Periodicals of Engineering and Natural Sciences ISSN 2303-4521
Vol. 8, No. 3, July 2020, pp.1331-1344

Solving competitive traveling salesman problem using Gray Wolf Optimization Algorithm

Mushreq Luay Taha[1], Belal Al-Khateeb [2], Yasser F. Hassan [3], Ossama M.M. Ismail[4], Ossama Abu Rawash[5]

[1] Alexandrea University – Factually of Sciences
[2,3,5]University of Anbar – College of Computer Science and Information Systems
[4] Arabic Academy for Sciences and Technology and Maritime Transport

ABSTRACT

In this paper a Gray Wolf Optimization (GWO) algorithm is presented to solve the Competitive Traveling Salesman Problem (CTSP). In CTSP, there are numbers of non-cooperative salesmen their goal is visiting a larger possible number of cities with lowest cost and most gained benefit. Each salesman will get a benefit when he visits unvisited city before all other salesmen. Two approaches have been used in this paper, the first one called static approach, it is mean evenly divides the cities among salesmen. The second approach is called parallel at which all cities are available to all salesmen and each salesman tries to visit as much as possible of the unvisited cities. The algorithms are executed for 1000 times and the results prove that the GWO is very efficient giving an indication of the superiority of GWO in solving CTSP.

Keywords: Gray Wolf Optimization, CTSP, TSP, Random Neighbour

Corresponding Author:

Mushreq Luay Taha
Computer Science
Alexandrea University
Egypt, Alexandrea
E-mail: mushrik8888@gmail.com

1. Introduction

The Traveling Salesman Problem (TSP) is an important classical problem, it is (NP-hard) problem at which there is no polynomial time to solve it and find exact solution **[1, 2]** .Researchers used many algorithms in order to find a best solution. TSP idea is finding the shortest tour for salesman that must visit a number of cities exactly once and then return to the start city (ie, a Hamiltonian cycle) and between each city there is a distance represents the cost of moving form node or city to another one[3, 4]**.** TSP is one of the most studied combinatorial optimization problems that have many applications such as routing, operation research, scheduling and transportation routing. TSP has some of types such as Multi Traveling Salesman Problem (mTSP) and Competitive Traveling Salesman Problem (CTSP), asymmetric TSP, clustered TSP [5, 6]. CTSP is proposed by Fekete in (2004), is a complex non-cooperative problem where non-cooperative means that every salesman works to his advantage competitively with other salesmen as each salesman looking for increasing profits by trying to travel to the largest possible number of cities that have not been visited before by knowing the paths used by the rest of the salesmen[7]. The first salesman who arrives at a city will got the benefit and the others will get nothing and they loss the distance cost and if more than one salesmen arrives at same time they will share the benefit so that the salesmen aim to obtain highest benefit by visiting highest number of cities[8-11].

2. Competitive traveling salesman problem model

The interaction behavior and the dynamicity of CTSP, there are a number of agents Let M= {1……m} can be considered as set related to non-cooperative agents. Those agents must visit (n) cites therefor N= {1….n} of cites.

- Benefit (Bi): with regard to each one of the cities i ∈ {1,….n }, also there have been continuous benefits for each city.

The first agent arriving at a city will obtains the benefit and the other receives zero if they arrived after him and if two or more agents visit the same city at same time, they will share the benefits.

- Cost: when agents travel from city ***i*** to city ***j*** they have to pay a cost C_{ij}.

Payoff: it is calculated for each agent by compute the rewards of visiting unvisited sites and calculate the cost of traveling and it is computed as:

$$u = \sum_{j=1}^{k0} B_j - \sum_{i=1}^{k-1} C_{\,i(i+1)} - C_{k1} \qquad (1)$$

$\sum_{k=1}^{k0} \boldsymbol{B_j}$ refers to the aggregation reward from k_0 ($k_0 < k$) cites

- **Path:** Initially, the salesmen are positioned in various cities. They should be travelling between cities and they might not be changing the destination as soon as starting the trip. For completing the travel, they must be returning to their departure city.
- **Trip Speed:** all salesmen have constant trip's speed V_K.
- **Common Knowledge:** there are a set of knowledge contains the speed, cost of this direction, path traveled, rewards and costs.

The aim of all agents is maximizing the reward and decrease the cost and each agent choses his tour independently [4][12].

3. Related works

In 2004, Fekete et.al introduced the Competing Salesmen Problem (CSP) as a new variant of traveling salesman problem. They suggested that multiple salesmen compete against each other to visit the largest number of nodes instead of cooperating in finding the shortest route. At any given time, all salesmen know their opponents' positions. A salesman wins when he visits more cities than his competitors[13].

In 2013, Kendall and Li have offered new type related to TSP, where a few salesmen are planning on visiting a few cities, also the relation between then have been non-cooperative. The salesman gets award if he visits a city that has not been visited by other salesmen. They have to pay the cost of traveling to a city. The aim of each salesman is to visit the largest number of cities that have not been visited before with minimum travel traveling distance. All salesman must be predicting the competitor's tours in the case when they plan their tour. The researchers provided hyper-heuristic algorithm for the purpose of solving such problem, such includes low-level heuristic could be utilized for building tour, also high-level heuristics utilized for choosing among low-level heuristic at each one of the decision points. The simulation results of the study indicated that the suggested algorithms achieved good results in computing CTSP's approximate solutions, also the algorithm has the ability for inheriting excellent features related to low-level heuristic [14].

Mohtadi and Nogondarian in 2014 presented a game theoretic approach to solve TSP in the competitive situations. Furthermore, they used game theory as a mathematical model to test the problems. They used genetic and tabu search algorithms. Experimental results showing that the computational error has been in in sensible range. Furthermore, Tabu search algorithm created certain solutions with optimum quality, also not much time has been required [4][15, 16]**.**

In 2015, Li and Kendall presented hyper-heuristic approach for generating games' adaptive strategies. On the basis of low-level heuristic, the hyper-heuristic game player has the ability of generating the strategies that are adapting to co-player's behavior as well as dynamics of the game, also introduced hyper-heuristic game player for three games iterated prisoner dilemma (IPD), CTSP, as well as the repeated Go of spiel. The study

utilized RN+2opt, RN, NN+2opt, as well as NN. High-level algorithm has been identifying heuristics that have been used through the other agents and after that selecting from the low-level heuristics. Furthermore, the study's computational results indicated that the hyper-heuristic game player outperforming the low-level heuristic in the dynamic as well as repeated games and also the hyper-heuristic game player generating adaptive approaches in the case when low-level heuristics have been deterministic. The straightforward heuristic selection approaches could be utilized for construing automated game players in various games[17, 18].

In 2016 Mohannad Abdul-Sattar and Belal Al-Khateeb applied ant colony optimization algorithm for solve CTSP. They used two different approaches to solve the CTSP, the first approach divided the cities among salesmen in order to approve the ability of ACO algorithm in solving the CTSP and then compared the obtained results with Nearest Neighbors (NN) and Random Neighbors (RN) algorithms, the obtained results prove that ACO algorithm was better than the other strategies. The second approach uses unspecified number of cities as each salesman will try to visit as much as possible cities according to the salesman's strategy. In this approach, two directions are taken, the first one uses the same plan for each salesman, while the other uses an update plan, for each salesman, ACO algorithm results were better than another algorithm (NN & RN & ACS algorithm with the first approach) [3].

4. Difficulty of CTSP

In CTSP the salesmen behavior is not corporative each salesman dose not known the other salesmen plan and each salesman wont to obtain maximum benefit by visiting the largest number of unvisited nodes so that we need efficient tools to find the optimal track that achieves the largest revenues rapidly, traditional approach has been extremely slow to find the exact solutions so that we proposed using heuristic approach to find approximate salutations, this approach is simple and easy: because it is inspired from very simple biological behavior and flexible because it can be applied without change in its structure, also it is a derivation-free mechanisms and can avoid local optima because of its stochastic nature. One of the heuristics approaches is swarm intelligence, it is term that describes the collective behavior of autonomic decentralized systems, it deals with the simulation of the behavior of less intelligent pings with limited possibilities such as ant, birds and fish, which at a same time exhibit a highly intelligent social behavior. It preserves information that is related to search space throughout iteration, sometimes utilizing memory for saving optimum solution and typically having not much parameters to alter[19-22]. GWO is one of swarm intelligence algorithms that will used in this paper to solve the competitive travelling salesmen problem.

5. GWO algorithm

This algorithm simulates the hierarchy as well as the mechanism of hunting of the gray wolves which have been suggested via Ali Mirjalili in 2014. There have been four gray wolves' types (alpha, beta, delta as well as omega) simulating the steps to hung, encircle as well as attack. The Figure 1 shows the strict social hierarchy of gray wolves

.

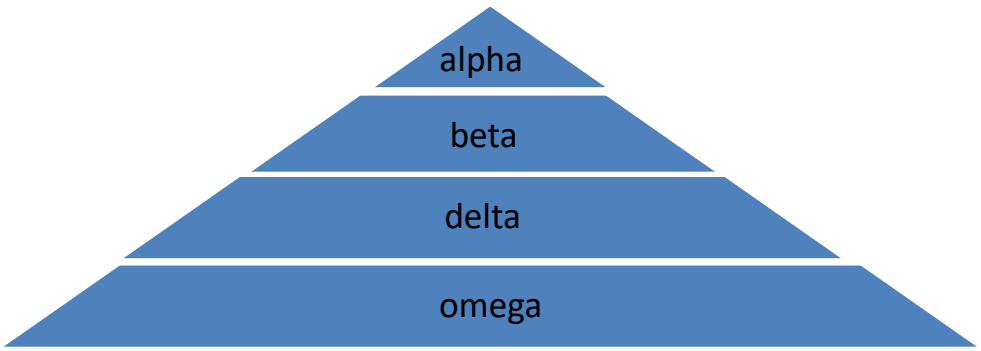

Figure 1. The hierarchy related to The Gray Wolves Groups

The alpha represents the optimal solution and the second is beta then delta and omega at last. Grey wolves mostly search according to the position of the alpha, beta, and delta. They diverge from each other to search for prey and converge to attack prey. The search agent may be in any location between the current location and the prey's location and the pseudo code of GWO in figure 2 explain how the algorithm solve the optimization problems. [23-26].

- Initialize the population of the gray wolf X_i (I =1,2,…..n)
- Initialize a, A,C
- Calculate every search agent's fitness
- $X_\alpha$represents the optimal search agent
- X_β represents the second optimal search agent
- $X_\delta$represents the third optimal search agent

While (t is smaller than the maximum number of iterations)

For every one of the search agents

- Update the current search agent's positions according to the equation 8.2
- End For
- Update a,A,C
- Calculate the fitness values for each search agent
- Update $X_\alpha, X_\beta \& X_\delta$
- i=i+1
- End While
- Output X_α

Figure 2. GWO pseudo code

6. Problem representation

CTSP can be represented as a graph, where nodes represent the cites that each salesman will visit and the arrows represent the ways that each salesman used to move from one city to another and the length of them represents the distances or cost.

7. GWO solution generation

The construction of solution consists of some stages that are started with determining the time, restrictions and predefined relationship and then creates initial population that represented as the pack of wolves that spread among nodes; this called initial system case for solution[27, 28].

After solution construction, the system begins to work through movement of wolves from node (city) to another in order to build solutions and in hope to find the best solution, the memory saves three optimal solutions and update it when find better solutions.Building solutions process will continue until finding the best possible solution or achieve stop condition; the gray wolf optimization is a dynamic algorithm because it avoids local solutions by the ability to changing locations within the hierarchy such as the alpha may be converting to delta if he be very old and beta candidate to be alpha.

8. Proposed Gray Wolf System (GWOS)

Two approaches to solve competitive traveling salesman problem are taken. The first one is static and the second one is dynamic, the static means dividing cites among salesmen or agents equally and the dynamic means no dividing of nodes (cities) and each agent tries to visit largest numbers of cites as possible according his strategy.

8.1 Initialization

It is an important stage in solution construction. It provides data and other requirements and then submits it in acceptable form to algorithm to be able to start its function.

8. 2 Preparing

It consists of multi sages; the first stage is reading the problem dataset information. In the dataset, each city has two points, one of them on X-axis and the second on the Y-axis. The distance between each city and other cites calculates by the following equation:

$$\text{Dis (i,j)} = \sqrt{(X_i - X_j)^2 + (Y_i - Y_j)^2} \quad (2)$$

X_i and y_i represents the city (I) points on x-axis and y-axis so X_j and Y_j represent the city (J) points on x-axis and y-axis, this equation will used to compute the distances between all cites. The result of this step is two dimensional array that contains values that represent the distances between cites, in this step the salesmen will know the number of cities that they must be visit it.

After collecting knowledge about the problem such as number of cities and distances of it, the wolves pack distributes randomly, this step represents the population.

The population is firstly initialized randomly and then uses the equation below to determine the destination.

$$\vec{D} = |\vec{C}.X_p(t) - \vec{x}(t)| \quad (3)$$

Where: X_p Represents the prey's position vector. A, C represent the vector of the coefficient. X represents the gray wolves' position vector.

$$\vec{A} = 2\vec{a}.\vec{r}_1 - \vec{a} \quad (4)$$

$$\vec{C} = 2\vec{r}_2 \quad (5)$$

$$\vec{a} = 2\left(1 - \frac{it}{N}\right) \quad (6)$$

Where the component of $\vec{a}$ linearly decrease from 2 to 0 throughout the iterations, it represents the initialized value and N is maximum iterations number, and $\vec{r1}, \vec{r2}$ random vectors [0-1].[29, 30].

Because of there are three candidate solutions in this algorithm we must use equation for the best three types so that we shall use the equations below to create population:

$$\vec{D}_\alpha = |\vec{C_1}.\vec{X}\alpha - \vec{X}|, \vec{D}_\beta = |\vec{C}_2.\vec{X}_\beta - \vec{X}|, \vec{D}_\delta = |\vec{C}_3.\vec{X}_\delta - \vec{X}| \quad (7)$$

$$\vec{X}_1 = \vec{X}_\alpha - \vec{A}_1.(\vec{D}_\alpha), \vec{X}_2 = \vec{X}_\beta - \vec{A}_2.(\vec{D}_\beta), \vec{X}_3 = \vec{X}_\delta - \vec{A}_3.(\vec{D}_\delta) \quad (8)$$

$$(\vec{X}_{t+1}) = \frac{\vec{X}_1 + \vec{X}_2 + \vec{X}_3}{3} \quad (9)$$

The wolves change their position in accordance with the alpha and the prey's position until achieves the optimal tour or number of iterations condition or exploring for another prey better than the last.
After initializing and preparing the number of cities for each salesman, then the cities will be divided equally among salesmen and number of wolves for each salesman [26].

8.3 Solution construction by GWO

After initialization and preparing stages, the algorithm starts to move from the start node to other nodes that determined for each salesman in initialization stage.
The wolves move from node to another until amount to prey (the goal node) and then return to start node. To create a path from start node to goal node wolves will use the equation below in order to determine the next node in particular node

$$\vec{X}_{(t+1)} = \begin{cases} |\overrightarrow{C}.X_P(t) - \overrightarrow{A}.\overrightarrow{D} \text{ if } \overrightarrow{A} < 1 \text{ explotation} \\ \text{else} \qquad\qquad \overrightarrow{A} \geq 1 \text{ exploraton} \end{cases} \quad (10)$$

Where $\vec{X}_{(t+1)}$ is the next position or updating position for each wolf after use the equation (7) and $\overrightarrow{C}$ a coefficient vector contain random values simulate the impact of the obstacles to the approaching preys in the nature calculated by equation (5), and $\overrightarrow{A}$ represent as a coefficient vectors it is also random value and calculated by equation (4) and $\overrightarrow{D}$ is the direction of prey finally X_P the position of prey. After determining the city to be traveled and the salesman the wolves' locations are updated based on current locations and using the equation (7). The using of random values with $\overrightarrow{A}$ greater than 1 or less than 1 to compel wolves of diverging from the prey to emphases explorations and allow search globally and diverge from prey for the sake of finding a better prey [27][28][29].

8.4 Position updating

After searching and finding the best location by alpha and determining the next city, salesman traveling to it and all other wolves changes its direction based on the alpha wolf. Algorithm1 shows the proposed static GWO method, where (L) represents the number of unvisited cities that determined for each salesman and (S) represent the visited city of it. Algorithm 1 shows the proposed static GWO method, where (L) represents the number of unvisited cities that determined for each salesman and (S) represent the visited city of it.

Algorithm 1: Static GWOS.

INPUT: get the dataset information

1. Compute the distances between cities according to the equation (2).
2. Set the number of salesmen.
3. Setting the number of cities for every salesman.
4. Setting the number of wolves for each salesman.
5. Setting the start city for each salesman.

Solution

6. Find the tour for each salesman (L)
   For i = 1 to the number of the salesmen
      For j = 1 to the number of the cities
         For k =1 to the number of the wolves
            Find the next city (S) by equation (8)

The best agent is alpha
The 2nd best agent is beta
The 3rd best agent is delta
End for

L=L-S
Update city (S) by equation (9)
Tour =S
Return to start city
End for
End for
Find the best tour by equation (1) for each salesman
End
OUTPUT: the best tour.

Algorithm 2 shows the proposed parallel GWO method, where (L) represents the number of unvisited cities that determined for each salesman and (S) represent the visited city of it.

Algorithm 2: Parallel GWOS.

1. Compute the distances between cities according to the equation (2).
2. Set the number of salesmen.
3. Set the number of cities
4. Set the number of wolves for each salesman.
Solution
5. Find the start city for each salesman
6. Find the first move for each salesman
For i= 0 to number of salesman
Find the next cities according to equation (8)
Calculating the fitness of every one of the agents

Xα represents the optimal search agent
Xβ represents the second optimal agent
Xδ represents the third optimal agent
Update position of the current search agent by equation (9)
7. Find the tour for each salesman
Number of city = number of city -1
While (number of city >0)
Identify which salesman begin first according equation (2)
Find the next city by equation (8)

Update position by equation (9)
Number of cites = number of cites -1
End while
Return to start city
For i= 1 to the number of the salesmen
For j=1 to the number of the cities
Update position by equation (9)
End for
End for
OUTPUT: best tour for each salesman

9. Experimental setup and results

This section presents the experimental results of the proposed system. Several tests are applied to evaluate the performance of the proposed system.

9.1 General system implementation

The proposed system will apply set of tests on a dataset that is presented by Li and Kendal with 20 cities, the system contains two parts

- Static Gray Wolf Optimization.
- Parallel Gray Wolf Optimization.

The first step to test gray wolf optimization is load cities from dataset, then the system will compute distances between all cities using equation (2). The next step is to specify start city for each salesman then the system will generate path for each salesman by using algorithm 1**.**

The last step is to find the best path with maximum sales by using equation (1), in this step, the system gets the information for each salesman then compute fitness function between summation of path and summation of sales, the maximum value that means this salesman has the best path and best sales. Table (1) shows the distances between dataset cities.

Table 1. Distances between cities

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0	97	55	21	49	70	63	33	61	85	78.4	21	71	43	79	27.9	54.6	48.2	94.9	79.25
2	97	0	53	100	62	29	35	95	56	32.3	40.1	80	78	58	75	71.8	48.8	73.7	30.2	58.25
3	55	53	0	67	51	38	21	42	57	31.2	59.5	34	80.4	14	36	41.4	41.7	61.7	41.2	27.66
4	21	100	67	0	42	72	70	54	54	94.9	74.1	37	57.1	54	96	29.1	53	37	104	94.01
5	49	62	51	42	0	34	41	67	12	67.5	31.6	44	29.7	40	87	22.1	14.1	12	75	77.82
6	70	29	38	72	34	0	17	74	30	38	22.7	55	54.6	36	71	43.8	20.2	45.9	43.4	56.73
7	63	35	21	70	41	17	0	60	42	27	39.7	45	66.6	23	54	40.9	27.5	52.5	35.4	40.36
8	33	95	42	54	67	74	60	0	78	72.5	90	24	95.3	38	52	46.1	66.4	72	82.2	57.45
9	61	56	57	54	12	30	42	78	0	67.1	20.5	55	25.1	47	93	34	14.9	19.2	73.3	81.27
10	85	32	31	95	67	38	27	72	67	0	59.6	65	92	42	43	66.5	53.9	79.4	10	26.17
11	78	40	60	74	32	23	40	90	21	59.6	0	68	38.2	54	93	50.6	24.3	39.7	63.5	79.4
12	21	80	34	37	44	55	45	24	55	64.7	68.3	0	71.6	23	60	22.4	44	48.4	74.6	58.55
13	71	78	80	57	30	55	67	95	25	92	38.2	72	0	69	117	49.2	39.4	23.4	98	106
14	43	58	14	54	40	36	23	38	47	42.5	54.1	23	69.3	0	48	27.5	32.6	49.1	52.4	41.59
15	79	75	36	96	87	71	54	52	93	43.2	93.5	60	117	48	0	74.8	77.8	97.4	49.2	17.03
16	28	72	41	29	22	44	41	46	34	66.5	50.6	22	49.2	27	75	0	26.8	26.1	75.7	69.05
17	55	49	42	53	14	20	28	66	15	53.9	24.3	44	39.4	33	78	26.8	0	26.1	61.1	66.57
18	48	74	62	37	12	46	52	72	19	79.4	39.7	48	23.4	49	97	26.1	26.1	0	87	88.77
19	95	30	41	104	75	43	35	82	73	10	63.5	75	98	52	49	75.7	61.1	87	0	32.39
20	79	58	28	94	78	57	40	57	81	26.2	79.4	59	106	42	17	69.1	66.6	88.8	32.4	0

The first row and first column in this table represent the number of cities (nodes), the other cells represent distances between these nodes depending on the location of node using x-axes and y-axes.

9.2 Experimental Results: This section shows all the obtained results for static and parallel gray wolf optimization methods that are proposed to solve the CTSP problem.

9.2.1 Number of Wolves: In order to determine the most number of wolves to be used in the experiments, different number of wolves are tested. The obtained results with 1000 iterations indicate that the best number of wolves is 4-7 wolves for 10 cities and 3-5 for 20 cities as shown in the table (2) and Figure (3).

Table 2. The effects of the numbers of wolves in GWOS

Number of Cities for Each Salesman	Number of Salesmen	Numbers of Wolves	Best Track	Cost	Benefit
10	2	3	1	422.6184	1025
10	2	4	2	541.94	1175
10	2	5	2	541.94	1175
10	2	6	2	541.94	1175
10	2	7	2	573.2	1250
10	2	8	2	541.94	1000
10	2	9	2	474.395	1000
10	2	10	1	422.618	1050
10	2	11	1	422.618	1050
10	2	12	1	422.618	1050
20	2	3	2	850.337	2100
20	2	4	2	909.85	2175
20	2	5	2	909.85	2175
20	2	6	1	729.342	1825
20	2	7	1	729.342	2025
20	2	8	1	729.342	2025
20	2	9	1	729.342	1825
20	2	10	1	729.342	2025
20	2	11	1	729.342	2025
20	2	12	1	729.342	1950

Form table 2 that above various numbers of cities and salesman we can denote the best results obtained when the number of wolves (4-7) wolf for various number of cities and the Figure 3 show the carve of benefits based on number of wolves.

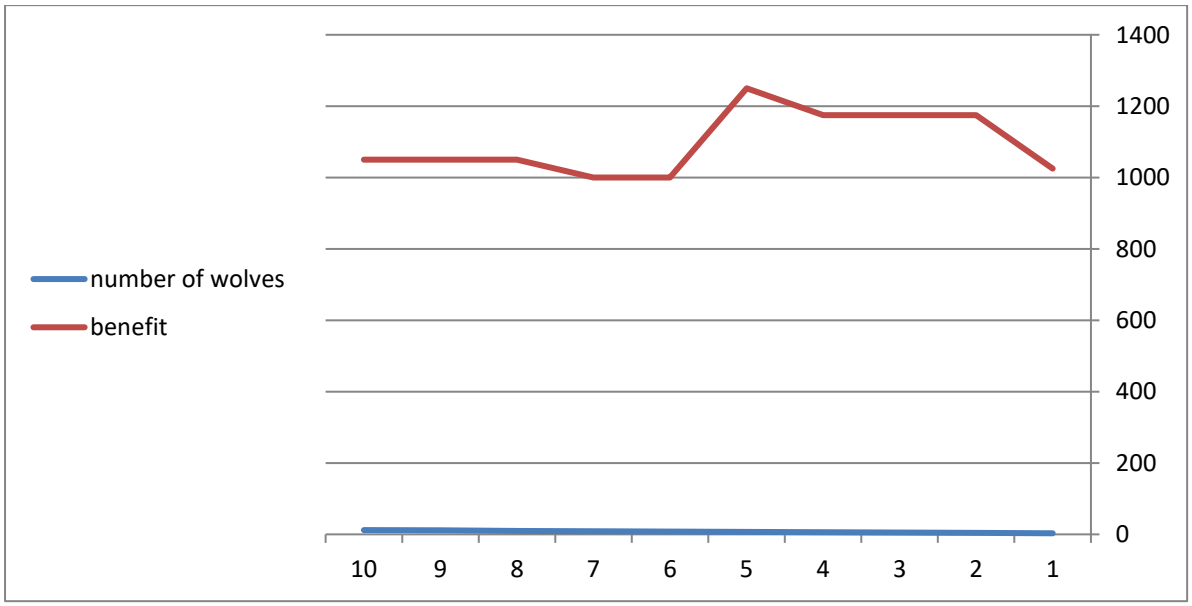

Figure 3. The number of wolves benefits (2 Salesmen and 10 Cities)

The orange carve represents benefit and the blue for number of wolves, from this Figure we can denote the proposed system achieve the maximum benefit when the number of wolves (4-7).

From all above it is possible to conclude that the use of a large number of wolves (more than 7 wolves) with a few cities leads to negatively affects the exploration and exploitation properties of the algorithm and adopt local solutions therefore, we notice an increase in total costs when increasing the number of wolves and a decrease in total benefits.

The number of wolves is an important factor to obtaining the optimal possible solution and most influences in problems which the number of cities is large.

9.2.2 Static Gray Wolf Optimization System (GWOS)

In this method, each salesman has the same number of cites at which each salesman has 10 cities from the 20 city CTSP, where the benefit is set to 150 for each city. The obtained results are then compared with Ant Colony Optimization, Nearest Neighbors (NN) and Random Neighbors (RN).

The Nearest Neighbor Algorithm (NN) Examines all the cities that have not been visited before and chooses the nearest one.

The Random Neighbor Algorithm (RN) Randomly selects the next city from the list of unvisited cities.

The results that are shown in tables (3), (4) and (5) prove the performance of GWOS as it has outperformed the other selected algorithms in best and average results for all tests, which proves the ability of GWOS in solving CTSP in static situation.

Table 3. Best results of two salesmen with 20 static cites and 1000 iteration

Algorithms		**Benefit**	
		Salesman 1	Salesman 2
NN	NN	869.82	945.99
NN	RN	869.82	696.65
NN	ACS	869.82	1034.14
RN	NN	805.77	945.99
RN	RN	805.77	696.65
RN	ACS	805.77	1034.14
ACS	NN	1029.47	945.99
ACS	RN	1029.47	757.74
ACS	ACS	1029.47	1034.14
GWOS	NN	1175	945.68
GWOS	RN	1000	757.74
GWOS	ACS	1175	1034.14
GWOS	GWOS	1575	1050
NN	GWOS	869.82	1050
RN	GWOS	805.77	900
ACS	GWOS	1029	1175

Table 3 shows the results of using GWOS and compare it with other algorithms that uses same dataset. GWOS outperformed the rest of other algorithms, this reflect the efficiency of GWOS to solve CTSP.

Table 4. Best results of 1000 executions for two salesmen and static 20 cities

Algorithm	Best benefit
Nearest Neighbor (NN)	945.99
Random Neighbor (RN)	869.82
Ant Colony System (ACS)	1043
Gray Wolf Optimization System (GWOS)	1575

Table 4 above show the best results for each algorithms that uses same dataset, the results show GWOS achieve higher benefit than other algorithms that uses the same dataset and then ACS in second level the NN and RN.

Table 5. Average Results of 1000 Executions for Two Salesmen with 20 Static Cities

Algorithm	Average of benefit
Nearest Neighbor (NN)	945.99
Random Neighbor (RN)	869.82
Ant Colony System (ACS)	1043
Gray Wolf Optimization (GWOS)	1052.885

In table 5 the average of benefits that obtained when applying algorithms on the same dataset. It is show the GWOS exceed on other algorithms and obtain higher benefits than others.

9.2.3 Parallel Gray Wolf Optimization System (GWOS)

In this approach, all cities are available for all salesmen. Each salesman will choose the next city form the available cities list. The salesman who arrives firstly will get all benefit and the others will lose the cost of travel therefore the salesmen must avoid visiting the visited cities, if more than one salesman coincides in same city at same moment the benefit will divided among them. The results of GWOS are compared with RN, NN and ACS algorithms. Nine salesmen and 20 parallel CTSP will uses and the compares with the results of the other algorithms. The results are shown in tables (6) and (7).

Table 6. Best Results of 1000 Executions for Nine Salesmen for 20 Parallel Cities

Algorithm	Benefit
Nearest Neighbor (NN)	1634.28
Random neighbor (RN)	1152
Ant Colony System(ACS)	1826
Gray Wolf Optimization System (GWOS)	2175

Table 7. Average Results of 1000 Executions for Nine Salesmen with 20 Parallel Cities

Algorithm	Average of Benefit
Nearest Neighbor (NN)	1569.43
Random neighbor (RN)	1113.62
Ant Colony System(ACS)	1560.42
Gray Wolf Optimization System (GWOS)	1951.633

The results in tables (6) and (7) reflect the efficiency and superiority of GWOS. Those results prove the ability and success of GWOS in solving CTSP over than NN, RN and ACO in parallel situation. Table (6) shows the best results that obtained when applying multi algorithms that used same dataset. The results show GWOS achieved the best benefit.
Table (7) shows the average of obtained benefit for each algorithm, GWOS achieved the highest benefit than other algorithms. This clearly reflects the superiority of GWOS.

9.3 Statistical Results of GWOS and the Selected Algorithms

To verify the obtained results in tables (3) and (6). T-test applied for two samples in order to check the efficiency (GWOS) with the highest payoff compared with other algorithms. The T-value will be compared with T-critical value, if the t-value is smaller than the t-critical value, the null hypothesis will be rejected. Alpha level assumed (0.05) it is the stander value of alpha for these tests. Assume the null hypotheses is there are no statistical differences between the (GWOS) and other algorithms. The results shown in tables (8) and (9).

Table 8. T-test for GWOS and Other Selected Algorithms in Static System

t-Test: Two-Sample Assuming Equal Variances		
	GWOS	*OTHERS*
Mean	1116.667	912.52
Variance	10208.33	19923.9292
Observations	3	3
Pooled Variance	15066.13	
Hypothesized Mean Difference	0	
Df	4	
t Stat	2.036981	
P(T<=t) one-tail	0.055663	
t Critical one-tail	2.131847	
P(T<=t) two-tail	0.111325	
t Critical two-tail	2.776445	

From table (8), we can notice the value of t Stat < t critical for one tail and two tail and the value of P(t<=t) two tail > alpha (0.05) that means the results of GWOS are better than other algorithms in static system.

Table 9. T-test for GWOS and other selected algorithms in parallel system

t-Test: Two-Sample Assuming Equal Variances		
	GWOS	*Others*
Mean	1728.333	1537.426667
Variance	214658.3	120604.4261
Observations	3	3
Pooled Variance	167631.4	
Hypothesized Mean Difference	0	
Df	4	
t Stat	0.57107	
P(T<=t) one-tail	0.299253	
t Critical one-tail	2.131847	
P(T<=t) two-tail	0.598506	
t Critical two-tail	2.776445	

The results in table (9) show that t Stat < t critical for one tail and two tail and the value of P(t<=t) two tail > 0.05, this show that GWOS results is statically better than other algorithms for solving CTSP in parallel system.

10. Conclusion

In this paper we used GWOS to solve CTSP, at which GWOS has been created and implemented. Two methods are used to prove efficiency of GWOS, the first one is the static method, its idea is to divide cites evenly among salesmen and each salesman must find the best track that increases benefits and decreases the cost. The tests and results proves the efficiency of our system compared with other selected algorithms (Nearest Neighbors, Nearest Random and Ant Colony System) and the results are shown in tables (3, 4, and 5).

The parallel method, in this manner all cites is available to all salesman but the salesman must visit the unvisited city in order to obtain full benefit of this city. If the salesman visits a visited city, he will pay the cost of arrive only and not obtain any benefit and if more than one salesman coincides at same city the benefit will be divided among them. Many datasets were tested and the experiments results prove the superiority of GWOS over others, to increase the dynamics of exploration and exploitation salesmen are used in the experiments and the results was validating our expectations and are shown in tables (6 and 7). The salesmen visited all cites to obtain grater benefits unlike other algorithms that visited less number of cities.

Based on above we can have concluded that GWOS algorithm proved its ability and efficacy in solving CTSP providing a new and good way to find the optimal path with the lowest costs, time and highest returns that can be used in many areas.

11. References

[1] M. J. u. G. R. G. Rinaldi, "THE TRAVELING SALESMAN PROBLEM," in *Handbooks in Operations Research and Management Science* Amsterdam: North-Holland, 1994, pp. 1-115.

[2] N. S. Alseelawi, E. K. Adnan, H. T. Hazim, H. Alrikabi, and K. Nasser, "Design and Implementation of an E-learning Platform Using N-Tier Architecture," 2020.

[3] P. M. Hariyadi, Phong Thanh Nguyen , Iswanto Iswanto ,Dadang Sudrajat. (2020, Jan) Traveling Salesman Problem Solution using Genetic Algorithm. *Journal of Critical Reviews , ISSN-2394-5125 Vol 7, Issue 1, 2020*. 56-61.

[4] N. Hussien, I. Ajlan, M. M. Firdhous, and H. Alrikabi, "Smart Shopping System with RFID Technology Based on Internet of Things," 2020.

[5] H. Demez, *Combinatorial Optimization: Solution Methods of Traveling Salesman Problem*. Gazimağusa,North Cyprus: Eastern Mediterranean University, 2013.

[6] A. E. Muyassar Dalli Hamad, Walid Abdelmoez ,Mahmoud M. El-Borai. (2016, Oct) Considering Stakeholders' Feedback in Requirements Prioritization using Social Network Analysis. *International Journal of Computer Science and Engineering volume 3 Issue 10*. 66-81.

[7] B. A.-K. Mohannad Abdul-Sattar Hameed, *Solving Competitive Traveling Salesmen Problem Using Ant Colony Algorithm*. Ramadi - Iraq: University of Anbar- College of Computer Science and Information Technology, 2016.

[8] k. N. Mohammad Mahdi Mohtadi. (2014) Solving TravelingSalesman Problem in Competitive Situations Using The Game Theory. *Applied Mathematics In Engneering ,Management and technology2(3)*. 311-325.

[9] M. Y. Belal Al-Khateeb. (2019, june) SOLVING MULTIPLE TRAVELING SALESMAN PROBLEM BY MEERKAT SWARM OPTIMIZATION ALGORITHM. *JOURNAL OF SOUTHWEST JIAOTONG UNIVERSITY*. Vol.54 No.3 1-10.

[10] B. A.-K. Mohammed Yousif. (2018) A Novel Metaheuristic Algorithm for Multiple Traveling Salesman Problem. *Jour of Adv Research in Dynamical & Control Systems, Vol. 10*. 2113-2122.

[11] Y. F. Hassan. (2018, Feb 27) Multi-level thinking cellular automata usinggranular computing title. *IET Intelligent Transport Systems* Vol. 12 Iss. 6, pp. 440-448.

[12] J. L. Graham Kendall. (2012, April) competitive traveliing salesman problem : A Hyper Heurisitc approach. *journal of the operational research society* 12-37.

[13] R. F. S´andor P. Fekete, Aviezri Fraenkel,Matthias Schmitt. (2004, Feb 19) Traveling salesmen in the presence of competition. *ELSEVIER*. Volume 313, Issue 3 377-392.

[14] H. T. Alrikabi, A. H. M. Alaidi, A. S. Abdalrada, and F. T. J. I. J. o. E. T. i. L. Abed, "Analysis the Efficient Energy Prediction for 5G Wireless Communication Technologies," vol. 14, no. 08, pp. 23-37, 2019.

[15] A. S. Abdullah, M. A. Abed, and I. Al Barazanchi, "Improving face recognition by elman neural network using curvelet transform and HSI color space," *Period. Eng. Nat. Sci.*, vol. 7, no. 2, pp. 430–437, 2019.

[16] A. Alaidi, I. Aljazaery, H. Alrikabi, I. Mahmood, and F. Abed, "Design and Implementation of a Smart Traffic Light Management System Controlled Wirelessly by Arduino," 2020.

[17] J. K. Li, G. (2015) Hyper-Heuristic Methodology to Generate Adaptive Strategies for Games. *IEEE Transactions on Computational Intelligence and AI in Games*. 1-10.

[18] B. Mohammed, R. Chisab, and H. Alrikabi, "Efficient RTS and CTS Mechanism Which Save Time and System Resources," 2020.

[19] B. K. T. Sujata Dash, Atta ur Rahman, *Handbook of Research on Modeling, Analysis, and Application of Nature*. Hershey PA, USA: IGI Global, 2018.

[20] G. K. Jiawei Li. (2017, march) A Hyperheuristic Methodology to GenerateAdaptiveStrategies for Games. *IEEE TRANSACTIONS ON COMPUTATIONAL INTELLIGENCE AND AI IN GAMES*. VOL. 9, NO. 1 pp1-11.

[21] B. Alkhateeb. (2018) The Selection of Particle Swarm Optimization learning Factors Values in Solving Multiple Travelling Salesman Problem *jour of Adv Research in Dynamical &Control Systems vol10 no7* 439-445.

[22] Y. F. H. Ayah M Hassan, Mohamed H Kholief. (2018, Jul.) A Deep Classification System forMedical Data Analysis. *Journal of Medical Imaging and Health Informatics*. 250-256.

[23] V. P. S. a. N. P. Tapan Parkash. (2019) Gray Wolf Optimization - Based Controller Design for Two Tank System. *Springer- Applicatins of Artificial Intelligence Techniques*. 501-507.

[24] B. A. M. Radwan Basim Thanoon. (2019, March) Modified Grey Wolf Optimization Algorithm by using Classical Optimization Methods. *International Journal of Computer Networks and Communications Security*. 49-61.

[25] S. I. A. Dibbendu Singha Sopto, M. A. H. Akhand,N. Siddique, "Modified Grey Wolf Optimization to Solve Traveling Salesman Problem," in *2018 International Conference on Innovation in Engineering and Technology (ICIET)*, Dhaka, Bangladesh, 2018.

[26] S. M. M. Seyedali Mirjalili, Andrew Lewis. (2014, March) Grey Wolf Optimizer. *ELSEVIER*. Advances in Engineering Software 69:46–61.

[27] S. Rashid, A. Ahmed, I. Al Barazanchi, and Z. A. Jaaz, "Clustering algorithms subjected to K-mean and gaussian mixture model on multidimensional data set," *Period. Eng. Nat. Sci.*, vol. 7, no. 2, pp. 448–457, 2019.

[28] M. M. a. B. Al-Khateeb. (2019, Sep) The blue monkey: A new nature inspired metaheuristic optimization. *Periodicals of Engineering and Natural Sciences Vol. 7, No. 3*. 1054-1066.

[29] U. s. Nitin Mittal, Balwinder singh sohi. (2016) Modified Gray Wolf Optimizer for Global Engneering Optimization. *hindawi - Applied Computational Intelligence and Soft Computing*. 1-16.

[30] Z.-M. G. a. J. Zhao. (2019) An Improved Gray Wolf Optimization Algorithm with Variable Wieghts. *Hindawi - Computitional Intelligence and Neuroscience*. 1-13.