). In TSP, the challenge is to find the shortest possible tour that the salesman can follow to visits each city exactly once and return to the origin city.

The TSP was modelled as a Hamiltonian graph where vertices represent cities, and edges correspond to the distances between cities. A tour is now a Hamiltonian cycle (tour that passes through all the vertices), and optimal tour is the Hamiltonian cycle with the minimum cost (the shortest).

4 Use EHO to Solve the TSP

This section describes the proposal method to solve the TSP into two versions, the first one propose an adaptation, that was able to solve the TSP but in important execution time, after that this paper proposes an improved version of the adapted EHO, which had to prove its efficiency to solve the randomly chosen benchmarks instances of the TSPLIB.

4.1 Adapted Discrete EHO to Solve the TSP

The EHO method proposed by Wang et al. [14] was defined to solve continued optimization problems, so it cannot be applied to solve combinatorial optimization problems, given that in continued optimization, a number represent the solution, but in the case of combinatorial optimization, the solution is represented by an order that can be modeled as a vector. By considering that, we should adapt the operations and operators by respecting the real behavior of elephants:

- The position of an elephant represents a solution (Hamiltonian cycle).
- Subtraction between each two position (x1 – x2) present, the set permutation apply to x2 to obtain x1.
- Addition between the set of permutations and a position is the inversion of subtraction. The addition (x + sp) applies the set of permutations to the x position.

The initialization of solutions has a great impact on the functioning of the algorithm, in our case, we preferred to generate solutions (elephant's positions) in a random way.

After the initialization is completed, we perform the evaluation of the generated solutions according to their fitness, and at the end of this step, we determine the matriarch position (solution with the best fitness) for each clan.

Then, the clan-updating operator is implemented with a small modification in the update position of the matriarch. In the case of the TSP, we cannot apply Eq. (3) to calculate the position of $x_{center,ci}$, to solve this problem, we calculate the average fitness in the clan ci ($f_{moy,ci}$) as shown in Eq. (5), and we take the solution who has the closest fitness to this value as $x_{center,ci}$.

$$f_{moy,ci} = \frac{1}{n_{ci}} \times \sum\nolimits_{j=1}^{n_{ci}} f(x_{ci,j}) \qquad (5)$$

Finally, we implement the separation operator to replace the solution with the worst fitness in each clan using Eq. (4), and we increment the generation counter.

4.2 Improved Discrete EHO to Solve TSP

Since the results obtained after the execution of the adapted EHO algorithm, are not good enough as shown in Table 1, we decided to hybridize this algorithm with another heuristic such 2-opt [16] to obtain better results.

At the first generation, we determine the matriarch for each clan by taking the elephant with the best position in the clan, and as mentioned before, the elephant's positions are generated in a random way, so the probability to have good matriarch's positions in this generation is too low. For this, we implement 2-opt in a first place to improve the matriarch position of each clan.

Then, in each generation, we apply 2-opt after the implementation of the clan updating operator. The following flowchart describes the improved adapted EHO algorithm (Fig. 1):

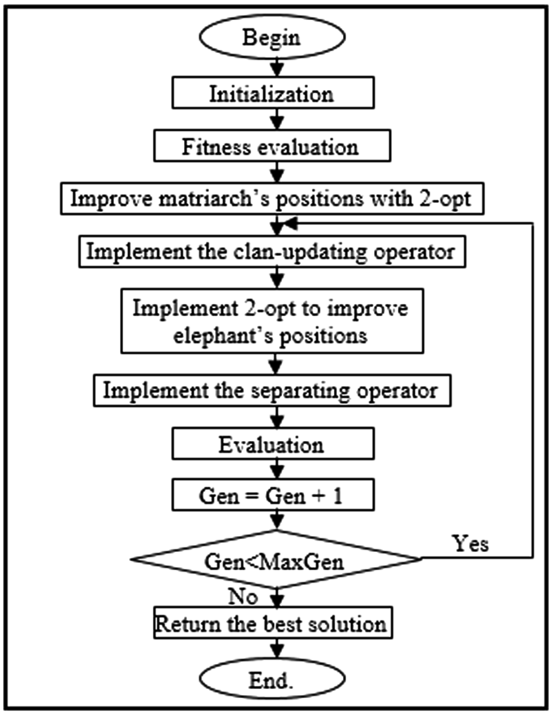
Begin
Initialization
Fitness evaluation
Improve matriarch's positions with 2-opt
Implement the clan-updating operator
Implement 2-opt to improve elephant's positions
Implement the separating operator
Evaluation
Gen = Gen + 1
Gen<MaxGen
Yes
No
Return the best solution
End.

Fig. 1. Flowchart of the improved adapted EHO algorithm.

5 Results and Comparison

The implementation of the adapted Elephants Herding Optimization algorithm and the improved version to solve the TSP were realized on C programming language, and simulations were performed on a personal computer equipped with CORE i5-3210M CPU at 2.50 GHz, 4 Gb RAM, and Windows 7 (64 bits) Professional operating system. Tables 1 and 2 shows the results of executions of these two algorithms on fifteen different benchmark instance of TSPLIB. For both algorithms, the parameters have been fixed as follows: the number of clan nClan was set to 5, the number of elephants in each clan was set to 1000, α was set to 0.9 and β was set to 0.3.

Each table display the following information:

- Inst: name of benchmark instance in TSPLIB library.
- Nb.node: number of nodes.
- Opt: the best solution known for the instance.
- BestR: the best solution found by the algorithm after ten different runs.
- WorstR: the worst solution found by the algorithm after ten different runs.
- Av: the average of ten different runs of the algorithm.
- Time: shows the average time in seconds of ten different runs of the algorithm.
- Err: the percentage of error is calculated by $\{\frac{Av-Opt}{Opt} \times 100\}$.

Table 1. Results obtained by applying Discrete EHO on TSP.

Inst	Nb_node	Opt	BestR	WorstR	Av	Time	Err
Eil51	51	426	630	760	704.8	6.430	65.44
Berlin52	52	7542	11150	12707	12160.7	5.148	61.23
St70	70	675	1286	1646	1476	14.820	118.66
Pr76	76	108159	222020	241240	230424.6	19.094	113.04
Eil76	76	538	1032	1142	1091.3	22.074	102.84
Kroa100	100	21282	60823	69348	66744.5	101.010	213.61
Krob100	100	22141	54181	71827	66078.5	85.199	198.44
Kroc100	100	20749	60773	75063	67603.2	85.815	225.81
Krod100	100	21294	62331	71031	65650.3	75.878	208.30
Kroe100	100	22068	59785	83388	69416.3	80.699	214.55
Eil101	101	629	1400	1588	1514.1	74.209	140.69
Lin105	105	14379	46095	50404	48498.8	111.494	237.28
Pr107	107	44303	177953	216406	201403.4	139.495	354.60
Pr124	124	59030	228204	280386	253235	193.206	328.99
Bier127	127	118282	242103	273829	260442.2	157.139	120.18
Ch130	130	6110	15558	21007	19522.9	235.779	219.52
Pr136	136	96772	342107	396634	362987.2	311.517	275.09
Ch150	150	6528	22027	25044	24233.6	375.680	271.22
Kroa150	150	26524	97372	112590	108611.5	466.316	309.48
Krob150	150	26130	107750	122903	109367	412.023	318.54

Table 1 presents the results obtained after ten different runs of the adapted EHO algorithm on fifteen instances of TSPLIB. This algorithm gives acceptable results but not good enough, and the execution time is quite long (Fig. 2).

Table 2. Results obtained by applying the Improved Discrete EHO on TSP.

Inst	Nb_node	Opt	BestR	WorstR	Av	Time	Err
Eil51	51	426	426	426	426	6.519	0.00
Berlin52	52	7542	7542	7542	7542	0.255	0.00
St70	70	675	675	675	675	4.792	0.00
Pr76	76	108159	108159	108159	108159	9.203	0.00
Eil76	76	538	538	539	538.5	189.903	0.09
Kroa100	100	21282	21282	21282	21282	8.634	0.00
Krob100	100	22141	22141	22141	22141	33.069	0.00
Kroc100	100	20749	20749	20749	20749	6.912	0.00
Krod100	100	21294	21294	21294	21294	18.074	0.00
Kroe100	100	22068	22073	22121	22100	58.125	0.14
Eil101	101	629	630	634	632.6	517.912	0.57
Lin105	105	14379	14379	14379	14379	9.714	0.00
Pr107	107	44303	44303	44303	44303	8.876	0.00
Pr124	124	59030	59030	59030	59030	4.165	0.00
Bier127	127	118282	118282	118392	118321.6	495.519	0.03
Ch130	130	6110	6110	6113	6112.25	658.553	0.03
Pr136	136	96772	96772	96781	96775	175.447	0.00
Ch150	150	6528	6550	6575	6564.2	1867.273	0.55
Kroa150	150	26524	26524	26524	26524	721.539	0.00
Krob150	150	26130	26132	26132	26132	163.478	0.00

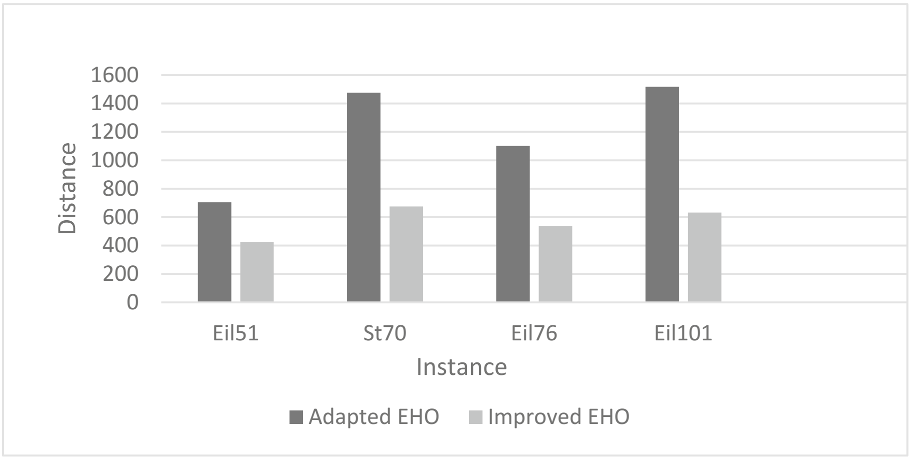

Fig. 2. Comparison between the results obtained after ten different runs of the adapted EHO algorithm and the improved adapted EHO algorithm on these instances: Eil51, St70, Eil76 and Eil101.

Table 2 presents the results obtained after ten different runs of the improved adapted EHO algorithm on fifteen instances of TSPLIB. This algorithm gives best results in a short execution time.

6 Conclusion

In this paper, we have first presented an adaptation of EHO algorithm via Gai-Ge Wang, without modification or hybridization, to solve the symmetric TSP, this adaptation did not give good results compared to the results obtained for the continuous problems. Secondly, we have improved the adapted EHO algorithm by hybridization with 2-opt to obtain best solutions. The simulation results were compared to the best solution, the worst solution and the average after ten different runs on each instance. The results of the comparison have proved the performance of the improved adapted EHO algorithm. In the future, we will aim to extend the proposed algorithm to be applied on various real applications area based on TSP.

References

1. Flood, M.M.: The travelling-salesman problem. Oper. Res. **4**(1), 61–75 (1956)
2. Hochbaum, D.S.: Approximation Algorithms for NP-Hard Problems. PWS Publishing Co., Boston (1996)
3. Korupolu, M.R., Plaxton, C.G., Rajaraman, R.: Analysis of a local search heuristic for facility location problems. J. Algorithms **37**(1), 146–188 (2000)
4. Aarts, E., Korst, J., Michiels, W.: Simulated annealing. In: Search Methodologies, p. 187–210. Springer, Boston (2005)
5. Gendreau, M., Hertz, A., Laporte, G.: A tabu search heuristic for the vehicle routing problem. Manag. Sci. **40**(10), 1276–1290 (1994)
6. Fonseca, C.M., Fleming, Peter, J., et al.: Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization. In: ICGA, pp. 416–423 (1993)
7. Dorigo, M., Birattari, M.: Ant colony optimization. In: Encyclopedia of Machine Learning, pp. 36–39. Springer, Boston (2011)
8. Poli, R., Kennedy, J., Blackwell, T.: Particle swarm optimization. Swarm Intell. **1**(1), 33–57 (2007)
9. Wong, L.-P., Low, M.Y.H., Chong, C.S.: A bee colony optimization algorithm for travelling salesman problem. In: Second Asia International Conference on Modeling & Simulation, AICMS 2008, pp. 818–823. IEEE (2008)
10. Bouzidi, M., Riffi, M.E.: Discrete novel hybrid particle swarm optimization to solve travelling salesman problem. In: 5th Workshop on Codes, Cryptography and Communication Systems (WCCCS), pp. 17–20. IEEE (2014)
11. Bouzidi, A., Riffi, M.E.: Discrete cat swarm optimization to resolve the travelling salesman problem. Int. J. Adv. Res. Comput. Sci. Softw. Eng. **3**(9), 13–18 (2013)
12. Mzili, I., Riffi, M.E.: Discrete penguins search optimization algorithm to solve the travelling salesman problem. J. Theor. Appl. Inf. Technol. **72**(3), 331–336 (2015)
13. Bouzidi, S., Riffi, M.E.: Discrete swallow swarm optimization algorithm for travelling salesman problem. In: Proceedings of the 2017 International Conference on Smart Digital Environment, pp. 80–84. ACM (2017)

14. Wang, G.-G., Deb, S., Gao, X.-Z., et al.: A new metaheuristic optimisation algorithm motivated by elephant herding behaviour. Int. J. Bio-Inspired Comput. **8**(6), 394–409 (2016)
15. Reinelt, G.: TSPLIB—a travelling salesman problem library. ORSA J. Comput. **3**(4), 376–384 (1991)
16. Mcgovern, S.M., Gupta, S.M.: 2-opt heuristic for the disassembly line balancing problem. In: Environmentally Conscious Manufacturing III, pp. 71–85. International Society for Optics and Photonics (2004)