Discrete bat-inspired algorithm for travelling salesman problem

Yassine Saji[1] *, Mohammed Essaid Riffi [2]
Computer Science Department, LAROSERI
Faculty of Science, Chouaïb Doukkali University,
El Jadida, Morocco
[1]yassine.saji@gmail.com, [2]said@riffi.fr

Belaïd Ahiod [3]
LRIT, Associated Unit to CNRST (URAC 29),
Faculty of Science, Mohammed V- Agdal University,
BP 1014, Rabat, Morocco
[3]ahiod@fsr.ac.ma

Abstract—**Bat algorithm (BA) is a new nature-inspired metaheuristic optimization algorithm based on the echolocation behavior of bats to find their prey and to avoid obstacles in the darkness. This new algorithm has showed a higher efficiency in solving continuous optimization problems. In this study, we have proposed a novel adaptation of BA for solving travelling salesman problem (TSP), which is known as an NP-hard combinatorial optimization problem. We have also redefined some operators used in basic BA. Implementation is carried out in MATLAB on examples of symmetric instance. The results are optimistic and clearly demonstrate the efficiency of the proposed algorithm in terms of convergence towards optimal solution.**

Keywords- Travelling Salesman Problem; Meta-heuristic; NP-hard Problem; Combinatorial Optimization; Bat Algorithm

I. INTRODUCTION

The Travelling Salesman Problem (TSP) is one of the well-known and extensively studied problems in discrete or classical combinatorial optimization; it was introduced in 1800s by the Irish mathematician W.R. Hamilton and the British mathematician Thomas Kirkman. The goal of this problem is to find the shortest tour that visits each city in a given list exactly once and then returns to the starting city. TSP is an NP-hard problem [7] and it is easy to describe but very difficult to solve. Many studies have been devoted to solve TSP such as Tabu Search (TS) [16], Simulated Annealing (SA)[1], Genetic Algorithms (GA)[11] [12], and Greedy Randomized Adaptive Search Procedure (GRASP) [9]. Recently, in last fifteen years, several studies based on nature-inspired algorithms or Swarm Intelligence Methods, such as Ant Colony Optimization (ACO)[3][4], Particle Swarm Optimization (PSO) [6] [8] and Cuckoo Search (CS)[10] have been proposed for the solution of the TSP.

The traveling salesman problem can be stated as a permutation problem with the objective of finding a shortest closed tour which visits all the cities in a given set; it can be modeled as completely connected graph in a d-dimensional Euclidean space, which cities are the graph's vertices, paths are the graph's edges, and a path's distance is the edge's length. Mathematically, it can be defined as given a set of n cities, named $\{c_1, c_2,\ldots,c_n\}$, and permutations $\pi_1, \ldots, \pi_n$, the goal is to find a number of permutation $\pi_i=\{c_1,c_2,\ldots,c_n\}$, such that minimizes $f(\pi)$ the sum of all Euclidean distances d between each city from the same path π and it given as follows :

$$f(\pi)=\sum_{i=1}^{n-1} d(c_i,c_{i+1})+d(c_n,c_1)$$

The Euclidean distance d, between any two cities with coordinate (x_1, y_1) and (x_2, y_2) is calculated by:

$$d=\sqrt{(x_1-x_2)^2+(y_1-y_2)^2}$$

In this study, the TSP problem is described as a symmetric problem where all of the edge costs are symmetric. This means that, for all vertices in the graph, the cost incurred, when moving from vertex c_i to vertex c_{i+1}, is the same as the cost incurred when moving from vertex c_{i+1} to vertex c_i ($d(c_i, c_{i+1}) = d(c_{i+1}, c_i)$).

Main aim of this paper is to introduce a new Bat Algorithm (BA) for solving Travelling Salesman Problem (TSP). The rest of this paper is organized as follows: we introduce by a brief literature review of classic Bat Algorithm. In the next part we will redefine the basic concept used in BA. Finally we conclude and give some perspectives of research.

II. THE BASIC BAT ALGORITHM

Recently, a new metaheuristic search algorithm is presented by Yang in 2010 [14], namely the bat-inspired algorithm or bat algorithm (BA) [14] [15]. The bat-inspired search is based on the echolocation behavior of bats to find the prey and discriminate different types of insects even in complete darkness with varying pulse rates of emission and loudness. They achieve this by emitting calls out to the environment and listening to the echoes that bounce back from them. They are able to identify location of other objects and to measure the distance from the targets by following delay of the returning sound. These emitting calls are a very loud sound pulses vary in properties according to their hunting strategies and depending on the species. The echolocation pulses are characterized by three features; pulse frequency, pulse emission rate and loudness or intensity. Yang [14] idealized echolocation behavior of bats and its associated parameters in a numerical optimization algorithm. The BA algorithm has been empirically tested and compared with other existing algorithms using some single and multi-objective standard functions of unconstrained optimization in Yang [14] [15]. Furthermore, Yang and Gandomi [13], and Gandomi and al. [5] were validating the performance of BA in benchmark problems of constrained engineering optimization. These studies have clearly demonstrated a better efficiency of the bat-inspired algorithm over other methods.

The basic bat algorithm developed by Xin-She Yang [14] is described in following steps:

1. All bats use echolocation to sense distance, and they also `know' the difference between food/prey and background barriers in some magical way;

2. Bats fly randomly with velocity v_i at position x_i with a fixed frequency f_{min}, varying wavelength λ and loudness A_0 to search for prey. They can automatically adjust the wavelength (or frequency) of their emitted pulses and adjust the rate of pulse emission r in the range of [0, 1], depending on the proximity of their target;

3. Although the loudness can vary in many ways, we assume that the loudness varies from a large (positive) A_0 to a minimum constant value A_{min}.

Firstly, Initializing bat population: position x_i, velocity v_i and frequency f_i. The movement of the virtual bats is given by updating their velocities and positions at time step t using Equations 1, 2 and 3, as follows:

$$f_i = f_{min} + (f_{max} - f_{min})\beta, \quad (1)$$

$$v_i^t = v_i^{t-1} + (x_i^{t-1} - x_*) f_i \quad (2)$$

$$x_i^t = x_i^{t-1} + v_i^t \quad (3)$$

Where $\beta \in [0, 1]$ denotes a randomly generated vector from uniform distribution. The variable x_* denotes the current global best location (solution), which is located after comparing all the solutions provided by the m bats.

After selection of one solution among the current best solutions of bat, a random number is applied; if this random number upper to pulse emission rate r_i, a new solution will be accepted around the current best solutions; it can be represented by:

$$x_{new} = x_{old} + \varepsilon A^t \quad (4)$$

Where $\varepsilon \in [-1, 1]$ is a random number, while $A^t = \langle A_i^t \rangle$ is the average loudness of all the bats at current generation. Furthermore, the loudness A_i and the pulse emission rate r_i will be updated, and a solution will be accepted if a random number is less than loudness A_i and $f(x_i) < f(x_*)$. A_i and r_i are updated by:

$$A_i^{t+1} = \alpha A_i^t, r_i^{t+1} = r_i^0[1 - exp(-\gamma t)] \quad (5)$$

Where α, γ are constants and $f(\cdot)$ is fitness function. The algorithm repeats until maximal number of cycles is reached.

III. THE DISCRETE BAT ALGORITHM FOR TSP

A. Position and Velocity

The main problem to switch from continuous to discrete space is the notion of position and velocity. In the TSP we consider a position as a sequence of n cities and velocity allows passing from a position to another, so it's defined as a set of permutation. Then the changes occur in the equations 2 and 3 and algebraic operators ($\oplus$, +, - and $\otimes$) haven't here their usual meaning and should be redefined. It's similar to the discrete particle swarm optimization (PSO) algorithm for the traveling salesman problem [2]. The new equations are described as follow:

$$v_i^{t+1} = v_i^t \oplus (x_i^t - x_*) \otimes f_i \quad (6)$$

$$x_i^{t+1} = x_i^t + v_i^{t+1} \quad (7)$$

It can be given a concrete example: Suppose there is a TSP problem with five nodes, here is a solution X= (5 2 1 4 3) and velocity V= {(2, 3) (5, 1)}.

The + operator in (7) can apply to sort a set of permutation. Then X+ V = (1 3 5 4 2).

The operator $\oplus$ in (6) is used to concatenate two permutations. Let V_1= {(2, 3) (5, 1)} and V_2= {(4, 3) (5, 4)} then $V_1 \oplus V_2$ = {(2, 3) (5, 1) (4, 3) (5, 4)}.

The – operator in (6) returns a set of permutations that allow passing from a position to another. Let X_1 = (1 2 3 4 5) and X_2 = (3 4 1 2 5), so $X_1 - X_2$ = {(3, 1) (4, 2)}.

The operator $\otimes$ in (6) is a multiplication of permutation by a positive real (k) and we can distinct the following cases: if 0< k <1 we truncate from V a set of N elements with N= $E(k*|V|)$. E is the integer part and |V| is the number of elements. If k is an integer then we repeat the permutation k times. Finally if k is a real upper to 1 so we separate the integer part and the decimal part and we refer to the previous two steps.

B. TSP-Bat Algorithm Description

The specific process of the proposed algorithm is shown as follows:

TSP_BAT_1

Initialize all parameters

Generate a set of starting solution by nearest neighbor method.

For each bat initialize velocities v_i

TSP_BAT_2

If the algorithms are ended, go to TSP_BAT_8.

TSP_BAT_3

For all bats generate new solutions by adjusting frequencies, and updating velocities and locations [equations 1, 6 and 7].

TSP_BAT_4

A random number is applied; if it's upper to r_i then select a solution among the best solutions.

Generate a local solution by exchanging 2 arcs from the selected best solution.

TSP_BAT_5

Generate a new solution by flying randomly.

TSP_BAT_6

Generate a random number, if this number is less than loudness A_i and the last new solution is better than the best solution so accept the new solutions.

TSP_BAT_7

Rank the bats and find the current best.

TSP_BAT_8

Sketch the global best solution.

IV. Results and Finding

To test the validity of the proposed Discrete BA algorithm (DBA), we use some symmetric TSP benchmarks problem from TSPLIB where each instance is run for 10 times. The experiment has been made on a PC with processor Intel(R) Core(TM) 2 Duo CPU T4300@ 2.1GHZ 800MHz and 2.00 GB of RAM.

Table I shows the numerical results of our algorithm. The first column indicates the instance name, the second column indicates the problem size, the third column stands for the best known from TSPLIB, the fourth column indicates the best results obtained by DBA and the last one represents the worst results obtained. Compared to the optimal solutions known in the literatures, it can be seen that our algorithm achieves a good solutions quality when the problem size is ranged from 30 to 76. Additionally, Table II shows the computation results of DBA compared to Max-Min Ant System (MMAS), Ant System (AS), Ant System with elitist strategy (ASe), rank-based version of Ant System (ASrank) and Ant Colony System (ACS), Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Simulated Annealing (SA) and Chaotic Ant Swarm (CAS). The proposed algorithm demonstrates clearly its efficiency in solving some typical instances compared with most competitive methods for TSPs in the literatures.

TABLE I. RESULTS OF THE PROPOSED ALGORITHM FOR SYMMETRIC INSTANCES FROM TSPLIB

Problem	**Problem size**	**Optimal**	**Best**	**worst**
Att48	48	10628	**10628**	10628
Eil51	51	426	**426**	452
St70	70	675	**675**	742
Eil76	76	538	**538**	568
Gr96	96	55209	60263	63087

TABLE II. COMPARATIVE RESULTS OF DBA AND OTHER ALGORITHMS

Problem	**Eil51**	**St70**	**Eil76**	**kroA100**	**d198**
Optimal	**426**	**675**	**538**	**21282**	**15780**
DBA	**426.0**	**675**	**538**	21310.0	15888.0
MMAS [17]	427.6	-	-	21320.3	15972.5
ACS [18]	428.1	-	-	21420	16054
AS_{rank} [17]	434.5	-	-	21746	16199.1
AS [17]	437.3	-	-	21522.8	16205
AS_e [17]	428.3	-	-	22471.4	16702.1
GA [19]	429	-	-	21445	-
PSO [20]	436.9	697.5	560.4	-	-
SA [21]	432.5	-	564	-	-
CAS [22]	439	-	559	21552	-

Furthermore figure 1 and figure 2 respectively represent an initial and final solution of experiment Oliver30 with our discrete bat algorithm.

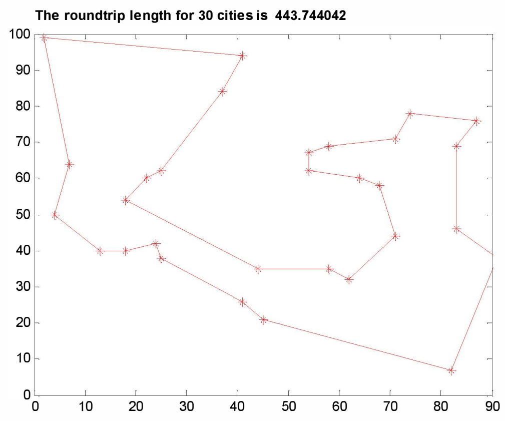

Figure 1. Initial solution of DBA with cost: 443.744042

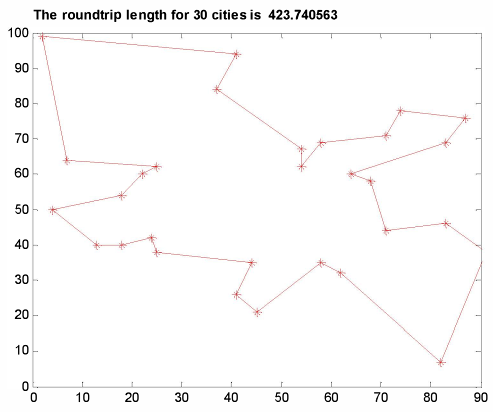

Figure 2. Final solution of DBA with cost: 423.740563

V. Conclusion

In this study, we are proposed the first adaption of Bat algorithm to a discrete problem as an example travelling salesman problem; the experimental results above show that our algorithm is a promising approach for solving the TSP in term of speed of convergence toward an optimal solution compared to some algorithms in the literatures. Although, our algorithm can't provide yet competitive results for large scale TSPs, nevertheless the encouraging result has given us the confidence of more improvement and further studies to adapt this new approach to other applications like logistic network models, scheduling models, vehicle routing models etc.

References

[1] Chen, Y., & Zhang, P. "Optimized annealing of traveling salesman problem from the< i> n</i> th-nearest-neighbor distribution". *Physica A: Statistical Mechanics and its Applications*, 371(2), 627-632, 2006.

[2] Clerc, M. "Discrete particle swarm optimization, illustrated by the traveling salesman problem". *In New optimization techniques in*

engineering (pp. 219-239). Springer Berlin Heidelberg. Computation 3(5), 4267–4274 32, 2004.

[3] Dorigo, M., & Gambardella, L. M. "Ant colonies for the travelling salesman problem". *BioSystems*, 43(2), 73-81, 1997.

[4] Dorigo, M., & Gambardella, L. M. "Ant colony system: A cooperative learning approach to the traveling salesman problem". *Evolutionary Computation, IEEE Transactions on*, 1(1), 53-66 , 1997a.

[5] Gandomi, A. H., Yang, X. S., Alavi, A. H., & Talatahari, S. "Bat algorithm for constrained optimization tasks". *Neural Computing and Applications*, 1-17, 2013.

[6] Goldbarg, E. F., Goldbarg, M. C., & de Souza, G. R. "Particle swarm optimization algorithm for the traveling salesman problem". *Greco*, 202-224. 2008.

[7] Lawler, E. L., Lenstra, J. K., Rinnooy Kan, A. H., & Shmoys, D. B. "The traveling salesman problem: a guided tour of combinatorial optimization". Johh Wiley & Sons, New York, NY, 1985.

[8] Li, X., Tian, P., Hua, J., & Zhong, N. "A hybrid discrete particle swarm optimization for the traveling salesman problem". *In Simulated Evolution and Learning* (pp. 181-188). Springer Berlin Heidelberg, 2006.

[9] Marinakis, Y., Migdalas, A., & Pardalos, P. M. "Expanding neighborhood GRASP for the traveling salesman problem". *Computational Optimization and Applications*, 32(3), 231-257, 2005.

[10] Ouaarab, A., Ahiod, B., & Yang, X. S. "Discrete cuckoo search algorithm for the travelling salesman problem". *Neural Computing and Applications*, 1-11, 2013.

[11] Potvin, J. Y. "Genetic algorithms for the traveling salesman problem". *Annals of Operations Research*, 63(3), 337-370, 1996.

[12] Qu, L., & Sun, R. "A synergetic approach to genetic algorithms for solving traveling salesman problem". *Information Sciences*, 117(3), 267-283, 1999.

[13] Yang, X. S., & Gandomi, A. H. "Bat algorithm: a novel approach for global engineering optimization". *Engineering Computations*, 29(5), 464-483, 2012.

[14] Yang, X.-S. "A new metaheuristic Bat-inspired algorithm".Nature inspired cooperative strategies for optimization (NICSO), *Studies in Computational Intelligence,* vol. 284, pp, 2010.

[15] Yang, X. S. "Bat algorithm for multi-objective optimisation". *International Journal of Bio-Inspired Computation*, *3*(5), 267-274, 2011.

[16] Zachariasen, M., & Dam, M. "Tabu search on the geometric traveling salesman problem". *In Meta-Heuristics* (pp. 571-587). Springer US, 1996.

[17] Stützle, T., & Hoos, H. H. "MAX–MIN ant system". Future generation computer systems, 16(8), 889-914, 2000.

[18] Gambardella, L. M., & Dorigo, M. "Solving Symmetric and Asymmetric TSPs by Ant Colonies". In International conference on evolutionary computation (pp. 622-627), (1996, May).

[19] Soak, S. M., & Ahn, B. H. "New genetic crossover operator for the TSP". In *Artificial Intelligence and Soft Computing-ICAISC* (pp. 480-485). Springer Berlin Heidelberg, 2004.

[20] Shi, X. H., Liang, Y. C., Lee, H. P., Lu, C., & Wang, Q. X. "Particle swarm optimization-based algorithms for TSP and generalized TSP". *Information Processing Letters*, 103(5), 169-176, 2007.

[21] Liu, Y., Xiong, S., & Liu, H. "Hybrid simulated annealing algorithm based on adaptive cooling schedule for TSP". In *Proceedings of the first ACM/SIGEVO Summit on Genetic and Evolutionary Computation* (pp. 895-898). ACM, (2009, June).

[22] Wei, Z., Ge, F., Lu, Y., Li, L., & Yang, Y. "Chaotic ant swarm for the traveling salesman problem". Nonlinear dynamics, 65(3), 271-281, 2011.