Decision Making: Applications in Management and Engineering
Vol. 5, Issue 2, 2022, pp. 287-299.
ISSN: 2560-6018
eISSN: 2620-0104
DOI: https://doi.org/10.31181/dmame0318062022m

A NOVEL DISCRETE RAT SWARM OPTIMIZATION (DRSO) ALGORITHM FOR SOLVING THE TRAVELING SALESMAN PROBLEM

Toufik Mzili[1*], Mohammed Essaid Riffi [1], Ilyass Mzili[2] and Gaurav Dhiman[3]

[1*, 1, 2] Department of Computer Science, Laboratory LAROSERI, Faculty of Science, Chouaib Doukkali University, EI Jadida, Morocco
[3]Department of Computer Science, Government Bikram College of Commerce, Patiala-147001, Punjab, India

Received: 11 February 2022;
Accepted: 25 May 2022;
Available online: 18 June 2022.

Original scientific paper

Abstrac: *Metaheuristics are often used to find solutions to real and complex problems. These algorithms can solve optimization problems and provide solutions close to the global optimum in an acceptable and reasonable time. In this paper, we will present a new bio-inspired metaheuristic based on the natural chasing and attacking behaviors of rats in nature, called a Rat swarm optimizer. Which has given good results in solving several continuous optimization problems, and adapted it to solve a discrete, NP-hard, and classical optimization problem that is the traveling salesman problem (TSP) while respecting the natural behavior of rats. To test the efficiency of the adaptation of our proposal, we applied the adapted rat swarm optimization (RSO) algorithm to some reference instances of TSPLIB.*
The obtained results show the performance of the proposed method in solving the traveling salesman problem (TSP).

Keywords: *Travelling Salesman Problem, Rat swarm Optimization Combinatorial optimization, Metaheuristic, and Nature-inspired.*

1. Introduction

The traveling salesman problem (TSP) (Pintea et al., 2017) This is an NP-Hard (Tanaev et al., 1994) problem in combinatorial optimization, in which the traveling salesman wishes to visit a certain number of cities, starting and ending its route with the same city of departure, visiting these cities only once, making the shortest possible route. This is an important problem in theoretical computer science and operations research.

* Corresponding author.
E-mail addresses: Toufikmzili95@gmail.com(T.M),saidriffi2@gmail.com(M.R), dr.mzili.ilyass@gmail.com(I.M), gdhiman0001@gmail.com(G.D)

TSP has several applications, even in its purest formulation, such as astronomy, logistics, transportation, and telecommunications.

the resolution of this problem in a reasonable execution time led the researchers to propose an approximation algorithm such as heuristics, such as simulated annealing (SA) (Johnson et al., 1991), Taboo search (TS) (Glover, 1989), local search (Korupolu et al., 2000), etc. and metaheuristics such as Ant Colony (Wang, 2012), genetic algorithm (GA) (Rybickova et al., 2014), Particle Swarm Optimization (PSO) (Wang et al., 2003), Hybrid methods: A novel hybrid penguins search optimization algorithm (Mzili et al., 2015a), discrete optimization of the search for penguins (Mzili et al., 2015b), New discrete hybrid PSO (Bouzidi & Riffi, 2014), Particle Swarm Optimization with Simulated Annealing (Fang et al., 2007), Discrete cat swarm optimization (Bouzidi & Riffi, 2013), Elephants Herding Optimization (EHO)(Hossam et al., 2019), Ant colony optimization (ACO)(Bao, 2015), Artificial bee colony (ABC), A Hierarchic Approach based Swarm Intelligence (Gündüz et al., 2015), Discrete social spıder (Baş & Ülker, 2021), etc.

In general, the most commonly used heuristics are based on nature, animal behavior, and artificial intelligence. These algorithms have several advantages over other heuristics: they can contain information about the entire search space and are very easy to implement. These algorithms have fewer parameters, which means they require less memory than other metaheuristics (Dhiman et al., 2021). These algorithms can explore the right balance between search spaces by traversing the entire operation space to find the optimal value.

These advantages are a huge motivation to adopt a new Bio-Inspire-based meta-heuristic algorithm (called RSO), which has been recently developed to solve continuous optimization problems such as the pressure vessel problem, and the gearbox design problem. Welded beam design problem, Tension/compression spring design problem, Rod support design problem, Bearing design problem.

The results are much better than the seven best-known meta-heuristics and are robust (Dhiman et al., 2021).

The objective of this work is to adapt the Rat swarm Optimization (RSO) algorithm, introduced in 2020 by Gaurav Dhiman, to solve the traveling salesman problem (TSP) a classical discrete optimization problem, very well known for its complexity and very useful in several domains. A new fitness function based on the Euclidean distance is proposed to deal with the discontinuity, which has been neglected in other algorithms.

This adaptation consists in reconstructing again this method by introducing new mathematical operators and by modifying just the values of the parameters of the method. without touching the definition of the proposed rat behavior.

The organization of this research is as follows: In Sect. 2, a presentation of the traveling salesman problem; In Sect. 3, a presentation of the RSO algorithm introduced to solve continued optimization problems. In Sect. 4, the adaptation of the RSO Optimization Algorithm to solve the traveling salesman problem. In Sect. 5, the results of tests using TSPLIB instances. Finally, comes the conclusion in the last section.

2. The Travelling Salesman Problem

The traveling salesman problem (TSP) is one of the oldest and most studied combinatorial optimization problems. this problem aims to find the shortest circuit which allows him to visit a certain number of cities and pass once and only once per city and return to his starting point, at a lower cost, by covering the shortest distance

possible. The distances between cities are known. We must find the path that minimizes the distance traveled.

2.1. The importance of resolving TSP:

The traveling salesman problem consists of determining whether it is possible to travel through n cities in such a way that the sum of the distances traveled in each city is the least costly. Solving the traveling salesman problem is usually very time-consuming. Therefore, new strategies must be found for the solvers. The Trade Traveler Problem (TSP) essentially aims to find the shortest route through a set of points to minimize the cumulative cost of travel overall routes. Since a solution must determine the number of cities, it cannot always successfully solve large-scale problems. As such, TSP is one of the NP-complete tasks, which means that although there are efficient algorithms, none of them are sure to stop in a finite time. This makes solving TSP more difficult, more important, and more motivating.

3. RSO Algorithm

Gaurav Dhiman introduced the Rat Swarm Optimizer (RSO) (Dhiman et al., 2021) in 2021 to solve continuous optimization problems. The RSO algorithm is inspired by the chasing and attacking behaviors of rats. Rats are long-tailed and medium-sized rodents, socially intelligent by nature and they are territorial animals that live in groups of two males and females, they participate in various activities such as jumping, running, and tumbling. And boxing. But they are very aggressive, which in many cases results in the death of some animals. This aggressive behavior when hunting and fighting with prey gave rise to this algorithm. The hunting and fighting behaviors of the rats are mathematically modeled to design the RSO algorithm and perform the optimization.

3.1. Rats behavior modeling

This subsection describes the behavior of rats, chasing and fighting. Then the proposed RSO algorithm is outlined.

3.2. Chasing the prey

In general, rats are social animals that hunt prey in groups due to their agonistic social behavior. To define this behavior mathematically, we assume that the best researcher knows the location of the prey. Other search agents can update their positions against the best search agent obtained so far. To model this mechanism the following equations are proposed:

$$X = A \times X_i + C \times (X_{Best} - X_i) \quad (1)$$

Where $\vec{X}$ represents the positions of rats and X is the best optimal solution. A and C are calculated as follows:

$$A = R - x\left(\frac{R}{Max_{Iteration}}\right) \ , \ 1 \leq R \leq 5 \quad (2)$$

Where x in [0 , 1, 2…Max-iteration].Therefore, R and C are random numbers between [1, 5] and [0, 2], respectively. Parameters A and C are responsible for better exploration and exploitation during iterations.

3.3. Fighting with prey

The process of fighting rats with prey is mathematically defined by the following equation:

$$X_{i+1} = |X_{Best} - X_i| \quad (3)$$

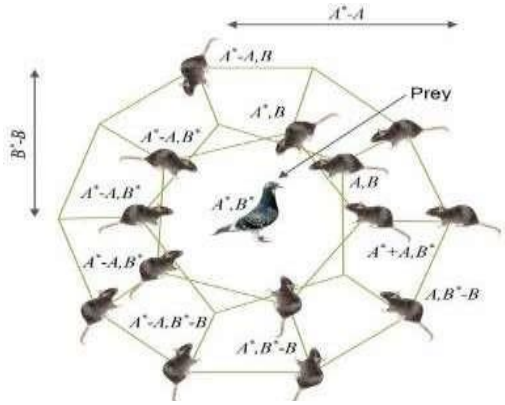


Figure 1. 3D position vectors of rats

Where X_{i+1} defines the newly updated position of the rat. It saves the best solution and updates the positions of other search agents against the best search agent.

Figure 1 shows how rats change their positions according to equations (1) and (3) in a three-dimensional environment. In this figure, the rat (A, B) can actualize its position toward the prey position (A*, B*). By adjusting the parameters, as shown in Eq. (2). the different number of positions can be reached around the current position. The adjusted values of parameters A and C guarantee good exploration and exploitation. The RSO algorithm will record the optimal solution with the fewest operators.

3.4. Rat Swarm Optimization (RSO) algorithm

Begin:
Step 1: Initialize the rats population Pi where i = 1, 2...n.
Step 2: Choose the initial parameters of RSO: A, C, and R.
Step 3: Now, calculate the fitness value of each search agent.
Step 4: The best search agent is then explored in the given search space.
Step 5: Update the positions of search agents using Eq(1).
Step 6: Check whether any search agent goes beyond the boundary limit of a search space and then amend it.
Step 7: Again, calculate the updated search agent fitness value and update the vector Xbest if there is a better solution than the previous optimal solution.
Step 8: Stop the algorithm if the stopping criteria are satisfied. Otherwise, return to Step 5.
Step 9: Return the best obtained optimal solution.
END

4. Use RSO to Solve the TSP

This section presents the adaptation of the RSO method to solve a TSP. The adaptation of RSO consists in redefining the algebraic operators of the algorithm and the structures and stages of this algorithm

4.1. Adapted Discrete RSO to Solve the TSP

The RSO method (Figure 2) proposed by Gaurav Dhiman was defined to solve continuous optimization problems, so it cannot be applied to solve discrete combinatorial optimization problems, since in continuous optimization a number represents the solution, on the other hand in combinatorial optimization, the solution is represented by an order which can be modeled as a vector or a sequence of numbers. To adapt the RSO to solve the problems of discrete combinatorial optimization, it is necessary to adapt the operations and the operators while respecting the real behavior of the Rats:

- *The position of a Rat represents a solution and it is modeled as a vector of integers where each integer represents a city X = {1 2 3 4 5}.*
- *Subtraction between each of the* two positions (X - Y) presents the list of permutations to apply to the Y path (vector) to obtain the X path:
  - X = {1 2 3 4 5} and Y = {1 2 4 3 5}
  - Then Q=X - Y
  - Return Q= {(2, 3) (4, 5)}..
- The operation of multiplication (*) is an operation performed between a real k [0;1] and a set of permutations Q,
- The result Q' is part of the set Q according to the value of k.
- The addition operation $\oplus$ is an effective operation between the solution $\vec{X}$ and the set of permutations Q, the result is a new solution X'. This operation consists in applying permutations of Q' on X to obtain a new X' solution.
  - X= {1 2 3 4 5 6}
  - Q = {(1, 2) (4, 5)}
  - X' =X $\oplus$ Q
  - X'= {2 1 3 5 4 6}.

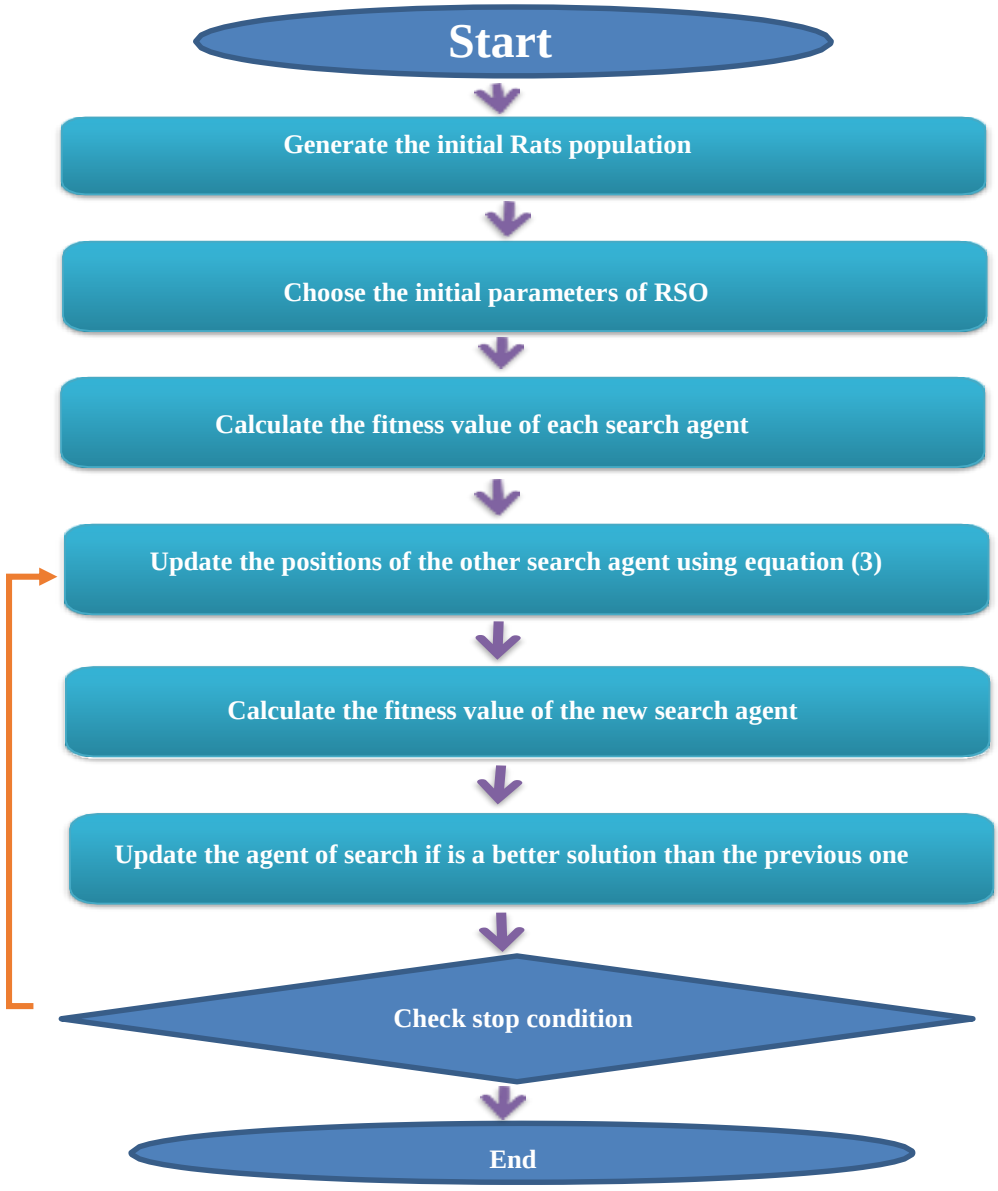


Figure 2. The Flowchart of the RSO Algorithm

5. Experimental Results and Comparison

5.1. Experimental results

The implementation of the RSO optimization algorithm adapted to solve the TSP was carried out on the programming language C ++, and the simulations were carried out on a personal computer equipped with a CORE i7-3540 M CPU at 3.00 GHz, 8 GB

of RAM, and Windows 10 (64 bits). Table 1 shows the results of the executions of this algorithm on several different reference instances of TSPLIB.

The parameters were set as follows: the number of rats was set at 100, the number of iterations varied between 6000 and 8000 depending on each instance, C is a random variable between 0 and 1, and R and X remain the same as in the original algorithm.

The table displays the following information:

- Inst: name of the benchmark instance in the TSPLIB library.
- Nb.node: number of nodes.
- Opt: the best-known solution for the instance.
- BestR: the best solution found by the algorithm after ten different executions.
- WorstR: the worst solution found by the algorithm after ten different executions.
- Average: the average of ten different executions of the algorithm
- Time: displays the average time in seconds of ten different executions of the algorithm
- Err: (%) is the percentage relative error
- PDbest(%): the percentage deviation of the best solution length from the optimal

Table 1. Results produced by RSO method.

Instance	Opt	BestR	Average	WorstR	Err(%)	PDbest	Time(s)
eil51	426	426	432,57	442	1,54	0	6,76
berlin52	7542	7542	7788,79	8111	3,27	0	3,84
st70	675	675	685,46	699	1,55	0	9,39
Oliver30	420	420	421,62	426	0,38	0	0,39
eil76	538	549	569,5	591	5,85	2,00	13,35
kroA100	21282	21353	21748,4	21986	2,19	0,33	18,76
kroB100	22141	22337	23165,5	23929	2,62	0,87	9,50
eil101	629	653	672,62	696	6,93	3,67	0,39
ch130	6,110	6275	6507	6816	6,49	2,62	13,53
Rat99	1211	1229	1274,44	1308	5,23	1,46	0,39
D198	15780	16045	16517,8	16994	4,67	1,65	19,89

5.2. Comparison and discussion:

In this part, we will try to compare the results of the application of the RSO with the results of other more known metaheuristics (Table 2 and Figures 3-6). The results of ACO (ant colony optimization) were taken from (Bao, 2015), the results of ABC (artificial bee colony) (Gündüz et al., 2015), and the results of HA (Wang, 2011).

Table 2. Comparison of experimental results of Improved DSRO with ACO, ABC, HA, and DSSA.

Instance	Method	Best	Mean	Worst	Time(s)
Oliver30	RSO	420	421,62	426	0,39
	ACO	423.74	424.68	429.36	35.20
	ABC	439.49	462.55	484.83	1.26
	HA	423.74	423.74	423.74	19.63
	DSSA	-	-	-	-
Eil51	RSO	426	432,57	442	6,76
	ACO	450.59	457.86	463.55	112.11
	ABC	563.75	590.49	619.44	2.16
	HA	431.74	443.39	454.97	58.33
	DSSA	431.87	-	483.53	-
Berlin52	RSO	7542	7788,79	8111	3,84
	ACO	7548.99	7659.31	7681.75	116.67
	ABC	9479.11	10,390.26	11,021.99	2.17
	HA	7544.37	7544.37	7544.37	60.64
	DSSA	7659	-	31432	-
St70	RSO	675	685,46	699	9,39
	ACO	696.05	709.16	725.26	226.06
	ABC	1162.12	1230.49	1339.24	3.15
	HA	687.24	700.58	716.52	115.65
	DSSA	-	-	-	-
Eil76	RSO	549	569,5	591	13,35
	ACO	554.46	561.98	568.62	271.98
	ABC	877.28	931.44	971.36	3.49
	HA	551.07	557.98	565.51	138.82
	DSSA	559,31	-	2720,4	-
Kroa100	RSO	21353	21748,4	21986	18,76
	ACO	22455.89	22,880.12	23,365.46	615.06
	ABC	49519.51	53,840.03	57,566.05	5.17
	HA	22122.75	22,435.31	23,050.81	311.12
	DSSA	21363	-	189,380	-
Eil101	RSO	653	672,62	696	0,39
	ACO	678.04	693.42	705.65	527.42
	ABC	1237.31	1315.95	1392.64	5.17
	HA	672.71	683.39	696.04	267.08
	DSSA	-	-	-	-

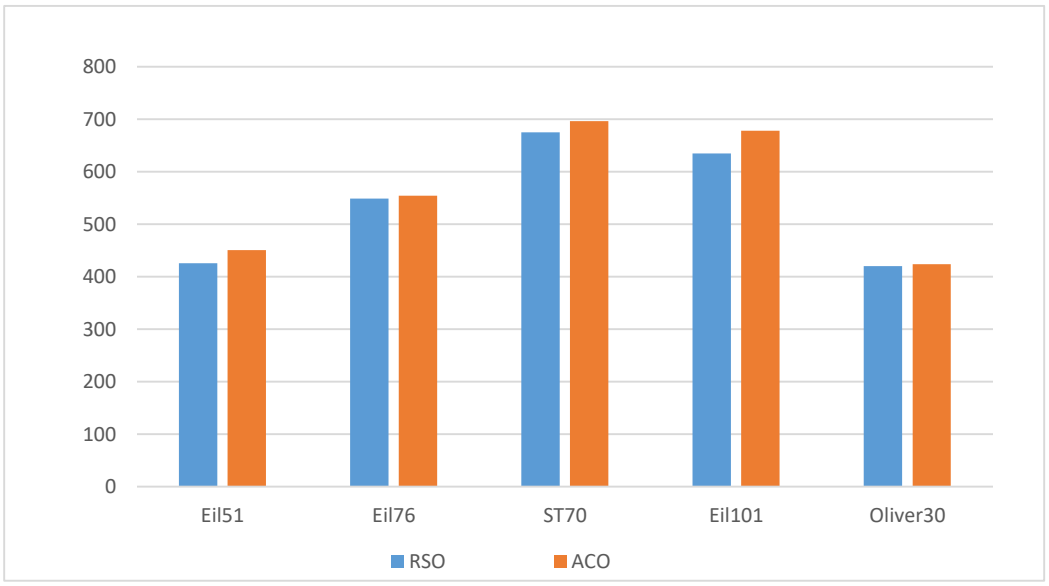


Figure 3. Comparison of experimental results of Improved DSRO with ACO

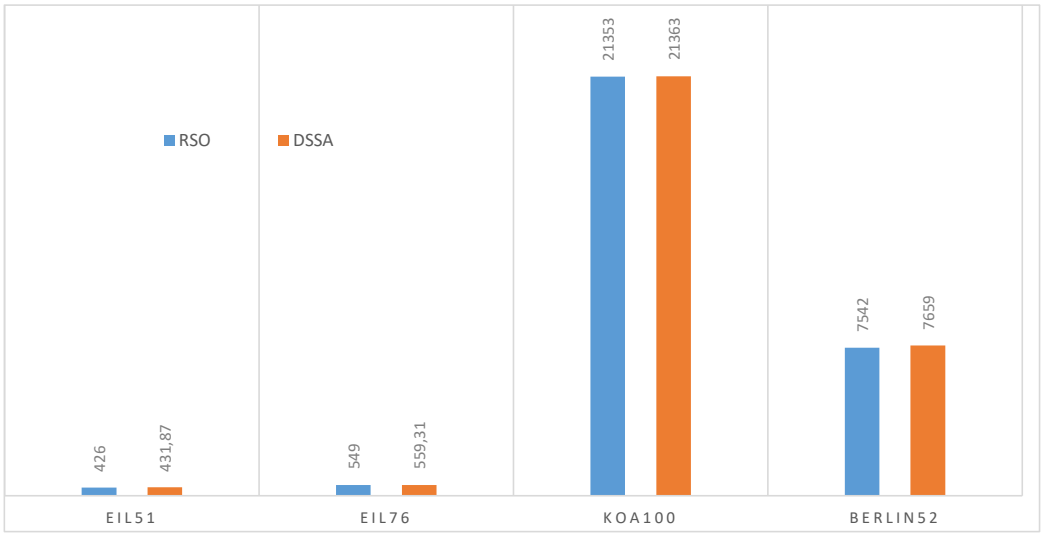
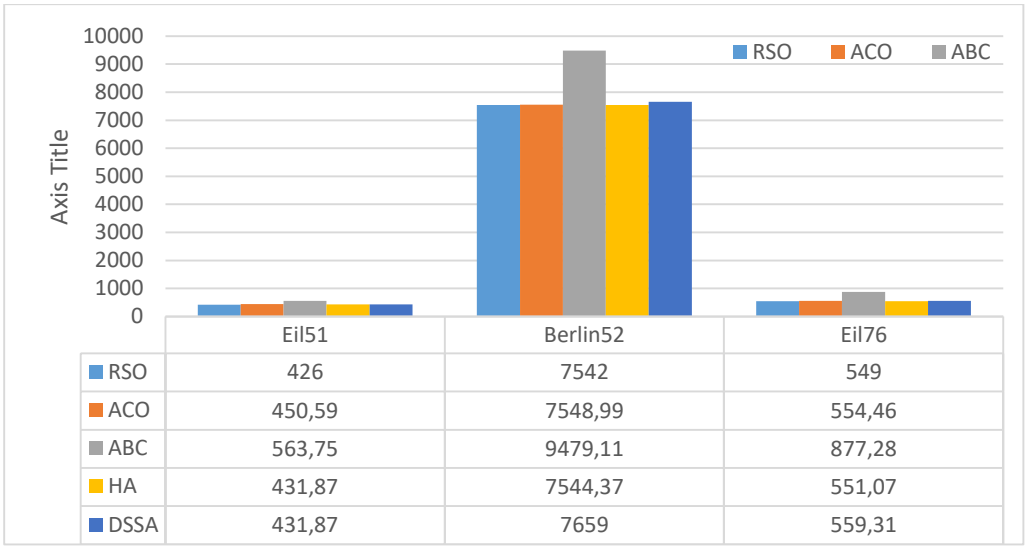


Figure 4. Comparison of experimental results of Improved DSRO with DSSA



	Eil51	Berlin52	Eil76
RSO	426	7542	549
ACO	450,59	7548,99	554,46
ABC	563,75	9479,11	877,28
HA	431,87	7544,37	551,07
DSSA	431,87	7659	559,31

Figure 5. Comparison of experimental results of Improved DSRO with ACO, ABC, HA, and DSSA

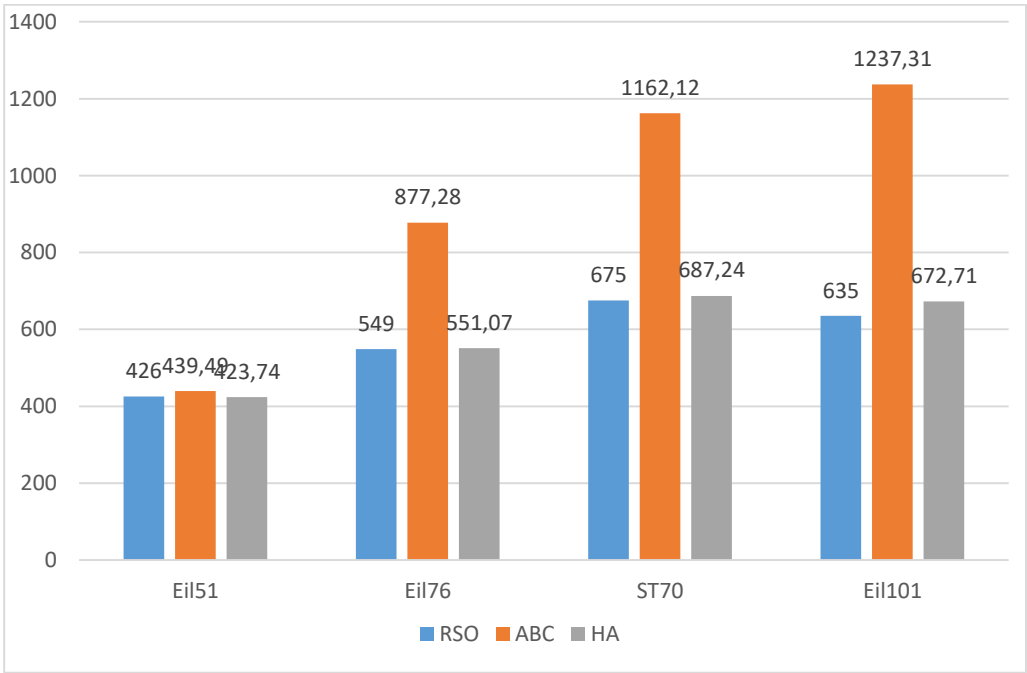


Figure 6. Comparison of experimental results of Improved DSRO with ABC, HA

6. Discussion

The following examples show a comparison of the objective function values of the proposed method with other existing metaheuristics for the Oliver30, Eil51, Berlin52, St70, Eil76, Kroa100, and Eil101 instances of TSPLIB.

The results in Table 2 confirm that the RSO algorithm can solve multiple TSPLIB instances in a very reasonable runtime.

To prove the robustness of the algorithm, Table 1 compares the average solution proposed by the RSO algorithm with other methods applied to solve the TSP using TSPLIB.

The table also compares the execution time obtained by the algorithm with the other bio-inspired algorithm that solves TSP.

According to the results in Table 2 and Figures 3, 4, and 5, the results obtained by the RSO algorithm are good compared to the other methods and this is evident from the RSO curve that is lower than the other methods.

This can be justified by the simplicity of RSO and its parameters, which can guarantee a much faster convergence than other algorithms.

In some cases, this algorithm gave results close to the optimum without reaching it, is it because this algorithm is guided by a single search agent which is the best global search agent, and after several iterations, all the agents and solutions converge towards this optimum.

To solve this limitation and to make this optimizer more robust, we will consider adding other improvements or hybridization heuristics.

Conclusion

In this paper, we first presented an adaptation of the RSO algorithm proposed by Gaurav Dhiman without hybridization to solve the symmetric PSD. This adaptation achieved good performance compared to several metaheuristics.

Discrete rat swarm optimization (DRSO) is one of the most intelligent and powerful algorithms. This algorithm can be used to solve any optimization problem, including the traveling salesman problem (TSP). this algorithm has great potential to become a very powerful optimization technique. In DRSO, each rat is a simple entity capable of moving in one dimension to find the best path in its environment. Each rat can visit a point in its neighborhood or search for a new point in a radius area around the current point.

The algorithm has been tested on a set of benchmark instances of TSPLIB. Its performance exceeds that of recent methods used to solve TSP, such as ACO, ABC, HA, and DSSA. Moreover, the robustness and speed of the RSO algorithm encourage its use to solve other combinatorial optimization problems.

In the future, we will try to improve this algorithm to obtain better results than the majority of methods and we aim to extend the algorithm to apply it to various application domains and solve any discrete optimization problem such as network optimization, scheduling, transportation problems, vehicle routing problem, electronic manufacturing units, etc.

Several improvements have been made to the algorithm to solve the quadratic assignment problem, a new problem as important as the TSP, designed to minimize the overall cost of building and operating a facility.

Author contributions: Research problem, M.R., T.M. and I.M.; Conceptualization, M.R., T.M. and I.M.; Methodology, M.R., T.M., G.D., and I.M.; Formal analysis, M.R., T.M.; Resources, M.R., I.M.; Original drafting, M.R. and T.M.; Reviewing and editing, M.R., T.M., I.M; Project administration, M.R., T.M., IM; Supervision, M.R., I.M.; Proposal, improvement and ideation, M.R., I.M, G.D.
All authors have read and approved the published version of the manuscript.

Funding: This research received no external funding**.**

Acknowledgment: The authors would like to express their gratitude to the editors and anonymous referees for their informative, helpful remarks and suggestions to improve this paper as well as the important guiding significance to our research.

Conflict of Interest: The authors declare that they have no conflict of interest.

References

Bao, H. (2015). A two-phase hybrid optimization algorithm for solving complex optimization problems. Int J Smart Home, 9(10), 27-36.

Baş, E., & Ülker, E. (2021). Dıscrete socıal spıder algorıthm for the travelıng salesman problem. Artificial Intelligence Review, 54(2), 1063-1085.

Bouzidi, A., & Riffi, M. E. (2013). Discrete cat swarm optimization to resolve the traveling salesman problem. International Journal of Computer Science and Software Engineering, 3(9), 13-18.

Bouzidi, M., & Riffi, M. E. (2014). Discrete novel hybrid particle swarm optimization to solve travelling salesman problem. In 2014 5th Workshop on Codes, Cryptography and Communication Systems (WCCCS) (pp. 17-20). IEEE.

Dhiman, G., Garg, M., Nagar, A., Kumar, V., & Dehghani, M. (2021). A novel algorithm for global optimization: rat swarm optimizer. Journal of Ambient Intelligence and Humanized Computing, 12, 8457-8482.

Fang, L., Chen, P., & Liu, S. (2007). Particle swarm optimization with simulated annealing for TSP. In Proceedings of the 6th WSEAS International Conference on Artificial Intelligence, Knowledge Engineering and Data Bases (pp. 206-210).

Glover, F. (1989). Tabu Search—Part I. ORSA Journal on Computing 1(3), 190-206.

Gündüz, M., Kiran, M. S., & Özceylan, E. (2015). A hierarchic approach based on swarm intelligence to solve the traveling salesman problem. Turkish Journal of Electrical Engineering and Computer Sciences, 23(1), 103-117.

Hossam, A., Bouzidi, A., & Riffi, M. E. (2019). Elephants herding optimization for solving the travelling salesman problem. In Advanced Intelligent Systems for Sustainable Development (AI2SD'2018) Vol 2: Advanced Intelligent Systems Applied to Energy (pp. 122-130). Springer International Publishing.

Mzili, I & Riffi, M. (2015a). Discrete Penguins search optimization algorithm to solve the traveling salesman problem. Journal of Theoretical and Applied Information Technology. 72(3), 331-336.

Mzili, I., Bouzidi, M., & Riffi, M. E. (2015b). 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.

Korupolu, M. R., Plaxton, C. G., & Rajaraman, R. (2000). Analysis of a local search heuristic for facility location problems. Journal of algorithms, 37(1), 146-188.

Pintea, C. M., Pop, P. C., & Chira, C. (2017). The generalized traveling salesman problem solved with ant algorithms. Complex Adaptive Systems Modeling, 5(1), 1-9.

Johnson, D., Aragon, C., McGeoch, L., & Schevon, C. (1989). Optimization by Simulated Annealing: An Experimental Evaluation. Part I, Graph Partitioning. Operations Research. 37, 865-892.

Rybickova, A., Mockova, D., & Karaskova, A. (2014). Application of Genetic Algorithm to The TsP Problem. Paripex - Indian Journal Of Research. 3(7), 105-107.

Tanaev, V. S., Gordon, V. S., & Shafransky, Y. M. (1994). NP-Hard Problems. Scheduling Theory. Single-Stage Systems, 253-311.

Wang, X., & Xu, G. (2011). Hybrid differential evolution algorithm for traveling salesman problem. Procedia Engineering, 15, 2716-2720. https://doi.org/10.1016/j.proeng.2011.08.511.

Wang, K., Huang, L., Zhou, C., & Pang, W. (2003). Particle swarm optimization for traveling salesman problem. In 2003 International Conference on Machine Learning and Cybernetics (Vol. 3, pp. 1583-1585). IEEE Press. https://doi.org/10.1109/ICMLC.2003.1259748.

Wang, Z. Y. (2012). An improved ant colony algorithm for solving TSP problems. Mathematics in Practice and Theory, 24(4), 133-140.