SOCIETY FOR SCIENCE AND EDUCATION
UNITED KINGDOM

TMLAI TRANSACTIONS ON
MACHINE LEARNING AND ARTIFICIAL INTELLIGENCE

VOLUME 5 NO 4, 2017
ISSN 2054-7390

Comparative Analysis of Three Metaheuristics For Solving The Travelling Salesman Problem

Safaa Bouzidi, Moahmmed Essaid Riffi, Abdelhamid Bouzidi
Dept. of Computer Science, Faculty of Science, ChouaibDouakkali University
El Jadida, Morocco
sfbouzidi@gmail.com, said@riffi.fr, mr.abdelhamid.bouzidi@gmail.com

This research paper aims to do a comparative study of three recent optimization metaheuristic approaches that had been applied to solve the NP-hard optimization problem called the travelling salesman problem. The three recent metaheuristics in study are cuckoo search algorithm, cat swarm optimization algorithm and bat-inspired algorithm. To compare the performances of these methods, the three metaheuristics are applied to solve some benchmark instances of TSPLIB. The obtained results are collected and the error percentage is calculated. The discussion, will present which method is more efficient to solve the real application based on the travelling salesman problem.

Keywords- Metaheuristic; Cat Swarm Optimization; Bat-inspired algorithm; Cuckoo Search; Travelling salesman problem; NP-hard

1 Introduction

The Travelling salesman problem (TSP) is one of the best known in the operations research, studied since the 19th century, and was introduced at first as a game by Wiliam Rowan Hamilton and Thomas Penyngton Kirkman [1]. This problem aim is to find the minimal possible path, by starting from a city, and visiting atonce each city in the problem, and return to its city of start. In general, we try to minimize the total travel time and total distance. In addition, this problem is NP-hard problem [2].

The importance to the resolution of traveling salesman problem appears in various complex application area in our real life, such as:

- Drilling of printed circuit boards
- Overhauling gas turbine engines
- Computer wiring
- Vehicle routing

In front of its importance, and the difficulty to solve it. The researchers had try to solve it by numerous methods. One of them is the computational intelligence based on nature observation and study, called the metaheuristics. There is also the exact methods, but it can only solve the TSP instances problem with minimal size, but when the number of city is high, its impossible to find the global solution. But after using the metaheuristic method, it's possible to find the best optimal solution in a reasonable execution time, as the simulated annealing [3], Tabu search [4], Harmony search [5], cuckoo search algorithm [6], Genetic algorithm [7], Ant colony optimization [8], Particle swarm optimization [9], Bee colony optimization [10], cat swarm optimization algorithm [11], bat swarm optimization [12]. To improve the efficiency of some

DOI: 10.14738/tmlai.54.3211
Publication Date: 15th August, 2017
URL: http://dx.doi.org/10.14738/tmlai.54.3211

methods, the researchers had proposed some hybrid methods that had proven their efficiency to solve the TSP, such as, the hybrid genetic algorithm [13], hybrid ant-colony optimization [14], hybrid particle swarm optimization [15], hybrid bee colony optimization [16]. And more. However, the problem is to know which one is more efficient so it can be applied for solving a real TSP problem based.

The aims of this paper is a comparison study, between three recent algorithms to solve the TSP, and know which one is efficient to be applied in others, in order to apply them to a real application based on the travelling salesman problem. The rest of this research paper is organized as follows: section II presents the formulation of TSP. Section III, is a brief description of the three metaheuristics in study. Section IV shows the result of this study, and a discussion. Finally, the conclusion.

2 Traveling Salesman Problem

Given a graph G = (X, E), where X = {1,...,n} is a set of nodes, each node presents a city, and E = {1,...,m} presents a set of edges, each edge presents a path between two selected cities. and costs, dij , presented the arcs distance between vertices i and j, the TSP consists in finding the Hamiltonian cycle of G with the best minimal total length D.

$$D=\sum_{i=1}^{n}\sum_{i=1}^{n} d_{ij} \qquad (1)$$

3 Metaheuristics Description

This section is dedicated to describe the three metaheuristic that have been recently improved to be able to solve the NP-combinatorial optimization problem, in specal the TSP.

These methods are :

- The cat swarm optimization algorithm.
- The bat swarm optimization algorithm.
- The Cukoo search algorithm.

3.1 Cat swarm optimization

The metaheuristic Cat swarm optimization (CSO) is an evolutionary method proposed in 2006 by Chu and Tsai [17]. To describe the natural behaviors of cats. The real cat spends the majority of his life, about Seventeen five percent; at rest, attentive and vigilant in observing its surroundings for his next move. Moreover, twenty percent of its life in chase, by moving quickly to chase a pray or any moving object.

In the CSO algorithm, each cat can be considered as a candidate solution. There are two mode; the seeking mode (SM) and the tracing mode (TM). The seeking mode presents the real cat at rest. The tracing mode presents the cat when it traces its path, according to its own velocity to chasing a prey or any moving object. To combine these two modes, the mixture ratio (MR) is defined.

The characteristic of each cat are:

- The position presents the solution.
- the velocity to update the position.
- the flag defines the mode that characterizes a cat.

The description of the total algorithm process of CSO is:

```
BEGIN
Create N cats;
    Initialize the position, velocity, and flag of every cat;
    Repeat
    Evaluate the position of each cat;
    keep the position of the cat with the best fitness;
    value into Gbest parameter;
    For(each cat_k in the swarm)
        If (flag of cat_k is SM)
          Begin
            Make j copies of cat_k;
            If (SPC==true)
              Begin
            includes cat_k as a candidate;
                j=SMP-1;
              End
            ELSE
                j= SMP;
            EndIf
            For(each copy)
                Select a number of dimensions based on CDC;
            Update their value using SRD percent of their
            current value;
     Evaluate the fitness of each copy;
        For( each cat)
            $P_k = \frac{|FS_k - FS_{min}|}{FS_{max} - FS_{min}}$
            Randomly select a new position for cat_k;
        EndFor
      END
    ELSE
          Begin
            $V_i = V_i + r * c * (X_{best} - X_i)$   (2)
            $X_i = X_i + V_i$(3)
          End
    EndIf
      Reinitialize the flag of each cat;
     Until (Condition is satisfied)
    END
```

Where, in equation (2):

- **X** $_i$The position of the selected cat$_i$.
- **X**$_{best}$: is the best solution/position of the cat who has the best fitness value.
- **V** $_i$: The old speed value (current value).
- **c**: is a constant.
- **r**: a random value in the range [0, 1]

Moreover, in equation (3):

- **X** $_k$: The position of selected cat $_i$
- **V** $_k$: The velocity of cat $_i$

The adaptation of operation used in CSO algorithm:

- Movement: the movement of cat $_k$ is a swap applied to solution of this cat.
- Addition:
- The addition between position **X** and a velocity **V**, is applying the swap in velocity **V** to position **X**, the result is a new position **x'**.
- The addition between two velocities **v** and **v'**, is a new velocity containing all the couple of swaps of **v** and **v'**.
- Subtraction: the result of the subtraction between two positions **x** and **x'** is a velocity **v**, it is the opposite of addition:

$$\mathbf{x} + \mathbf{v} = \mathbf{x'} \Leftrightarrow \mathbf{x'} - \mathbf{x} = \mathbf{v}$$

- Multiplication: the multiplication is performed between a float value and velocity, the result is a velocity. The different possible cases according to the real **k** are:

- If k = 0: k * v = 0
- If (k>0 & k<=1) :r * v = (i_k,j_k)[k : 0 → (c*|v|)]
- If k>1: then we separate. Decimal and integer part, k = n +x. Where n is the integer part of r, and x corresponds to the decimal parts. We will then return each party to the previous cases.
- If k<0: k * v= (-k)*¬v. Now (-k) >0, and you will consider one of the previous cases.

This methode was improved in 2012, by A.bouzidi and M.E. RIFFI [11], to solve the travelling salesman problem.

3.2 Bat-inspired algorithm

Bat algorithm (BA) is a bio-inspired algorithm and stochastic optimization technique developed by Yang in 2010 [18]. This metaheuristic is inspired behavior of bats. In nature, bats emit ultrasound pulses to the surrounding environment with hunting and navigation purposes. After the emission of these pulses, bats hear the echoes, and based on them they can locate themselves and identify obstacles and preys. Furthermore, each bat of the swarm is able to find the most nutritious areas performing an individual search, or moving towards a nutritious location previously found by the swarm.

The basic Bat algorithm developed by Xin-She Yang [18] is described in following steps:

```
BEGIN
    Define the objective function f(x);
    Initialize the bat population X = (x1,x2,...,xn);
    for each bat xi in the population do
      Initialize the pulse rate ri, velocity vi and loudness Ai;
      Define the pulse frequency fi at xi;
    End
    Repeat
    for each bat xi in the population do
       Generate new solutions through Equations 4 and 5;
    if rand>rithen
         Select one solution among the best ones;
         Generate a local solution around the best one;
    End
    if rand<Ai and f(xi)<f(x∗) then
         Accept the new solution;
```

```
            Increase ri and reduce Ai;
        end
        end
        until termination criterion not reached
        Rank the bats and return the current best bat of the
        population;
    END
```

Y.Saji and ME RIFFI [12] have improved in 2015 the bat-algorithm; by solving the NP-hard combinatorial optimization. That had proven its efficiency to solve the travelling salesman problem.

For each virtual bats, the movement is given by updating their velocities v_i^{t+1} and positions x_i^{t+1} at time step t+1 using (4) and (5), as follows:

$$v_i^{t+1} = \chi(x_i^t, x_*, f_i) \tag{4}$$

$$x_i^{t+1} = \varphi(x_i^t, v_i^{t+1}) \tag{5}$$

The function χ is a crossover function that has three input arguments and returns a set of permutations reached by applying 2-exchange crossover mechanism described in [12].

The function ϕ is a function to sort x_i^twhile taking into account the set of permutations of v_i^{t+1}.

3.3 Cuckoo search algorithme

The Cuckoo Search (CS) algorithm is a metaheuristic introduced first in 2009 by Xin-She Yang and Suash Deb [19], bio inspired by the breeding behavior of cuckoos. It was inspired by the obligate brood parasitism of some cuckoo species by laying their eggs in the nests of other host birds (of other species). Some host birds can engage direct conflict with the intruding cuckoos. Some cuckoo species such as the New World brood-parasitic Tapera have evolved in such a way that female parasitic cuckoos are often very specialized in the mimicry in colors and pattern of the eggs of a few chosen host species.

In the CS method, each egg in a nest represents a solution, and a cuckoo egg represents a new solution. The aim is to use the new and potentially better solutions (cuckoos) to replace a not-so-good solution in the nests. In the simplest form, each nest has one egg. The algorithm can be extended to more complicated cases in which each nest has multiple eggs representing a set of solutions.

The process of the CS is as follow:

```
    BEGIN
      Objective function f(x),x=(x_1,...,x_d)^T
      Generate initial population of n host nests x_i (i = 1,2,..., n)
    while (t <MaxGeneration) or (stop criterion)
    Get a cuckoo randomly by Levy flight;
    Evaluate its quality/fitness Fi;
       Choose a nest among n (say, j) randomly;
    if (Fi > Fj)
    replace j by the new solution;
    End if
       A fraction (p_a) of worse nestsare abandoned and new
    ones are built;
       Keep the best solutions(or nests with quality solutions);
       Rank the solutions and find the current best;
    Endwhile
      Postprocess results and visualization;
END
```

A cuckoo i generates new solution$x_i^{(t+1)}$by a Levy flight, according to (6) :

$$x_i^{(t+1)} = x_i^{(t)} + \alpha \oplus Levy(s,\lambda) \tag{6}$$

Where$\alpha$is the step size that follows the Levy distribution in (7):

$$Levy(s,\lambda) \sim s^{-\lambda} \tag{7}$$

Which has an infinite variance with an infinite mean [19]. Here,s is step size drawn from a Levy distribution.

Some rules about CS should be respected, are:

(1) Each cuckoo lays one egg at a time and selects a nest randomly.
(2) The best nest with the highest quality of egg can pass onto the new generations.
(3) The number of host nests is fixed, and the egg laid by a cuckoo can be discovered by the host bird with a probability pa∈[0 , 1] .

The improved Cuckoo search algorithm had succeeded to solve the travelling salesman problem by A. Ouaarab and al. [6].

4 Metaheuristique to Solve Tsp

This part is devoted to the collect and analysis of the results after applying the three metaheuristics to solve fourteen-benchmark instance of TSPLIB [20]. The collected result will show which method is more efficient to solve the real application based TSP.

The following table shows the collected result of each instances benchmark taken from TSPLIB, the size of the instance (number of edge in the graph), the best-known optimal global value, and for each method, the best, and the worst result got in ten iteration. In the last, the relative percentage error (RPD) calculated in ten iteration as follow:

Table 9 : results by the application to solve some banchmark instances of TSP

Instance	size	Optimum	CSO algorithm			Bat algorithm			Cuckoo-search		
			BestR	WorstR	RPD(%)	BestR	WorstR	RPD(%)	BestR	WorstR	RPD(%)
eil51	51	426	426	427	0,07	426	426	0.00	426	426	0.00
berlin52	52	7542	7542	7693	1.00	7542	7542	0.00	7542	7542	0.00
st70	70	675	675	682	0.51	675	675	0.00	675	675	0.00
eil76	76	538	538	549	1,02	538	542	0.14	538	539	0.17
kroA100	100	21282	21282	21717	1,0220	21282	21282	0.00	21282	21282	0.00
kroB100	100	22141	22141	23014	1,9715	22141	22141	0.00	22141	22157	0.00
kroC100	100	20749	20749	21669	2,2170	20749	20880	0.02	20749	20749	0.00
kroD100	100	21294	21294	22878	3,7195	21294	21374	0.04	21309	21389	0.04
eil101	101	629	629	636	0,56	629	637	0.54	630	633	0.22
bier127	127	118282	118282	122128	1.62	118282	118693	0.08	118282	118730	0.06
ch130	130	6110	6110	6394	2.32	6110	6155	0.23	6141	6174	0.42
ch150	150	6528	6528	6782	2.07	6528	6584	0.34	6528	6611	0.33
gil262	262	2378	2378	2670	6.14	2380	2410	0.08	2382	2418	0.68
A280	280	2579	2579	2783	3.96	2579	2611	0.30	2579	2592	0.51

To assess the collected results, the content of the results in the table 1 is translated into the following graph.

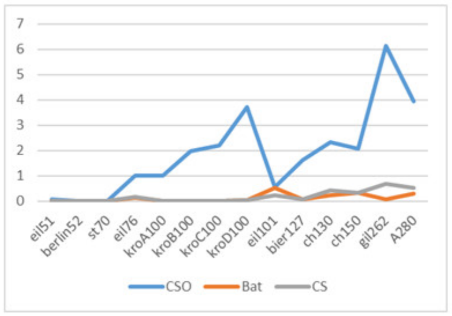

Figure 1 : RPD CSO, BA, CS to solve TSP

By analyzing the variation of the error percentage, after the application to some benchmark instance of TSPLIB, it seems clear that the BA and CS and algorithm is the best one to solve the TSP then the CSO, because the error percentage, as appears in the graphs, is lower than the CSO. Which means that the BA and CS algorithms are more efficient to solve the real life application based the TSP.

5 Conclusion

This research paper presented the application of some metaheuristics to solve the travelling salesman problem, to choose the best metaheuristic to solve some real applications based TSP problems. The studied methods are the cat swarm optimization algorithm, Bat Algorithm, Cuckoo search algorithm. This research paper aims to compare the relative percentage error by the application of these recent metaheuristics to some benchmark instances. The computational results show that the bat algorithm, and cuckoo search are more efficient than other cat swarm optimization, and had almost a similar error percentage.

In future researches, we will try to provide an application of a more performant method to solve a real application area based TSP such as the scheduling problem, Frequency Assignment Problem, applications in medical image processing.

REFERENCES

[1] N. Biggs, E. K. Lloyd and R. J. Wilson, Graph Theory, 1736-1936, Oxford University Press, 1976.

[2] M. Garey and D. S. Johnson, Computers and intractability : a Guide to the Theory of NP-Completeness Freeman San Francisco, San Francisco: Freeman, 1979.

[3] J. R. Allwright and D. Carpenter, "A distributed implementation of simulated annealing for the travelling salesman problem," Parallel Computing, vol. 10, no. 3, pp. 335-338, 1989.

[4] M. Gendreau, G. Laporte and F. Semet, "A tabu search heuristic for the undirected selective travelling salesman problem," European Journal of Operational Research, vol. 106, no. 2, pp. 539-545, 1998.

[5] M. Bouzidi and M. E. Riffi, "ADAPTATION OF THE HARMONY SEARCH ALGORITHM TO SOLVE THE TRAVELLING SALESMAN PROBLEM," Journal of Theoretical & Applied Information Technology, vol. 62, no. 1, 2014.

[6] A. Ouaarab, B. Ahiod and X.-S. Yang, "Discrete cuckoo search algorithm for the travelling salesman problem," Neural Computing and Applications, vol. 24, no. 7-8, pp. 1659-1669, 2014.

[7] Y. Nagata and D. Soler, "A new genetic algorithm for the asymmetric traveling salesman problem," Expert Systems with Applications, vol. 39, no. 10, pp. 8947-8953, 2012.

[8] K. Jun-man and Z. Yi, "Application of an improved ant colony optimization on generalized traveling salesman problem," Energy Procedia, vol. 17, pp. 319-325, 2012.

[9] X. H. Shi, Y. C. Liang, H. P. Lee, C. Lu and Q. Wang, "Particle swarm optimization-based algorithms for TSP and generalized TSP," Information Processing Letters, vol. 103, no. 5, pp. 169-176, 2007.

[10] L.-P. Wong, M. Y. H. Low and C. S. Chong, "A bee colony optimization algorithm for traveling salesman problem," in Second Asia International Conference on Modelling & Simulation, IEEE, 2008, pp. 818-823.

[11] A. Bouzidi and M. E. Riffi, "Discrete Cat Swarm Optimization to Resolve the Traveling Salesman Problem," Interneational Journal of Advanced Research in Computer Science and Software Engineering, vol. 3, no. 9, pp. 13--18, 2013.

[12] Y. Saji and M. E. Riffi, "A novel discrete bat algorithm for solving the travelling salesman problem," Neural Computing and Applications, pp. 1-14, 2015.

[13] K. Katayama, H. Sakamoto and H. Narihisa, "The efficiency of hybrid mutation genetic algorithm for the travelling salesman problem," Mathematical and Computer Modelling, vol. 31, no. 10, pp. 197-203, 2000.

[14] H. Duan and X. Yu, "Hybrid ant colony optimization using memetic algorithm for traveling salesman problem," in Approximate Dynamic Programming and Reinforcement Learning, 2007. ADPRL 2007. IEEE International Symposium on, IEEE, 2007, pp. 92-95.

[15] Y. Marinakis and M. Marinaki, "A hybrid multi-swarm particle swarm optimization algorithm for the probabilistic traveling salesman problem," Computers & Operations Research, vol. 37, no. 3, pp. 432--442, 2010.

[16] L.-P. Wong, M. Y. H. Low and C. S. Chong, "Bee colony optimization with local search for traveling salesman problem," International Journal on Artificial Intelligence Tools, vol. 19, no. 3, pp. 305-334, 2010.

[17] S.-C. Chu, P.-W. Tsai and J.-S. Pan, "Cat swarm optimization," in PRICAI 2006: Trends in artificial intelligence, Springer, 2006, pp. 854--858.

[18] X.-S. Yang, "A new metaheuristic bat-inspired algorithm," in Nature inspired cooperative strategies for optimization (NICSO 2010), Springer, 2010, pp. 65-74.

[19] X.-S. Yang and S. Deb, "Cuckoo search via Lévy flights," in Nature & Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on, IEEE, 2009, pp. 210-214.

[20] G. Reinelt, "TSPLIB—A traveling salesman problem library," ORSA journal on computing, vol. 3, no. 4, pp. 376-384, 1991.