Dokuz Eylül Üniversitesi Mühendislik Fakültesi Fen ve Mühendislik Dergisi
Dokuz Eylul University Faculty of Engineering Journal of Science and Engineering

Basılı/Printed ISSN: 1302-9304. Elektronik/Online ISSN: 2547-958X

Application of a Hybrid Camel Traveling Behavior Algorithm for Traveling Salesman Problem

Bir Hibrid Deve Gezgin Davranışı Algoritmasının Gezgin Satıcı Problemi için Uygulaması

Mehmet Fatih Demiral [1*]

[1] Burdur Mehmet Akif Ersoy Üniversitesi Mühendislik-Mimarlık Fakültesi Endüstri Mühendisliği Bölümü, Burdur, TÜRKİYE
*Sorumlu Yazar / Corresponding Author **: mfdemiral@mehmetakif.edu.tr

Geliş Tarihi / Received: 23.09.2021 *Araştırma Makalesi/Research Article*
Kabul Tarihi / Accepted: 16.02.2022 *DOI:10.21205/deufmd.2022247204*
Atıf şekli/ How to cite: DEMIRAL, M.F.(2022). Application of a Hybrid Camel Traveling Behavior Algorithm for Traveling Salesman Problem. DEUFMD 24(72), 725-735.

Abstract

Camel Traveling Behavior Algorithm (CA) is a nature-inspired meta-heuristic proposed in 2016 by Mohammed Khalid Ibrahim and Ramzy Salim Ali. There exist few publications that measure the performance of the CA on scientific literature. CA was implemented to global optimization and some engineering problems in the literature. It was shown that CA demonstrates better performance than Particle Swarm Optimization (PSO) and Genetic Algorithm (GA) in global optimization. However, it gives poor solutions at combinatorial optimization as well as in traveling salesman problems (TSP). Besides, a modified camel algorithm (MCA) was applied in the field of engineering and was proved that it is better than Cuckoo Search (CS), PSO, and CA. Therefore, it is a need for improvement in CA by hybridizing with a constructive heuristic (Nearest Neighbor Algorithm-NN). A set of thirteen small and medium-scale datasets that have cities scales ranging from 29 to 195 was used in the comparative study. The results show that the hybrid algorithm (HA) outperforms Tabu Search (TS), GA, CA, and Ant system (AS) for 70% of all datasets, excluding wi29, eil76, pr76, and rat99. Also, it was given that a detailed analysis presents the number of best, worst, average solutions, standard deviation, and average CPU time. The metrics also stress that the hybrid meta-heuristic demonstrates 64% performance in finding acceptable solutions. Finally, the hybrid algorithm solves the discrete problem in short computational times when compared to other test algorithms for small and medium-scale datasets.

Anahtar Kelimeler: *Camel Algorithm, Hybrid Algorithm, Meta-heuristics, Traveling Salesman Problem*

Öz

Deve gezgin davranışı algoritması (CA) 2016 yılında Mohammed Khalid Ibrahim ve Ramzy Salim Ali tarafından önerilmiş, doğadan ilham alan bir meta-sezgiseldir. Bilimsel literatürde CA' nın performansını ölçen birkaç çalışma bulunmaktadır. CA literatürde global optimizasyon ve mühendislik problemlerine uygulanmıştır. CA' nın global optimizasyonda parçacık sürü optimizasyonu ve genetik algoritmadan daha iyi performans sergilediği gösterilmiştir. Buna karşın, bu algoritma gezgin satıcı probleminde olduğu gibi kombinatoryal optimizasyonda düşük kaliteli çözümler vermektedir. Bunun yanında, değiştirilmiş deve algoritması mühendislik alanında uygulanmıştır ve CS, PSO, CA' dan daha iyi olduğu ortaya koyulmuştur. Bu sebeple, CA' nın tur oluşturucu bir sezgiselle (En yakın komşu algoritması-NN) hibrid edilerek iyileştirilmesi ihtiyacı bulunmaktadır. Bu karşılaştırmalı uygulamada, 29-195 arasında değişen boyutlarda şehir içeren 13 küçük ve orta ölçekli veriseti kullanılmıştır. Sonuçlar, hibrid algoritmanın (HA), tabu arama (TS), GA, CA, ve karınca sistemine (AS) göre wi29, eil76, pr76, ve rat99 dışında bütün verisetlerinin %70 inde daha üstün olduğunu göstermektedir. Çalışmada, detaylı bir analiz verilerek en iyi, en kötü, ortalama çözümler, standard sapma, ve ortalama CPU zamanları sunulmaktadır. Metrikler, ayrıca hibrid meta-sezgiselin kabul edilebilir çözümleri bulmada 64% performans sergilediğini vurgulamaktadır. Sonuç olarak, hibrid algoritma küçük ve orta ölçekli verisetlerinde diğer test algoritmalarına kıyasla kesikli problemi daha kısa hesaplama zamanlarında çözmektedir.

Keywords: *Deve Algoritması, Hibrid Algoritma, Meta-sezgiseller, Gezgin Satıcı Problemi*

1. Introduction

Combinatorial optimization is a challenging research field in operations research. In addition, solving discrete problems using meta-heuristics is a popular methodology in the scientific literature [1-3]. Heuristics are generally problem-based approaches that bring near-optimal solutions to specific optimization problems [4-7]. On the other hand, meta-heuristics have been successfully applied to many combinatorial problems in literature. There are lots of meta-heuristics developed in recent years. Some important meta-heuristics are simulated annealing (SA), tabu search (TS), genetic algorithm (GA), ant colony optimization (ACO), harmony search (HS), artificial immune system (AIS), differential evolution (DE), firefly algorithm (FFA), bat algorithm (BA), and black-hole algorithm [8-17].

Meta-heuristics do not guarantee optimal solutions. Thus, the deviations of solutions are getting larger when the size of problem data increases. In principle, the most appropriate methodology should be used with the preferred data size. Population-based meta-heuristics generally give better solutions than trajectory-based meta-heuristics. The initial population may be randomized or start with a constructive heuristic. Initially, a randomized population demonstrates lower performance than a population that starts with a heuristic. Starting with a heuristic provides more diversification and also intensification during the computational process. The algorithm searches unexplored areas and would find new solutions in the solution space. Conversely, intensification means that the algorithm adjusts the exploitation and converges to the optimal solution. Improving and hybridizing meta-heuristics are convenient ways for optimization. They generally give better solutions than other algorithms [18-22]. Meta-heuristics find application in mathematics, engineering, computer technology, production, logistics, other arts, and sciences.

Camel algorithm stimulates the traveling behavior of camel caravan in the desert under difficult conditions. The algorithm searches optimal solutions at an initial level of endurance and supply in cases that change according to a mathematical equation. However, temperature affects endurance and changes randomly in computation. CA generally solves the optimization problems at a reasonable level [23-26]. Although modified camel algorithm is a good methodology for engineering optimization, there is still a gap for research to improve the performance of CA [24-26].

Camel herds algorithm (CHA) is another swarm intelligent algorithm that depends on the behavior of the camels in the natural wild conditions, each herd has a leader, searching food and water under a level of humidity with neighboring strategy [27, 28].

In this study, the hybrid algorithm is applied to measure the performance in solving the traveling salesman problem (TSP). The traveling salesman problem is an NP-hard problem for evaluating the performance analysis of the optimization algorithms. The objective of the salesman is to find the optimal tour that covers all cities and return to the initial city.

The rest of the paper is organized as follows: In Sect. 2, the discrete problem and the used methodology are clearly explained. The experimental results are given in Sect. 3. Finally, in Sect. 4, the conclusions and future work are discussed.

2. Materials and Methods

2.1. Traveling Salesman Problem

The traveling salesman problem (TSP) is a broadly discussed and popular benchmark problem in the field of combinatorial optimization. The researchers try to solve the problem by using various exact, heuristic, and hybrid methods. Even if exact methods solve the problem optimally, it requires longer computational times than other methods. In traveling salesman problems, there exist dispersed locations in the space; so the objective is to minimize the distance or time under some of the side constraints, such as budget, capacity, fuel, and others. TSP can be examined as constraint, data, distance, objective, routing, salesman types, and the number of the salesman. In the past researches, TSP has been investigated in referred types and variants: symmetric TSP, asymmetric TSP, euclidean TSP, double TSP, multiple TSP, sequential ordering problem, traveling purchaser problem, capacitated vehicle routing problem, and many others [29-37]. In symmetric (basic) TSP (s-TSP), the symmetric matrices that have equal distances $\left(d_{ij} = d_{ji}\right)$ between cities are used for the problem. Otherwise, if asymmetric matrices are used $\left(d_{ij} \neq d_{ji}\right)$ for at least one edge, then the TSP turns into asymmetric TSP (a-TSP). If two salesmen travel along the tour, it specializes to double TSP (d-TSP). If multiple salesmen (m-TSP) travel, it is named as multiple TSP (m-TSP). There exist important applications of variants, such as the traveling purchaser problem and the vehicle routing problem.

Although it can be found optimal solutions to some combinatorial problems, there is no polynomial algorithm for the traveling salesman problem (TSP). It is still an NP-hard problem. If a large size of data is used, the search of feasible regions of solution space becomes impossible. When a different data structure or any complex type of problem is used, the computational time of TSP gets long. Thus, classical and more effective meta-heuristics are the potential solvers for the referred problem.

In the mathematical description of the problem, the cities, edges, sets, and vertices: T is the set of n cities, G is the set of the edges, and $D_{ij} = \left(d_{ij}\right)$ is the distance matrix between city *i* and city *j*. $P_k = \{v_1, v_2, \ldots, v_n, v_1\}$ is the permutation set of the candidate tours for $k = 1,2,3, \ldots, m$. V_1 represents the first vertex; V_n represents the *nth* vertex of all the permutations. Then, the model of the discrete problem is shortly given in Eq. 1.

$$\mathrm{Min.} \sum_{i=1}^{n-1} \left(d_{v_i, v_{i+1}}\right) + d_{v_n, v_1} \tag{1}$$

The Euclidean distance is implemented to calculate the distance between cities using Eq.2. In Eq. 2, the x_i, x_j represent the x coordinates of euclidean nodes, and the y_i, y_j represent the y coordinates of these nodes.

$$d_{i,j} = \sqrt{\left(x_i - x_j\right)^2 + \left(y_i - y_j\right)^2} \tag{2}$$

2.2. Nearest Neighbor Algorithm

The Nearest Neighbor algorithm (NN) is a greedy heuristic algorithm that starts a random tour by selecting and adding the nearest cities to the last added city in the tour until the traveling salesman visits all the cities at once. [38]. Although it solves the TSP worse than other algorithms, it strengthens the weakness of meta-heuristics when it is initially used. In this study, the NN algorithm is applied to all of the tours in the population as follows:

Step 1. Select a random city, and add it to a tour in the population
Step 2. Sort the distances to the current city (last city),
Step 3. Select the nearest unvisited city,
Step 4. Add the nearest city to the tour,
Step 5. If the tour is not completed, go to step 2.
Step 6. Exit the program if traveling salesman visits all the cities.

2.3. Discrete Hybrid Camel Algorithm

The camel algorithm (CA) has rising interest since it was put forth in the literature. The principle of the camel algorithm is based on the traveling behavior of camels during a long time horizon. In the CA, the camel caravan is looking for the optimal positions and trying to live with the food supply. Therefore, they both survive and bear the burden in the long run under strict conditions. In the camel caravan, each camel has its supply (S), temperature (T), and endurance (E). The minimum and maximum temperatures are T_{min} and T_{max}. The current temperature ($T_{now}^{i,iter}$) is changing using Eq. 3.

$$T_{now}^{i,iter} = (T_{max} - T_{min}) * rand + T_{min} \qquad (3)$$

Current endurance ($E_{now}^{i,iter}$) alters with the current temperature and increasing #of iterations. The endurance is defined as below in the basic camel algorithm using Eq. 4. Trsteps represents the current iteration number, and the Totalsteps represents the maximum iteration number. $E_{past}^{i,iter}$ indicates the previous endurance level of each camel in the population in Eq. 4.

$$E_{now}^{i,iter} = E_{past}^{i,iter} * \left(1 - \frac{T_{now}^{i,iter}}{T_{max}}\right) * \left(1 - \frac{Trsteps}{Totalsteps}\right) \qquad (4)$$

Here, it needs an improvement during the computation of the algorithm. In this study, the camel algorithm hybridizes with a constructive heuristic (Nearest neighbor), and then it is compared with the test algorithms. In the CA, the supply of camels (water and food) is randomly decreasing with the increasing number of iterations. That is so; supply is the critical factor that affects the position of the camel caravan during the journey.

$w \in (0,1]$ is a burden factor that reduces previous supply and renews the needs of camel caravan. The supply equation of camels is in the following by using Eq.5.

$$S_{now}^{i,iter} = S_{past}^{i,iter} * \left(1 - w * \frac{Trsteps}{Totalsteps}\right) \qquad (5)$$

The original position updating equation of each camel is realized in Eq. 6.

$$x_{now}^{i,j} = x_{old}^{i,j} + rand * \left(1 - \frac{E_{now}^{i,iter}}{E_{initial}^{i,iter}}\right) * exp\left(1 - \frac{S_{now}^{i,iter}}{S_{initial}^{i,iter}}\right) * \left(x_{best}^{*} - \min(x_{old}^{i,j})\right) \qquad (6)$$

In traveling salesman problem, the objective values of solutions are taken as the position of candidate solutions in dimensional space.

$$Obj_{now}^{i,j} = Obj_{old}^{i,j} + rand * \left(1 - \frac{E_{now}^{i,iter}}{E_{initial}^{i,iter}}\right) * exp\left(1 - \frac{S_{now}^{i,iter}}{S_{initial}^{i,iter}}\right) * \left(BObj - \min(Obj_{old}^{i,j})\right) \qquad (7)$$

The neighborhood solutions are compared with the objectives found via Eq. 7 in Eq. 8.

$$\begin{aligned} &NeighObj_{now}^{i,iter} < Obj_{now}^{i,iter} \\ &Obj_{new}^{i,iter} = NeighObj_{now}^{i,iter} \end{aligned} \qquad (8)$$

If the candidate solution is lower than the objective value found in Eq.7, then the new current solution is the neighborhood value. Otherwise, the new current solution is assigned as the previous value of the solution. When the camel has better solutions in the solution space, then the oasis condition will occur and the supporting factors are increased as in Eq. 9.

$$\begin{aligned} &fitness_{now}^{i,iter} > fitness_{old}^{i,iter} \\ &S_{past}^{i,iter} = S_{initial}^{i,iter} \\ &E_{past}^{i,iter} = E_{initial}^{i,iter} \end{aligned} \qquad (9)$$

In this algorithm, the dimension of space is used to select the best locations and increase the candidate solutions (j-loop). The pseudo-code of the discrete algorithm is shown in Figure 1.

```
Camel Algorithm (CA)
Initialize Camel caravan with NN algorithm.
Initialize Camel Algorithm parameter values.
Compute Camel Caravan objective function and find the current best objective.
While (Counter < Total journey steps)
For i =1: Camel Caravan
For j=1: Dimension of Space
$T_{now}^{i,iter}$ , $E_{now}^{i,iter}$ , $S_{now}^{i,iter}$ using Eqs. 3-5.
Update camels' locations using Eq. 7.
End For j
End For i
Decide the acceptance of new camels' locations using Eq. 8.
If (oasis condition occur)
Replenish Supply and Endurance using Eq. 9
End If
Rank Caravan individuals and find the best camel in the caravan
End While
State the final results (Final Statistics)
```

Figure 1. Pseudo-code of the Discrete Hybrid Algorithm (DHA)

3. Experimental Results

The 13 small and medium-scale datasets ranging from 29 to 195 nodes (cities) were selected from the TSPLIB library in the implementation. In this section, all of the experimentations were run on Intel® Core™ i7 3520-M CPU 2.9 GHz speed with 8 GB RAM using Matlab. The meta-heuristics which are CA+NN, CA, AS, GA, and TS are compared to demonstrate the performance of the hybrid algorithm. All the meta-heuristics were run 10 times independently using standard (adequate) parameters for all the datasets. The application has been implemented using 200-3000 standard iteration numbers for each dataset and increasing number of data (29-195). The current parameter setting has been also used for different studies in the literature [17, 23, 26, 39-41]. In the TS algorithm, the tabu length (L=30) is the optimal parameter. In GA, the crossover rate is 0.80, the mutation rate is 0.02. In GA, the crossover is one-point reversing type, and the mutation is the one-point operator . In CA, the dimension of space (dim=10), min. and max. temperature (T_{min}=0, T_{max} =100), initial endurance and supply (Init_End=1, Init_Supp =1), visibility threshold (Vis=0.5), dying rate (dye_rate=0) are the adequate parameters. In AS (Ant System), # of ants=20, alfa=1, beta=5, evaporation rate ($\boldsymbol{\rho = 0.7}$), Initial-Feremon=25. The population size is set to 100 for all the population-based meta-heuristics (GA, CA, and HA).

The distinct neighborhoods are used to produce new solutions in every iteration of the algorithm. The selected neighborhood operators generate new solutions [42, 43]. In this article, two operators; insert and swap_reverse are relatively diversified operators than swap and reverse operators. Thus, swap and reverse cause intensification in the TSP solution space. In this application, increasing iteration number is better because of using those operators and relatively large size of data [44]. Each operator is chosen respectively and applied once in each step. As previous papers indicate, multiple operators can search for feasible solutions [43, 45]. In summary, instead of using a single structure, it is expected to give near-optimal solutions when multiple structures are used. Then, the use of the best operator is defined by the minimum result of the combined operators using Eq. 7.

$$MNHs(x) = min \begin{pmatrix} swap(x), insert(x), reverse(x) \\ swap\_reverse(x) \end{pmatrix} \tag{7}$$

The hybrid algorithm (CA+NN) finds acceptable solutions in reasonable times using the multiple neighborhood technique that is highly plain, stable, and stark approach at 200-3000 iterations.

Table 1. Computational results of algorithms on the small and medium-scale TSP instances

TSP	*Measure*	*TS*	*GA*	*CA*	*AS*	*HA*
wi29	Best	27748.7	27603	27620.8	31844.9	28028.2
(27603)	Worst	30660.9	28189.8	31173.4	32812.1	29842.5
	Avg	29246.2	27807.3	28617.8	32098	28722.6
	Std.	1026.29	210.25	1074.28	410.63	627.8
	Time	0.39	16.73	14.71	16.07	3.33
	Number	1000	1000	1000	200	200
dj38	Best	7352.64	6659.43	6659.43	7896.11	6715.41
(6656)	Worst	8756.95	7711.26	8615.18	8234.19	6963.65
	Avg	8040.57	6859.56	7566.3	8031.35	6847.38
	Std.	461.66	357.2	615.77	174.58	81.75
	Time	0.28	21.16	17.26	26.91	2.91
	Number	1000	1000	1000	200	200
eil51	Best	492.28	472.49	486.69	469.12	456.78
(426)	Worst	592.64	522.5	565.71	502.02	491.15

	Avg	534.03	500.2	511.05	482.02	474.18
	Std.	27.37	17.11	22.71	11.16	11.71
	Time	0.23	42.46	18.1	42.89	3.24
	Number	1000	1000	1000	200	200
berlin52	Best	8582.17	8148.64	8689.26	8164.06	7819.25
(7542)	Worst	10422.7	9079.47	9939.23	8171.3	8120.48
	Avg	9387.1	8598.49	9375.6	8166.23	7923.54
	Std.	586.83	316.7	503.79	3.5	100.1
	Time	0.3	44.67	16.72	47.98	3.65
	Number	1000	1000	1000	200	200
st70	Best	838.07	975.16	961.18	777.83	761.32
(675)	Worst	1067.51	1082.02	1081.82	820.75	787.45
	Avg	971.7	1015.08	1001.55	793.18	778.84
	Std.	63.41	37.27	42.1	13.23	7.58
	Time	0.33	67.78	19.58	82.42	4.26
	Number	1000	1000	1000	200	200
eil76	Best	635.1	639.45	632.48	591.18	616.09
(538)	Worst	744.06	691.46	786.14	616.07	628.4
	Avg	689.02	658.69	709.98	606.14	622.37
	Std.	30.31	17.07	39.51	7.5	3.39
	Time	0.51	116.38	41.88	139.66	6.77
	Number	2000	2000	2000	300	300
pr76	Best	135971	120259	137215	125072	127090
(108159)	Worst	148044	135551	162441	135027	134121
	Avg	140374	127237	149411	130002	132463
	Std.	3664.91	3925.47	8172.79	3193.45	2194.7
	Time	0.49	81.18	42.61	135.78	6.93
	Number	2000	2000	2000	300	300
rat99	Best	1548.38	1400.99	1797.07	1382.35	1400.29
(1211)	Worst	1616.62	1507.61	2125.63	1420.08	1464.95
	Avg	1585.45	1452.62	1985.9	1387.81	1425.42
	Std.	25.69	33.86	128.71	12.52	21.59
	Time	0.67	131.43	46.76	235.11	7.71
	Number	2000	2000	2000	300	300
kroa100	Best	30145.5	26257.8	31587.8	25073.4	23842
(21282)	Worst	37467.5	32391.4	35904.8	26747.6	25096.7
	Avg	34131.5	29876.3	33535.3	26160.6	24402.2
	Std.	2258.21	2052.4	1782.45	508.83	428.86
	Time	0.9	218.1	74.93	397.39	12.12
	Number	3000	3000	3000	500	500

Table 1. continued

TSP	*Measure*	*TS*	*GA*	*CA*	*AS*	*HA*
eil101	Best	834.49	739.83	855.39	739.57	707.97
(629)	Worst	934.33	845.27	956.45	761.54	742.34
	Avg	888.78	802.91	900.57	751.94	722.76
	Std.	35.91	31.69	36.98	8.97	11
	Time	0.8	219.38	73.85	402.71	12.1
	Number	3000	3000	3000	500	500
bier127	Best	157508	154973	162329	129830	127783
(118282)	Worst	184032	178577	201660	130775	130247
	Avg	170867	166766	183272	130033	129155
	Std.	7432.93	7614.92	10385.6	264.18	677.55
	Time	0.92	304.9	80.94	670.66	15.28
	Number	3000	3000	3000	500	500
kroa150	Best	49136.7	47847.4	45872.6	31929.9	30460.7
(26524)	Worst	58658.5	56249.1	63081.3	32291.8	31561.6
	Avg	55770.3	51916.7	54840	32028.5	31038.8
	Std.	3089.75	2335.07	4790.61	131.05	309.95
	Time	1.01	416.89	93.49	912.53	15.43
	Number	3000	3000	3000	500	500
rat195	Best	3146.73	4135.58	4808.79	2691.89	2639.52
(2323)	Worst	3362.49	4709.22	5680.26	2809.67	2760.63
	Avg	3259.5	4358.68	5180.3	2729.68	2711.97
	Std.	68.47	208.62	312.98	43.41	40.49
	Time	1.66	549.39	110.85	1735.76	19.57
	Number	3000	3000	3000	500	500

Table 1 shows the experimental results and comparison between HA, AS, CA, GA, and TS. In this table, the results are given as best, worst, average solution, standard deviation, and CPU Time.

As inferred from Table 1, it can be observed that the quality of the hybrid algorithm (CA+NN) is better compared to AS, CA, GA, and TS for 70% of all datasets, excluding wi29, eil76, pr76, and rat99. Besides, in Table 2, HA finds 33, AS finds 12, GA finds 7, CA finds 1, and TS finds never acceptable solutions among 52 best results. In summary, Table 2 shows that the hybrid algorithm outperforms AS, CA, GA, and TS for 64% of all solutions.

HA (CA+NN) and AS have reasonable standard deviations among all the algorithms. A low standard deviation specifies that the hybrid algorithm is a more stable approach to find the acceptable results. Lastly, the hybrid algorithm solves the TSP problem in competitive times (8.72 secs.) in comparison to test algorithms for small and medium-scale instances.

In general, the experimental analysis shows that the hybrid algorithm is a robust and clear approach for solving the symmetric traveling salesman problems. This hybrid meta-heuristic would give better results and low standard deviations in short iteration numbers as compared to the other test algorithms.

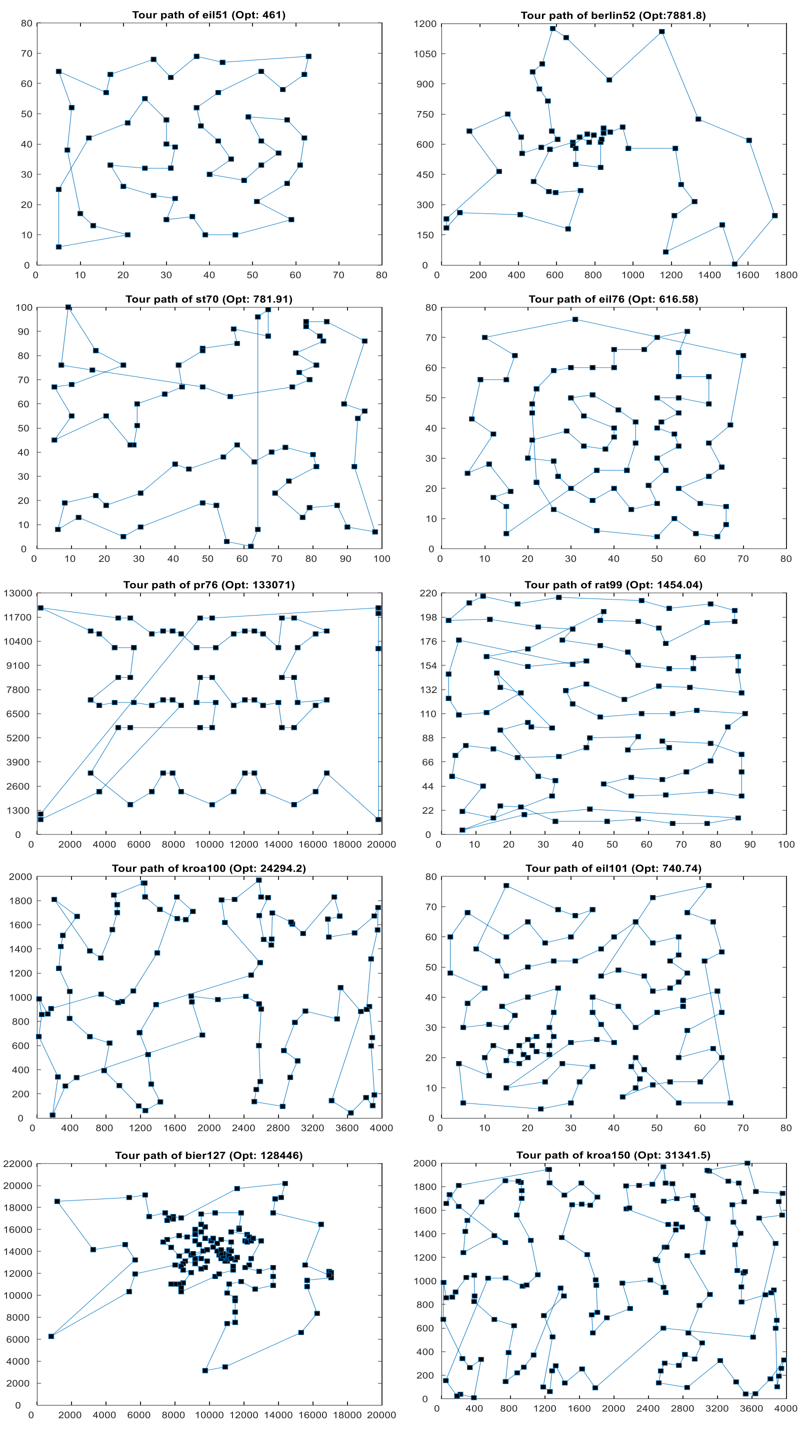

Figure 2. A set of solutions found by the hybrid algorithm (HA)

Figure 2 shows a set of solutions found by the hybrid algorithm on the small and medium-scale TSP instances.

4. Conclusions and Future Work

In recent decades, solving discrete problems via modern meta-heuristics is a popular research area. In this paper, the hybrid algorithm is implemented to the symmetric TSP instances. To evaluate the performance of the hybrid algorithm, it has been tested on 13 benchmark datasets. The computational results show that the hybrid algorithm can find better solutions compared to the ant system (AS), camel algorithm (CA), genetic algorithm (GA), and tabu search (TS) for 70% of all datasets and 64% of all solutions. As CPU time is considered, the hybrid algorithm is quite fast (8.72 secs.) to find the acceptable results.

In future studies, the hybrid algorithm can be further improved and combined with other meta-heuristics to optimize the effectiveness and efficiency of the algorithm. Furthermore, many comparative studies can be done in scheduling, assignment, timetabling, routing, and other combinatorial problems.

Table 2. The #of best-average-worst solutions and average CPU time

Algorithm	*Best*	*Worst*	*Average*	*Standard Deviation*	*Average CPU Time*
HA	8	10	9	6	8.72
AS	2	2	2	6	372.76
CA	(1)	0	0	0	50.13
GA	3	1	2	1	171.57
TS	0	0	0	0	0.65

Acknowledgment

The author thank the reviewers for their valuable comments and suggestions, which increased the clarity and scope of the article.

References

[1] Rajabioun, R. 2011. Cuckoo optimization algorithm, Applied Soft Computing, 11(8), 5508-5518

[2] Mian, T.A., Muhammad, U., Riaz, A. 2012. Jobs scheduling and worker assignment problem to minimize makespan using ant colony optimization metaheuristic, World Academy of Science, Engineering and Technology, 6(12), 2823-2826

[3] Tawhid, M.A., Savsani, P. 2019. Discrete Sine-Cosine Algorithm (DSCA) with Local Search for Solving Traveling Salesman Problem, Arabian Journal for Science and Engineering, 44(4), 3669-3679

[4] Teeninga, A., Volgenant, A. 2004. Improved Heuristics for the Traveling Purchaser Problem, Computers & Operations Research, 31, 139-150

[5] Soylu, B. 2015. A general variable neighborhood search heuristic for multiple traveling salesmen problem, Computers & Industrial Engineering, 90, 390–401

[6] Gendreau, M., Hertz, A., Laporte, G., Stan, M. 1998. A generalized insertion heuristic for the traveling salesman problem with time windows, Operations Research, 46(3), 330-335

[7] Zhang, R., Yun, W. Y., Kopfer, H. 2010. Heuristic-based truck scheduling for inland container transportation, OR Spectrum, 32, 787-808. DOI: 10. 1007/s00291-010-0193-4

[8] Xinchao, Z. 2011. Simulated annealing algorithm with adaptive neghborhood, Applied Soft Computing, 11(2), 1827-1836

[9] Misevicius, A. 2005. A tabu search algorithm for the quadratic assignment problem, Computational Optimization and Applications, 30, 95-111. DOI: 10. 1007/s10589-005-4562-x

[10] Laboudi, Z., Chikhi, S. 2012. Comparison of Genetic Algorithm and Quantum Genetic Algorithm, The International Arab Journal of Information Technology, 9(3), 243-249

[11] Chaharsooghi, S.K., Kermani, A.H.M. 2008. An effective ant colony optimization algorithm (ACO) for multi-objective resource allocation problem (MORAP), Applied Mathematics and Computation, 200(1), 167-177

[12] Woo, Z., Hoon, J., Loganathan, G.V. 2001. A New Heuristic Optimization Algorithm: Harmony Search,

Simulation,76(2), 60-68. DOI: 10.1177/0037549701 07600201

[13] Farmer, J.D., Packard, N.H., Perelson, A.S. 1986. The Immune System, Adaptation, and Machine Learning, Physica D, 22, 187-204

[14] Storn, R., Price, K. 1997. Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces, Journal of Global Optimization 11, 341–359. DOI: 10.1023/A:100820282 1328

[15] Yang, X-S. 2010. Nature-inspired metaheuristic algorithms, 2nd edition. Luniver Press, Frome.

[16] Yang, X-S. 2010. A New Meta-heuristic Bat Inspired Algorithm. pp 65-74. Gonzales, J.R., Pelta, D.A., Cruz, C., Terrazas, G., Krasnogor, G., ed. 2010. Nature Inspired Cooperative Strategies for Optimization (NICSO 2010), Springer, Berlin, Heidelberg, 397p.

[17] Hatamlou, A. 2018. Solving travelling salesman problem using black hole algorithm, Soft Computing, 22(24), 8167-8175. DOI: 10.1007/s00500-017-2760-y

[18] Wang, Z., Geng, X., Shao, Z. 2009. An effective simulated annealing algorithm for solving the travelling salesman problem, Journal of Computational and Theoretical Nanoscience, 6(7), 1680-1686

[19] Khanra, A., Maiti, M.K., Maiti, M. 2015. Profit maximization of tsp through a hybrid algorithm, Computers & Industrial Engineering, 88, 229-236. DOI: 10.1016/j.cie.2015.06.018

[20] Sahana, S.K. 2019. Hybrid optimizer for the travelling salesman problem, Evolutionary Intelligence, 12, 179-188. DOI: 10.1007/s12065-019-00208-7

[21] Ouaaraba, A., Ahioda, B., Yangb, X.-S. 2014. Discrete cuckoo search algorithm for the traveling salesman problem, Neural Computing and Applications, 24(7-8), 1659-1669. DOI: 10.1007/s00521-013-1402-2

[22] Chen, S.-M., Chien, C.-Y. 2011. Solving the traveling salesman problem based on the genetic simulated annealing ant colony system with particle swarm optimization techniques, Expert Systems with Applications 38(12), 14439-14450

[23] Ibrahim, M.K., Ali, R.S. 2016. Novel optimization algorithm inspired by camel traveling behavior, Iraqi Journal for Electrical and Electronic Engineering, 12(2), 167-177

[24] Ali, R.S., Alnahwi, F.M., Abdullah, A.S. 2019. A modified camel travelling behavior algorithm for engineering applications, Australian Journal of Electrical and Electronics Engineering, 16(3),176-186. DOI: 10.1080/1448837X.2-019.1640010

[25] Hassan, K.H, Abdulmuttalib, T.R., Jasim, B.H. 2021. Parameters estimation of solar photovoltaic module using camel behavior search algorithm, International Journal of Electrical and Computer Engineering (IJECE), 11(1), 788-793

[26] Demiral, M. F. 2021. An application of a modified camel traveling behavior algorithm for traveling salesman problem, Journal of Scientific Reports-A, 047, 88-98. https://dergipark.org.tr/en/pub/jsr-a/issue/6- 7627/901408

[27] Alobaidi, A.T., Abdullah, H.S., Ahmed, Z.O. 2017. Camel Herds Algorithm: a New Swarm Intelligent Algorithm to Solve Optimization Problems, International Journal on Perceptive and Cognitive Computing (IJPCC), 3(1), 6-10

[28] Ahmed, Z. O., Sadiq, A.T., Abdullah, H.S. 2019. Solving the Traveling Salesman's Problem Using Camels Herd Algorithm, 2nd Scientific Conference of Computer Sciences (SCCS), University of Technology-Iraq, 1-5

[29] Fagerholt, K., Christiansen, M. 2000. A traveling salesman problem with allocation, time window, and precedence constraints - an application to ship scheduling, International Transactions in Operational Research, 7, 231-244

[30] Nagata, Y., Soler, D. 2012. A new genetic algorithm for the asymmetric traveling salesman problem, Expert Systems with Applications, 39, 8947-8953

[31] Yildirim, A.E., Karci, A. 2018. Applications of artificial atom algorithm to small-scale traveling salesman problems, Soft Computing, 22(22):7619-7631. DOI: 10.1007/s00500-017-2735-z

[32] Petersen, H. L., Archetti, C., Sprenza, M. G. 2010. Exact solutions to the double travelling salesman problem with multiple stacks, Networks, 56(4), 229-243

[33] Yuan, S., Skinner, B., Huang, S., Liu, D. 2013. A new crossover approach for solving the multiple travelling salesmen problem using genetic algorithms, European Journal of Operational Research, 228, 72–82

[34] Montemanni, R., Mojana, M., Di Caro, G., Gambardella, L.M. 2013. A decomposition-based exact approach for the sequential ordering problem, Journal of Applied Operational Research, 5(1), 2-13

[35] Mansini, R., Tocchella, B. 2009. The Traveling Purchaser Problem with Budget Constraint, Computers & Operations Research, 36, p.2263-2274

[36] Chandra, A., Naro, A. 2020. A Comparative Study of Capacitated Vehicle Routing Problem Heuristic Model, International Journal of Engineering and Emerging Technology, 5(2), 94-100

[37] van Ee, M., Sitters, R. 2018. The A Priori Traveling Repairman Problem, Algorithmica, 80, 2818–2833. DOI: 10.1007/s00453-017-0351-z

[38] Burkard, R.E., Deineko, V.G., van Dal, R., van, J.A.A., Veen, der, Woeginger, G.J. 1998. Well-solvable special cases of the traveling salesman problem: a

survey, Society for Industrial and Applied Mathematics, 40(3), 496-546. DOI: 10.1137/S0036144596297514

[39] Wei, X. 2014. Parameters Analysis for Basic Ant Colony Optimization Algorithm in TSP, International Journal of u-and e-Service, Science and Technology (IJUNESST), 7(4), 159-170.

[40] Shweta, K., Singh, A. 2013. An Effect and Analysis of Parameter on Ant Colony Optimization for Solving Travelling Salesman Problem, International Journal of Computer Science and Mobile Computing (IJCSMC), 2(11), 222-229.

[41] Lin, W.-Y., Lee, W.-Y., Hong, T.-P. 2003. Adapting Crossover and Mutation Rates in Genetic Algorithms, Journal of Information Science and Engineering, 19(5), 889-903.

[42] Halim, A.H. and Ismail, I. 2019. Combinatorial optimization: comparison of heuristic algorithms in travelling salesman problem, Archives of Computational Methods in Engineering, 26, 367–380. DOI: 10.1007/s11831-017-9247-y

[43] Szeto, W.Y., Yongzhong, W., Ho, S.C. 2011. An artificial bee colony algorithm for the capacitated vehicle routing problem, European Journal of Operational Research, 215(1), 126-135. DOI: 10.1016/j.ejor.2011.06.006

[44] Feng, X., Liu, Y., Yu, H., Luo, F. 2019. Physarum-energy optimization algorithm, Soft Computing, 23, 871–888. DOI: 10.1007/s00500-017-2796-z.

[45] Demiral, M.F., Işik, A.H. 2020. Simulated annealing algorithm for a medium-sized tsp data. pp. 457-465. Hemanth, D. J., Kose, U., ed. 2020, Artificial Intelligence and Applied Mathematics in Engineering Problems, Springer, Cham, 1081p. DOI: 10.1007/978-3-030-36178-5_35