TRAVELING SALESMAN PROBLEM:

HERUSTICS AND EMPIRICAL EVALUATION

AHMET SEDAT KAYA

JANUARY 2015

TRAVELING SALESMAN PROBLEM:

HERUSTICS AND EMPIRICAL EVALUATION

A THESIS SUBMITTED TO

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES OF

ÇANKAYA UNIVERSITY

BY

AHMET SEDAT KAYA

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE

DEGREE OF

MASTER OF SCIENCE

IN

THE DEPARTMENT OF

COMPUTER ENGINEERING

JANUARY 2015

Title of the Thesis: **Traveling Salesman Problem: Heuristics and Empirical Evaluation.**

Submitted by **Ahmet Sedat KAYA**

Approval of the Graduate School of Natural and Applied Sciences, Çankaya University.

[illegible]

Prof. Dr. Taner ALTUNOK
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

[illegible]

Prof. Dr. Müslim BOZYİĞİT
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

[illegible]

Assist. Prof. Dr. Emre SERMUTLU
Supervisor

Examination Date: 20.01.2015

Examining Committee Members

Assist. Prof. Dr. Emre SERMUTLU	(Çankaya University)	[illegible]
Prof. Dr. Ali Aydın SELÇUK	(TOBB ETU University)	[illegible]
Assoc. Prof. Dr. Hadi Hakan MARAŞ	(Çankaya University)	[illegible]

STATEMENT OF NON-PLAGIARISM PAGE

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name	:	Ahmet Sedat, KAYA
Signature	:	[illegible]
Date	:	20.01.2015

ABSTRACT

TRAVELING SALESMAN PROBLEM:
HERUSTICS AND EMPIRICAL EVALUATION

KAYA, Ahmet Sedat

M.Sc., Department of Computer Engineering

Supervisor: Assist. Prof. Dr. Emre SERMUTLU

January 2015, 48 Pages

In this thesis, three different algorithms with different perspectives (Close Couple, Worm, and Spider Web) has been developed to solve the Symmetric Traveling Salesman (TSP) heuristically. Improved algorithms with different data sets Distance Rate, Target have been tested. The running time and value of the solution have been compared.

In this context, several steps of evaluation were used for the comparison and improvement of algorithms. After each evaluation step, one candidate algorithm is eliminated. Eventually, an improved version of the Spider Web algorithm is the winner of this contest.

Keywords: Traveling Salesman Problem, Heuristics, Algorithm, Close Couple, Worm, Spider Web.

ÖZ

GEZGİN SATICI PROBLEMİ:

SEZGİSEL YÖNTEMLER VE AMPİRİK DEĞERLENDİRME

KAYA, Ahmet Sedat

Yüksek Lisans, Bilgisayar Mühendisliği Anabilim Dalı

Tez Yöneticisi: Yrd. Doç. Dr. Emre SERMUTLU

Ocak 2015, 48 Sayfa

Bu tezde Simetrik Gezgin Satıcı Problemine (GSP) probleminin optimum sezgisel çözümüne yönelik olarak farklı bakış açılarıyla 3 farklı algoritma (Yakın Çift, Solucan, Örümcek Ağı) geliştirilmiştir. Geliştirilen algoritmalar farklı veri kümeleri ile Uzaklık Oranı ve Hedef üzerinden test edilmiştir. Çalışma süreleri ve çözümün değerleri karşılaştırılmıştır.

Bu kapsamda algoritmaların geliştirilmesi ve iyileştirmesi için aşamalı bir değerlendirme yöntemi kullanılmıştır. Her bir değerlendirme aşamasında sonuçlar kaydedilerek bir aday algoritma elenmiştir. Sonuçta, iyileştirilmiş Örümcek Ağı algoritması bu yarışın galibi olmuştur.

Anahtar Kelimeler: Gezgin Satıcı Problemi, Sezgisel Yöntemler, Algoritma, Yakın Çift, Solucan, Örümcek Ağı.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to Assist. Prof. Dr. Emre SERMUTLU for his supervision, special guidance, suggestions, and encouragement through the development of this thesis.

I would like to express my deep gratitude to my family for their endless and continuous encouragement and support throughout the years.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

TABLES

LIST OF ABBREVIATIONS

TSP	Traveling Salesman Problem
SW1	Spider Web First Version
SW2	Spider Web Second Version
SW3	Spider Web Third Version
TSPLIB	Discrete and Combinatorial Optimization Library

CHAPTER 1

INTRODUCTION

1.1. Traveling Salesman Problem (TSP)

TSP is one of the most famous problems in applied mathematics and theoretical computer science Many different methods were tried to solve this problem to achieve an optimum cost (i.e. for time and/or distance). Therefore, it is important to ongoing research and development to improve this success to find the optimum solution for TSP.

A Nondeterministic Polynomial time problem (NP-Problem) is a problem whose solution can be verified by a computer in polynomial time. Moreover, NP- hard problem is when every problem in NP can be reduced in polynomial time, also NP-complete is between both NP and NP-hard as what have in TSP [1]. Therefore try to develop heuristic methods that use experiment-based techniques to find the optimum solution for symmetric TSP. [2]

Three new proposed algorithms are Close Couple, Worm and Spider Web have been developed with different perspectives in this thesis. Different data sets have been used for evaluation and development of algorithms. For small data sets the results are compared to those found by Brute-Force technique. Besides that, results are used for update and optimize algorithms.

Three levels of comparisons and evaluations (based on speed and quality of solution) between algorithms are presented. In each level one algorithm is eliminated according to obtained results.

The results shows Spider Web Algorithm obtained the highest results. After that, second and third enhancements of Spider Web's algorithm have been developed based on the results of evaluations and use other algorithms techniques. Furthermore, the results show improved versions of Spider Web performed better than the original Spider Web Algorithm.

1.2. Objectives

This thesis aimed to propose a new algorithm to find the optimum solution for symmetric TSP, and then enhance these algorithms to achieve an optimum cost. All these algorithm are tested with different data sets.

1.3. Outlines

The reminder of this thesis structured as below.

Chapter two: Background and related works discuss the TSP problem and the previous studies about this problem.

Chapter three: Developed algorithms discuss the study principles of Close Couple, Worm and Spider Web Algorithms, pseudo codes developed within this thesis study have been given. Furthermore, the way those algorithms work have been explained in detail with a sample analysis.

Chapter four: Evaluation of the algorithms and the results discuss the information about the test environment of algorithms and the data sets used the evaluation method has been explained in detail. The evaluation results have also been shown both with tables and graphics and each evaluation step the obtained results have been summarised and detailed information has been given about the results.

Chapter five: Conclusions and future works is the last part the information obtained from the results of the studies within the thesis has been evaluated.

CHAPTER 2

BACKGROUND AND RELATED WORKS

2.1 TSP-Problem

TSP is the most studied discrete optimization problem. Its popularity is due to the facts that TSP is easy to formulate, difficult to solve, and has a large number of applications.

K. Menger [3] was the first researcher to consider the TSP. He observed that the problem can be solved by examining all permutations one by one. Realizing that the complete enumeration of all permutations was not possible for graphs with a large number of vertices, he also looked at the most natural nearest neighbour strategy and pointed out that this heuristic, generally, does not produce the shortest route. (In fact, the nearest neighbour heuristic will generate the worst possible route for some problem instances of each size [4].)

(P) Can be defined as a set of problems which could be solved in polynomial time (i.e. *easy to solve*). And (NP-hard) is a set of problems not known to be solvable in polynomial time (i.e. hard to solve). A Heuristic Algorithm is not guaranteed to find the optimal solution, but find a solution (i.e. reasonably close) to the optimal solution in reasonable amount of time.

2.2 Asymmetric TSP and Symmetric TSP

A salesman is required to visit each of n given cities once and only once, starting from any city and returning to the original place of departure. We have to find the tour he should choose in order to minimize the total travel distance, assuming the distances between any pair of cities are known. Distance can be replaced by another notion, such as time or money. In the following the term "cost" is used to represent any such notion [5].

Given a cost matrix $C = (c_{ij})$ where c_{ij} represents the cost of going from city i to city j, where $i, j = 1 \cdots n$ find a permutation $(i_1, i_2, \cdots i_n)$ of the integers from 1 through n that minimizes the quantity

$$c_{i_1 i_2} + c_{i_2 i_3} + \cdots + c_{i_n i_1}$$

Properties of the cost matrix C are used to classify problems. If $c_{ij} = c_{ji}$ for all i and j, the problem is said to be symmetric otherwise, it is asymmetric.

2.3 TSP Algorithms

The problem is easy to state, but hard to solve. The difficulty becomes apparent when one considers the number of possible tours, an astronomical figure even for a relatively small number of cities. For a symmetric problem with n cities there are $(n-1)!\,/\,2$ possible tours. If n is 20, there are more than 10^{18} tours. The 13,509 city problem, which is successfully solved by the algorithm described in this paper, contains about $10^{50,000}$ possible tours. In comparison it may be noted that the number of elementary particles in the universe has been estimated to be only10^{87}. It has been proven that TSP is a member of the set of NP-complete problems. [6]

This is a class of difficult problems whose time complexity is probably exponential. The members of the class are related so that if a polynomial time were found for one problem, polynomial time algorithms would exist for all of them. However, it is commonly believed that no such polynomial algorithm exists. Therefore, any attempt to construct a general algorithm for finding optimal solutions for the TSP in polynomial time must (probably) fail. Algorithms for solving the TSP may be divided into two classes: Exact algorithms and Approximate (or heuristic) algorithms.

2.3.1 Exact algorithms

The exact algorithms are guaranteed to find the optimal solution in a bounded number of steps. The most effective exact algorithms are cutting-plane or facet-finding algorithms [7, 8]. These algorithms are quite complex, with codes on the order of 10,000 lines. In addition, the algorithms are very demanding of computer power. The previously mentioned 13,509-city problem was solved over a period of three months using a cluster of 3 servers, a total of 12 processors, and 32 PCs [8].

2.3.2 Approximate algorithms (Heuristic)

Nearest Neighbour

This is perhaps the simplest and most straightforward TSP heuristic. The key to this algorithm is to always visit the nearest city. Nearest Neighbour, $O\ (n^2)$

1. Select a random city.
2. Find the nearest unvisited city and go there.
3. Are there any unvisited cities left? If yes, repeat step 2.
4. Return to the first city.

The Nearest Neighbour algorithm will often keep its tours within % 25 of the Held-Karp lower bound [9].

Greedy

The Greedy heuristic gradually constructs a tour by repeatedly selecting the shortest edge and adding it to the tour as long as it doesn't create a cycle with less than *N* edges, or increases the degree of any node to more than 2. We must not add the same edge twice of course.

Greedy, $O(n^2 \log n)$

1. Sort all edges.

2. Select the shortest edge and add it to our tour if it doesn't violate any of the above constraints.

3. Do we have *N* edges in our tour? If not, repeat step 2.

The Greedy algorithm normally keeps within 15- 20% of the Held-Karp lower bound [9].

Insertion Heuristics

Insertion heuristics are quite straightforward, and there are many variants to choose from. The basics of insertion heuristics is to start with a tour of a subset of all cities, and then inserting the rest by some heuristic. The initial sub tour is often a triangle or the convex hull. One can also start with a single edge as sub tour.

Nearest Insertion, $O(n^2)$

1. Select the shortest edge, and make a sub tour of it.
2. Select a city not in the sub tour, having the shortest distance to any one of the cities in the sub tour.
3. Find an edge in the sub tour such that the cost of inserting the selected city between the edge's cities will be minimal.
4. Repeat step 2 until no more cities remain. [10]

Christofides

Most heuristics can only guarantee a worst-case ratio of 2 (i.e. a tour with twice the length of the optimal tour). Professor Nicos Christofides extended one of these algorithms and concluded that the worst-case ratio of that extended algorithm was 3/2. This algorithm is commonly known as Christofides heuristic.

Original Algorithm (Double Minimum Spanning Tree), worst-case ratio 2, $O\ (n^2 log2\ (n))$

1. Build a minimal spanning tree (MST) from the set of all cities.
2. Duplicate all edges, we can now easily construct an Euler cycle.
3. Traverse the cycle, but do not visit any node more than once, taking shortcuts when a node has been visited. [10]

Two-opt exchange

Euclidean problems observed by this improvement approach. It can be easily shortened if a Hamiltonian cycle crosses itself. Namely, erase two edges that cross and reconnect the resulting two paths by edges that do not cross (this is always possible). The new cycle is shorter than the old one. [11]

2.4. TSP Applications

2.4.1 Practical applications

The algorithm aim to develop and heuristics for practical traveling to solve the salesman problem, give a survey on some of the possible applications. It covers some important cases but not complete. [12]

Drilling of printed circuit boards

TSP direct application is the drilling problem it plays an important solution role in economical manufacturing of printed circuit boards (PCBs). In an industry application a computational study of a large electronics company can be found in Gr’otschel, J’unger and Reinelt (1991). [11]

X-Ray crystallography

The TSP important application occurs in the analysis of the structure of crystals (Bland and Shallcross (1987), Dreissig and Uebach (1990). To obtain information about the structure of crystalline material X-ray diffract meter is used. Detector measures the intensity of X-ray reflections of the crystal from various positions. Whereas the measurement itself can be accomplished quite fast, there is a considerable overhead in positioning time since up to hundreds of thousands positions have to be realized for some experiments. [11]

The order-picking problem in warehouses

This problem is associated with material handling in a warehouse (Ratliff and Rosenthal (1981). Assuming a warehouse an order arrives for a certain subset of the items stored in the warehouse. This order to ship collected by some vehicle and all items of them to the customer. The relation to the TSP is immediately seen. The storage locations of the items correspond to the nodes of the graph. The distance between two nodes is given by the time needed to move the vehicle from one location to the other. Finding a shortest route for the vehicle with minimum pickup time can now be solved as a TSP. [11]

Scheduling with sequence dependent process times

If we are given n jobs that have to be performed on some machine. The time to process job j is tij if i the job is performed immediately before$j$. The task is to find an execution sequence for the jobs such that the total processing time is as short as possible. And also, this problem can be modelled as a directed Hamiltonian path problem. [11]

Vehicle routing

Suppose that in a city n mail boxes have to be emptied every day within a certain period of time, say 1 hour. The problem is to find the minimum number of trucks to do this and the shortest time to do the collections using this number of trucks. As another example, suppose that n customers require certain amounts of some commodities and a supplier has to satisfy all demands with a fleet of trucks. The problem is to find an assignment of customers to the trucks and a delivery schedule for each truck so that the capacity of each truck is not exceeded and the total travel distance is minimized. [11]

CHAPTER 3

DEVELOPED ALGORITHMS

3.1. Developed Algorithms Information

Three different algorithms have been developed to find optimum solution of symmetric TSP problem. These are respectively;

- Close Couple
- Worm
- Spider Web Algorithms.

Symmetric TSP problem is in this context the connection of provinces (nodes/points) with each other has been presumed as undirected graph.

$$Distance(A \rightarrow B) = Distance(B \rightarrow A)$$

Close Couple and Worm Algorithms have been developed similar to each other. These algorithms have been formed with the hypothesis of the solution of TSP problem's formation is between the closest points to each other.

The Spider Web Algorithm, has been created with the assumption that starting from the farthest point to each other in order to find the solution, the optimal solution can be found with the addition of another point close to remote locations. The developed algorithms find the optimal solution in the shortest time. Thus, focuses on the most important part of TSP which is increased points that are lead to more computation time.

3.2. Close Couple Algorithm

3.2.1. Study principle of Close Couple Algorithm

Close Couple Algorithm starts with the hypothesis that the most optimum solution in TSP problem occurs between two closest points.

In this context algorithm tries to extrapolate to optimum result by practising the steps below.

The closest couple sets are formed according to the distance between the points. Then if possible elimination process is carried out on specified couple sets. Within this context with the hypothesis $A \rightarrow B$, $B \rightarrow A$ couple, if the closest point to A is point B and the closest point to B is point A then $B \rightarrow A$ couple is eliminated. Among the rest of the couple sets the least distant couple set is chosen. The first chosen couple is evaluated as the start and end point. The acquisitions of the rest of the couple sets are estimated according to their distance to start and end points and it is decided whether to start from start point or end point. According to the decision the most suitable couple is chosen and start and end points are updated according to this selection. The process continues until the couple sets to be estimated finish.

```
Procedure CloseCouple ()
{
Set CloseCoupleList [ ] = Create Close Couple List according to distances
Eliminate Close Couple List
Set ShortestCloseCouple=Select Shortest Close Couple in CloseCoupleList
Set Start City=ShortestCloseCouple.FromCity
Set End City=ShortestCloseCouple.ToCity
While CloseCoupleList. Length>0
Begin
        Set StartCityGain=Calculate Gain (Start City, CloseCoupleList)
        Set EndCityGain=Calculate Gain (End City, CloseCoupleList)
        IF (StartCityGain > EndCityGain)
                THEN
                        Set Gain City=Start City
                ELSE
                        Set Gain City=End City
        Set FindCloseCouple=GetCloseCouple (Gain City)
        IF (Gain City= Start City)
                THEN
                        Set Start City=FindCloseCouple.ToCity
                ELSE
                        Set End City=FindCloseCouple.ToCity
        CloseCoupleList. Remove (FindCloseCouple)
End
}
```

3.2.2. Sample analysis

For the sample analysis of Close Couple Algorithm Ankara, Istanbul, Izmir and Antalya cities and the distance between them has been based by using the Turkey highway map distance table between provinces.

Cities	**Ankara**	**Istanbul**	**Izmir**	**Antalya**
Ankara	0	453	585	537
Istanbul	453	0	563	723
Izmir	585	563	0	465
Antalya	537	723	465	0

Table 1 Close Couple Algorithm Sample City Table

According to distance the closest couple sets

Ankara / Istanbul

Istanbul / Ankara

Izmir / Antalya

Antalya / Izmir

The couple sets after the elimination

Ankara / Istanbul = 453 Km

Izmir / Antalya = 465 Km

Ankara / Istanbul is chosen as the start couple set.

Start Point: Ankara

End Point: Istanbul

Start / End points are determined according to the acquisitions

Start Point = Ankara

The distance of the rest couple to Ankara

Ankara / Izmir = 585   Ankara / Antalya = 537   the gain = 585 – 537 = 48 km

End Point = Istanbul

The distance of the rest couple to Istanbul

Istanbul / Izmir = 563 Istanbul / Antalya = 723 the gain = 723 – 563 = 160 km

The most suitable couple is calculated according to the determined point and the process is finished.

As Istanbul is the most profitable **End point**, **Istanbul** is chosen.

Then choosing the closest couple to Istanbul, the closest province of the couple is chosen.

The closest couple to Istanbul is Izmir / Antalya and the closest province is Izmir. Then according to the selection the couple can change direction (If the closest provinces were Izmir, the direction would change to Antalya / Izmir instead of Izmir / Antalya).

The list is updated as **Ankara Antalya Izmir Istanbul.** **End point** is updated as **Antalya.** If there are other couples, the process continues until all the couples are operated.

Iteration Number	**Formation of Points**	**Explanation**
#1	Ankara Istanbul	Start and end points are determined according to the closest couple set
#2	Ankara Antalya Izmir Istanbul	Start and end points are decided according to the acquisition calculation

Table 2 Close Couple Algorithm Iteration Table

3.3. Worm Algorithm

3.3.1. Study principle of Worm Algorithm

Worm Algorithm uses the distance between points for the solution of TSP problem. In this context algorithm tries to reach the optimum result by applying the steps below.

The distance which has the least interval is determined from the list of distance between points. Afterwards the cities in the distance list are determined as the start and end points. Average value is calculated finding the distances including start and end points, after this point which has the farthest average value is chosen, the closest distance list to the chosen point is found. Start and end points are updated. The distances which include the defined point are removed from the distance list. The process continues until the distance length equal to 1.

```
Procedure **Worm** ( )
{
Set Shortest Distance = Select Shortest Distance from Distance List
Set Start City=ShortestDistance.FromCity
Set End City=ShortestDistance.ToCity
While DistanceList.Length>1
Begin
    Set StartCityGain=Calculate Gain (Start City, DistanceList. Where (FromCity|ToCity=Start City))
    Set EndCityGain=Calculate Gain (End City, DistanceList.Where (FromCity|ToCity=End City))
    IF (StartCityGain > EndCityGain)
        THEN
            Set Gain City=Start City
        ELSE
            Set Gain City=End City
    Set FindShortestDistance=GetShortestDistance (Gain City)
    DistanceList.RemoveAll.Where (FromCity|ToCity=Gain City)
    IF (Gain City= Start City)
        THEN
            Set Start City=FindShortestDistance.ToCity
        ELSE
            Set End City=FindShortestDistance.ToCity
End
}
```

3.3.2 Sample analysis

Table 3 shows the same Close Couple Algorithm that used for the sample analysis of Worm Algorithm.

Cities	**Distance**
Ankara-Istanbul	453
Ankara-Izmir	585
Ankara-Antalya	537
Istanbul-Izmir	563
Istanbul-Antalya	723
Izmir-Antalya	465

Table 3 Worm Algorithm Sample City Table

Distance list is formed.

Ankara / Istanbul = 453

Ankara / Izmir = 585

Ankara / Antalya = 537

Istanbul / Izmir = 563

Istanbul / Antalya = 723

Izmir / Antalya = 465

The closest distance Ankara / Istanbul is determined as the start and end point.

Start Point: Ankara

Finish Point: Istanbul

The acquisitions of the values in the distance list are calculated according to start and finish points.

Start point = Ankara

The distance of the rest of the couples according to Ankara

Ankara / Izmir = 585   Ankara/Antalya = 537   Profit = 585 – 537 = 48 km

End Point = Istanbul

The distance of the rest of the couples according to Istanbul

Istanbul / Izmir = 563 Istanbul / Antalya = 723 Profit = 723 – 563 = 160 km

The point which has the most profit is chosen. Start and finish points are updated by choosing the closest other point to the chosen one before.

Istanbul = 160 Km

Izmir is chosen as the closest province to the finish point. Finish point is updated according to the choice.

Distance list is cleared according to the changed list.

The distance list is cleared as the finish point Istanbul has changed.

The remaining Distance List

Ankara / Izmir = 585

Ankara / Antalya = 537

Izmir / Antalya = 465

The process continues as the distance list is bigger than 1.

Start Point according to Ankara

Ankara / Antalya = 537

Average value = 537/1 = 537

End point according to Izmir

Izmir / Antalya = 465

Average value = 465/1 = 465

In this case Ankara is chosen.

Ankara, Antalya, Izmir, Istanbul

Start Point =Antalya   End Point = Izmir

Remaining Distance list

Izmir / Antalya

The algorithm ends as the length of the distance list is 1.

At the end of the process the result is as it is below.

Iteration Number	**Formation of Points**	**Explanation**
#1	Ankara Istanbul	The determination of the start and finish point according to the distance
#2	Ankara Izmir Istanbul	The determination of the most suitable point according to the start and End points
#3	Ankara Antalya Izmir Istanbul	Other points are determined according to the remaining points

Table 4 Worm Algorithm Iteration Table

3.4. Spider Web Algorithm

3.4.1. Study principle of Spider Web Algorithm

Spider Web Algorithm starts with the hypothesis that the most optimum solution in TSP problem occurs between two farthest points. In this context algorithm tries to extrapolate to optimum result by practising the steps below.

The first step creates an average list formed by using the distances between points, the point which has the least average is determined. Second step by determined three reference points according to the distance by using this point. Afterwards, reference points are removed from the list. Third step starting from the point which has the highest average it is decided between which references points it will be added by using cost calculation. The addition is done according to the suitable reference point. The process continues until there is no point to calculate.

```
Procedure **Spider Web** ()
{
Set CityAverageList [] = Calculate Average for Each City from Distance List
CityAverageList. Sort (Descending Average)
Set Reference City=Select Shortest Average City from CityAverageList
Set First City=Farthest City according to Reference City
Set Second City=Farthest City according to First City
Set Third City=Select Longest City from CityAverageList except First City and Second City
CityLinkList.AddAll (FirstCity-SecondCity, SecondCity-ThirdCity, ThirdCity-FirstCity)
CityAverageList.RemoveAll (FirstCity|SecondCity|ThirdCity)
While CityAverageList. Length>0
Begin
        Set City=CityAverageList [0]
        Set Calculate City=Select Shortest City from CityLinkList according to City
        Set Link City=GetLinkCityWithMinimumCost (Calculate City, CityLinkList)
        CityLinkList. Add (LinkCity.FromCity-City)
        CityLinkList. Add (City-LinkCity.ToCity)
        CityLinkList. Remove (Link City)
        CityAveragelist.Remove (City)
End
}
```

3.4.2. Sample analysis

The list used in Close Couple and Worm Algorithms before is used as an example.

Average list is formed by using the list determined as an example.

Cities	Ankara	Istanbul	Izmir	Antalya	Average
Ankara	0	453	585	537	1575 / 3 = 525
Istanbul	453	0	563	723	1739 / 3 = 579
Izmir	585	563	0	465	1613 / 3 = 537
Antalya	537	723	465	0	1725 / 3 = 575

Table 5 Spider Web Algorithm Sample City Table

Reference point is determined according to the average value and other point is determined according to the reference point.

Reference point: Ankara

Izmir which is the farthest point to the reference point is start point

Istanbul which is the farthest point to start point is finish point

Out of reference, start and finish points Antalya which has the highest average point is determined as the middle point.

The imaginative picture is as it is below.

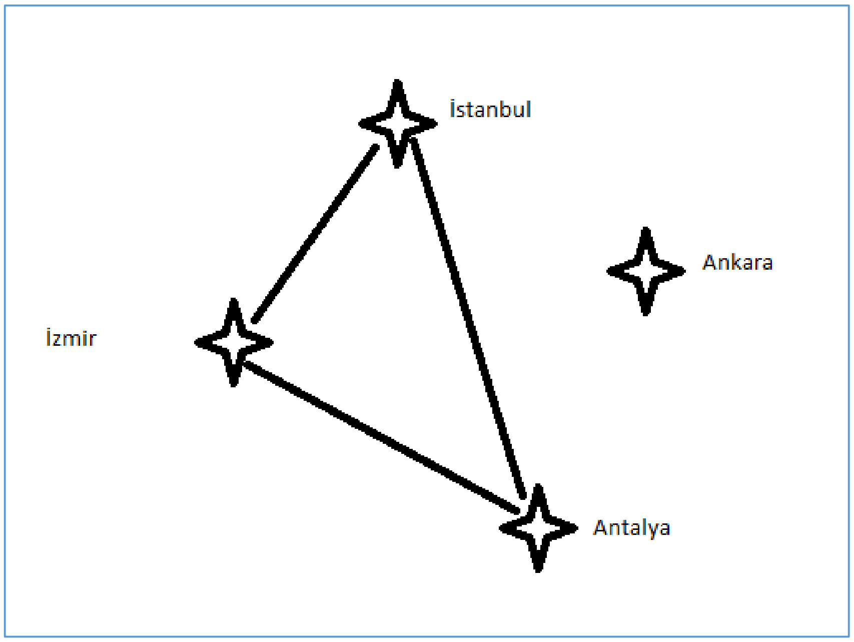


Figure 1 Spider Web Algorithm reference points

As Ankara is the remaining city, the calculation is done for Ankara.

Istanbul is determined as the closest city to Ankara.

The places to add are determined according to Istanbul.

Istanbul / Antalya or Istanbul / Izmir.

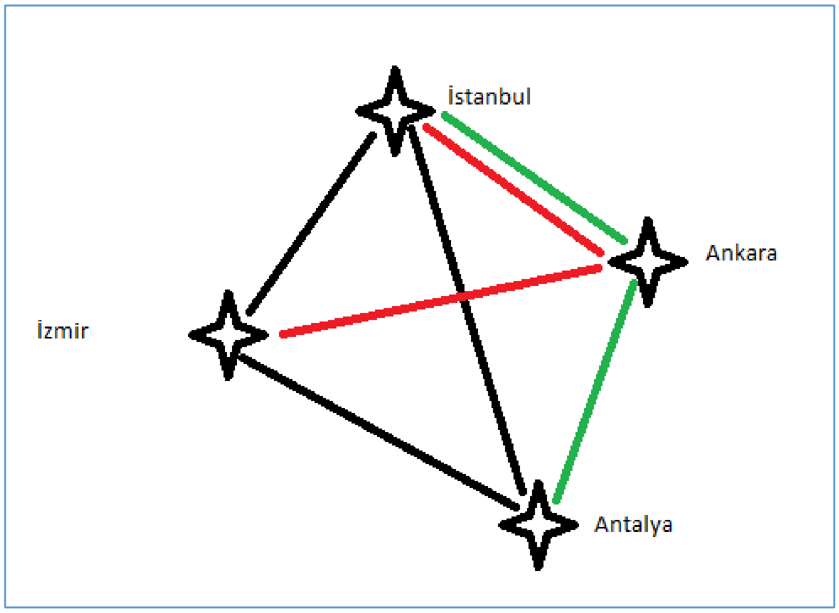


Figure 2 Spider Web Algorithm the determination of the alternative points

The cost of the roads to add is calculated.

Here the cost of two roads s calculated.

Addition cost of Istanbul/Antalya

Istanbul →Ankara →Antalya – Istanbul →Antalya = (453 + 537) – 723 = 267

The cost of Istanbul / Izmir

Istanbul →Ankara →Izmir – Istanbul →Izmir = (453+585) – 563 = 475

According to the cost calculation, addition of Ankara between Istanbul →Antalya effects the distance less. In this case new situation is as it is below.

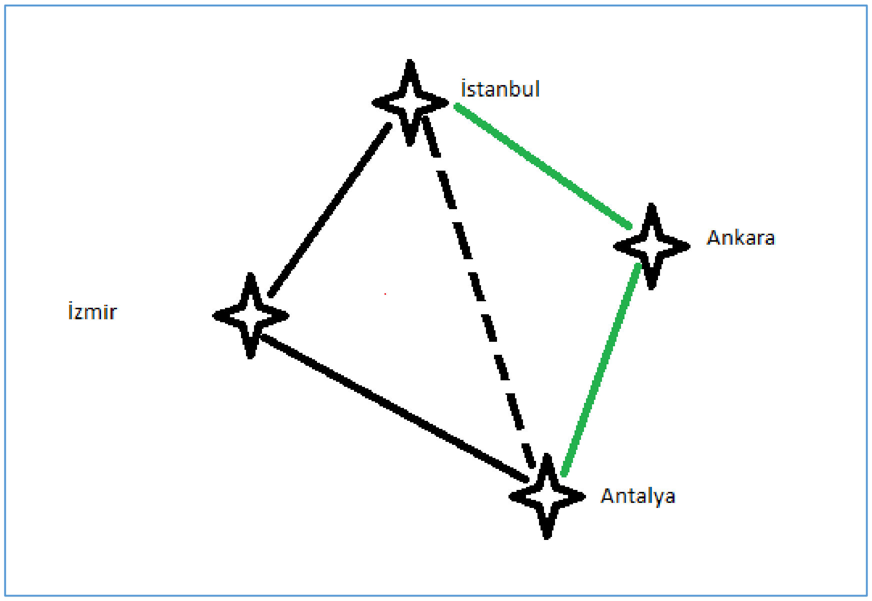


Figure 3 Spider Web Algorithm the calculation of the points

Algorithm is completed as there is no other city.

The most optimum way is determined like this:

Iteration Number	Formation of Points	Explanation
#1	Izmir Istanbul	Determination of the start and finish points according to the reference
#2	Izmir Antalya Istanbul	Determination of the middle point
#3	Izmir Antalya Ankara Istanbul	The determination of the point according to the most suitable place

Table 6 Spider Web Algorithm Iteration Table

CHAPTER 4

EVALUATION OF THE ALGORITHMS AND THE RESULTS

4.1. Testing Environment

For the evaluation of the algorithms developed within this thesis the computer which has the hardware and software features below is used.

Operating System	Windows 7
CPU	Intel Core I3-2.5 GHz
RAM	4 GB
HDD	500 GB

Table 7 Algorithm Evaluation Environment

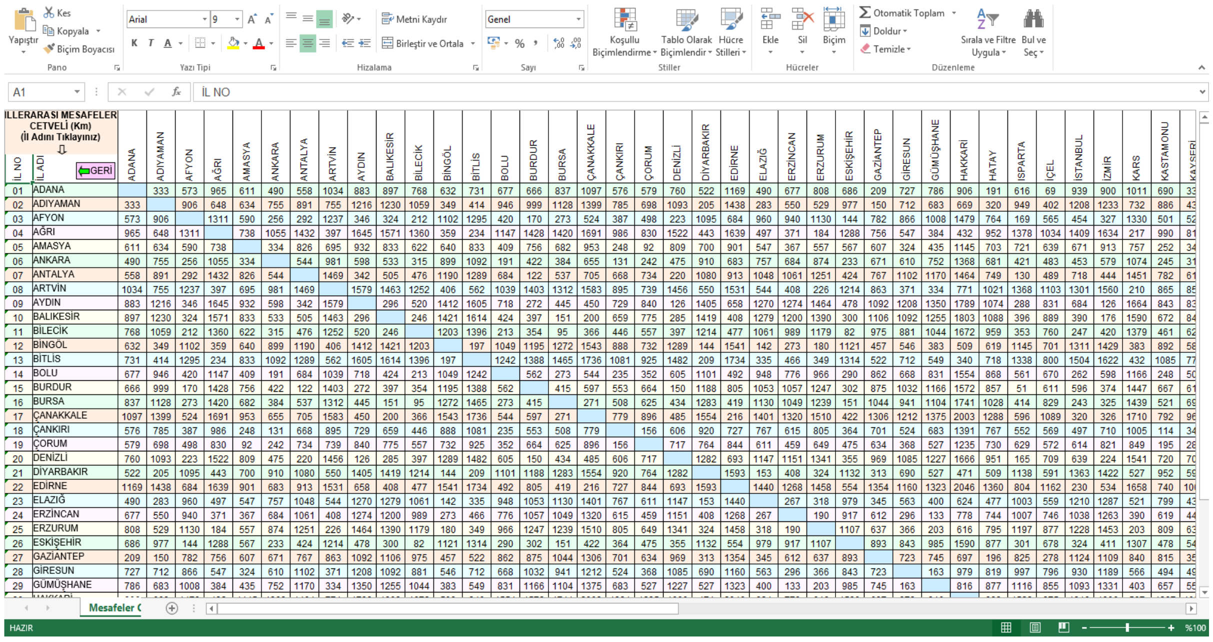
4.2. Data Set

Two data sets have been formed during the evaluation of the algorithms.

4.2.1. Cities data set

It has been used to calculate the Distance Rate of the values acquired from the algorithms via Brute-Force Method.

In this context the provinces of Turkey and the distance list of them has been used as the points. The values in The Map of Turkish Republic General Directorate of Highways are used as the reference data set for this.



ILLERARASI MESAFELER CETVELİ (Km)
(İl Adını Tıklayınız)

İL NO	İL ADI	ADANA	ADIYAMAN	AFYON	AĞRI	AMASYA	ANKARA	ANTALYA	ARTVİN	AYDIN	BALIKESİR	BİLECİK	BİNGÖL	BİTLİS	BOLU	BURDUR	BURSA	ÇANAKKALE	ÇANKIRI	ÇORUM	DENİZLİ
01	ADANA		333	573	965	611	490	558	1034	883	897	768	632	731	677	666	837	1097	576	579	760
02	ADIYAMAN	333		906	648	634	755	891	755	1216	1230	1059	349	414	946	999	1128	1399	785	698	1093
03	AFYON	573	906		1311	590	256	292	1237	346	324	212	1102	1295	420	170	273	524	387	498	223
04	AĞRI	965	648	1311		738	1055	1432	397	1645	1571	1360	359	234	1147	1428	1420	1691	986	830	1522
05	AMASYA	611	634	590	738		334	826	695	932	833	622	640	833	409	756	682	953	248	92	809
06	ANKARA	490	755	256	1055	334		544	981	598	533	315	899	1092	191	422	384	655	131	242	475
07	ANTALYA	558	891	292	1432	826	544		1469	342	505	476	1190	1289	684	122	537	705	668	734	220
08	ARTVİN	1034	755	1237	397	695	981	1469		1579	1463	1252	406	562	1039	1403	1312	1583	895	739	1456
09	AYDIN	883	1216	346	1645	932	598	342	1579		296	520	1412	1605	718	272	445	450	729	840	126
10	BALIKESİR	897	1230	324	1571	833	533	505	1463	296		246	1421	1614	424	397	151	200	659	775	285
11	BİLECİK	768	1059	212	1360	622	315	476	1252	520	246		1203	1396	213	354	95	366	446	557	397
12	BİNGÖL	632	349	1102	359	640	899	1190	406	1412	1421	1203		197	1049	1195	1272	1543	888	732	1289
13	BİTLİS	731	414	1295	234	833	1092	1289	562	1605	1614	1396	197		1242	1388	1465	1736	1081	925	1482
14	BOLU	677	946	420	1147	409	191	684	1039	718	424	213	1049	1242		562	273	544	235	352	605
15	BURDUR	666	999	170	1428	756	422	122	1403	272	397	354	1195	1388	562		415	597	553	664	150
16	BURSA	837	1128	273	1420	682	384	537	1312	445	151	95	1272	1465	273	415		271	508	625	434
17	ÇANAKKALE	1097	1399	524	1691	953	655	705	1583	450	200	366	1543	1736	544	597	271		779	896	485
18	ÇANKIRI	576	785	387	986	248	131	668	895	729	659	446	888	1081	235	553	508	779		156	606
19	ÇORUM	579	698	498	830	92	242	734	739	840	775	557	732	925	352	664	625	896	156		717
20	DENİZLİ	760	1093	223	1522	809	475	220	1456	126	285	397	1289	1482	605	150	434	485	606	717	
21	DİYARBAKIR	522	205	1095	443	700	910	1080	550	1405	1419	1214	144	209	1101	1188	1283	1554	920	764	1282
22	EDİRNE	1169	1438	684	1639	901	683	913	1531	658	408	477	1541	1734	492	805	419	216	727	844	693
23	ELAZIĞ	490	283	960	497	547	757	1048	544	1270	1279	1061	142	335	948	1053	1130	1401	767	611	1147
24	ERZİNCAN	677	550	940	371	367	684	1061	408	1274	1200	989	273	466	776	1057	1049	1320	615	459	1151
25	ERZURUM	808	529	1130	184	557	874	1251	226	1464	1390	1179	180	349	966	1247	1239	1510	805	649	1341
26	ESKİŞEHİR	686	977	144	1288	567	233	424	1214	478	300	82	1121	1314	290	302	151	422	364	475	355
27	GAZİANTEP	209	150	782	756	607	671	767	863	1092	1106	975	457	522	862	875	1044	1306	701	634	969
28	GİRESUN	727	712	866	547	324	610	1102	371	1208	1092	881	546	712	668	1032	941	1212	524	368	1085
29	GÜMÜŞHANE	786	683	1008	384	435	752	1170	334	1350	1255	1044	383	549	831	1166	1104	1375	683	527	1227

İL NO	İL ADI	DİYARBAKIR	EDİRNE	ELAZIĞ	ERZİNCAN	ERZURUM	ESKİŞEHİR	GAZİANTEP	GİRESUN	GÜMÜŞHANE	HAKKARİ	HATAY	ISPARTA	İÇEL	İSTANBUL	İZMİR	KARS	KASTAMONU
01	ADANA	522	1169	490	677	808	686	209	727	786	906	191	616	69	939	900	1011	690
02	ADIYAMAN	205	1438	283	550	529	977	150	712	683	669	320	949	402	1208	1233	732	886
03	AFYON	1095	684	960	940	1130	144	782	866	1008	1479	764	169	565	454	327	1330	501
04	AĞRI	443	1639	497	371	184	1288	756	547	384	432	952	1378	1034	1409	1634	217	990
05	AMASYA	700	901	547	367	557	567	607	324	435	1145	703	721	639	671	913	757	252
06	ANKARA	910	683	757	684	874	233	671	610	752	1368	681	421	483	453	579	1074	245
07	ANTALYA	1080	913	1048	1061	1251	424	767	1102	1170	1464	749	130	489	718	444	1451	782
08	ARTVİN	550	1531	544	408	226	1214	863	371	334	771	1021	1368	1103	1301	1560	210	865
09	AYDIN	1405	658	1270	1274	1464	478	1092	1208	1350	1789	1074	288	831	684	126	1664	843
10	BALIKESİR	1419	408	1279	1200	1390	300	1106	1092	1255	1803	1088	396	889	390	176	1590	672
11	BİLECİK	1214	477	1061	989	1179	82	975	881	1044	1672	959	353	760	247	420	1379	461
12	BİNGÖL	144	1541	142	273	180	1121	457	546	383	509	619	1145	701	1311	1429	383	892
13	BİTLİS	209	1734	335	466	349	1314	522	712	549	340	718	1338	800	1504	1622	432	1085
14	BOLU	1101	492	948	776	966	290	862	668	831	1554	868	561	670	262	598	1166	248
15	BURDUR	1188	805	1053	1057	1247	302	875	1032	1166	1572	857	51	611	596	374	1447	667
16	BURSA	1283	419	1130	1049	1239	151	1044	941	1104	1741	1028	414	829	243	325	1439	521
17	ÇANAKKALE	1554	216	1401	1320	1510	422	1306	1212	1375	2003	1288	596	1089	320	326	1710	792
18	ÇANKIRI	920	727	767	615	805	364	701	524	683	1391	767	552	569	497	710	1005	114
19	ÇORUM	764	844	611	459	649	475	634	368	527	1235	730	629	572	614	821	849	195
20	DENİZLİ	1282	693	1147	1151	1341	355	969	1085	1227	1666	951	165	709	639	224	1541	720
21	DİYARBAKIR		1593	153	408	324	1132	313	690	527	471	509	1138	591	1363	1422	527	952
22	EDİRNE	1593		1440	1268	1458	554	1354	1160	1323	2046	1360	804	1162	230	534	1658	740
23	ELAZIĞ	153	1440		267	318	979	345	563	400	624	477	1003	559	1210	1287	521	799
24	ERZİNCAN	408	1268	267		190	917	612	296	133	778	744	1007	746	1038	1263	390	619
25	ERZURUM	324	1458	318	190		1107	637	366	203	616	795	1197	877	1228	1453	203	809
26	ESKİŞEHİR	1132	554	979	917	1107		893	843	985	1590	877	301	678	324	411	1307	478
27	GAZİANTEP	313	1354	345	612	637	893		723	745	697	196	825	278	1124	1109	840	815
28	GİRESUN	690	1160	563	296	366	843	723		163	979	819	997	796	930	1189	566	494
29	GÜMÜŞHANE	527	1323	400	133	203	985	745	163		816	877	1116	855	1093	1331	403	657

Table 8 General Directorate of Highways the Distance Table between Cities

Afterwards on the determined cities in different combinations 1000 data have been formed. Besides as the Brute-Force Method is going to be used, the perspective of data formed has been increased by using the highest city number (4,5,6,7,8,9,10) to be calculated.

4.2.2. Random data set

It has been used to speed values are compared by using more points for comparison of algorithms with each other. Within this context a runtime $n \times n$ matrix has been formed by the preparation of an application and values between 1 - 1000 have been assigned to the matrix. It has been tried to reach the top level value with the values 8, 16, 32, 64, 128 until reaching this matrix value.

An interface has been prepared to form random data set. The parameters and the explanations used in the interface are indicated below.

Matrix Size: The size of the matrix to be formed is entered. For example when 2050 value is entered 2050×2050 matrix forms.

Random Start Value: is the value to be assigned to the entered matrix.

Random Stop Value: is value of random finished value to be assigned to the matrix. For example when Random Start Value equal to 1 and Random Stop Value is equal to 1000, the values of the elements in the matrix form random values are between 1 - 1000.

Sample Rate Value: Algorithm repeat value. After that this value using for calculation of average time.

Combination Value: 8, 16, 32, 64, 128, 256, 1024, 2048 according to the given matrix value the combination list is formed starting from 8 and increasing it twice each time up to the matrix size.

Total Duration (ms): It indicates the passing time according to the given Sample Rate value.

Average Duration (ms): Average time which is found by dividing total time by Sample Rate.

The durations which occur are tested for each algorithm and their speed values are recorded. Finally the algorithms chosen from the application are compared in terms of result and speed in the same matrix values and the distance values are recorded.

4.3. The Evaluation Method

The algorithms developed within the thesis have been evaluated passing through the methods and processes below.

A- Algorithms have been compared with Brute-Force method by using Cities Data Set. In this context the algorithms results and Brute-Force Method results have been compared and Distance Accept Rate values have been calculated.

B- Algorithms have been exposed to speed test in high point numbers by using Random Data Set. Then the speeds of the algorithms have been identified.

C- The algorithms which have been studied according to Distance Rate and Speed Values have been evaluated again and the number of the algorithms has been reduced to 2.

D- Determined two algorithms have been evaluated with more point numbers with each other in terms of both speed and result. After the evaluation the number of the algorithm is reduced to 1.

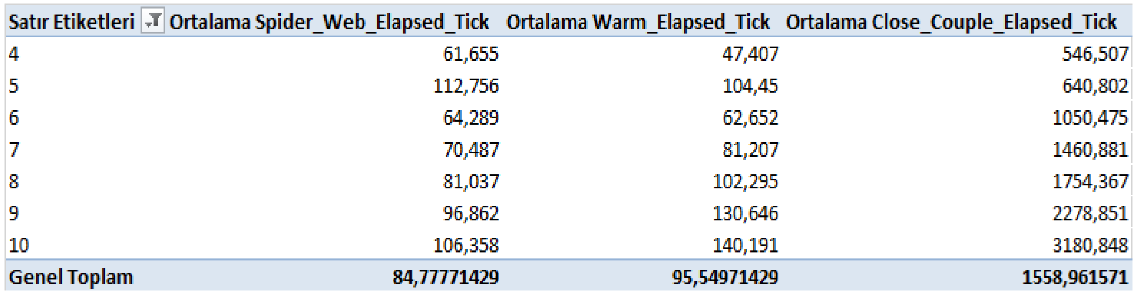
E- Revisions on the determined algorithms with various modifications have been made in terms of speed and result .Then the former and the latter versions have been evaluated and the development has been recorded.

4.4. Evaluation Results

The obtained results according to the evaluation method above are given in this section.

4.4.1 The evaluation of the algorithms according to the brute-force method

The Measurement of the Algorithms Time Tick Values



Satır Etiketleri	Ortalama Spider_Web_Elapsed_Tick	Ortalama Warm_Elapsed_Tick	Ortalama Close_Couple_Elapsed_Tick
4	61,655	47,407	546,507
5	112,756	104,45	640,802
6	64,289	62,652	1050,475
7	70,487	81,207	1460,881
8	81,037	102,295	1754,367
9	96,862	130,646	2278,851
10	106,358	140,191	3180,848
Genel Toplam	**84,77771429**	**95,54971429**	**1558,961571**

Table 9 Time Tick Result Table of the Algorithms

The average durations of the Algorithms on data set have been measured through Time Tick values.

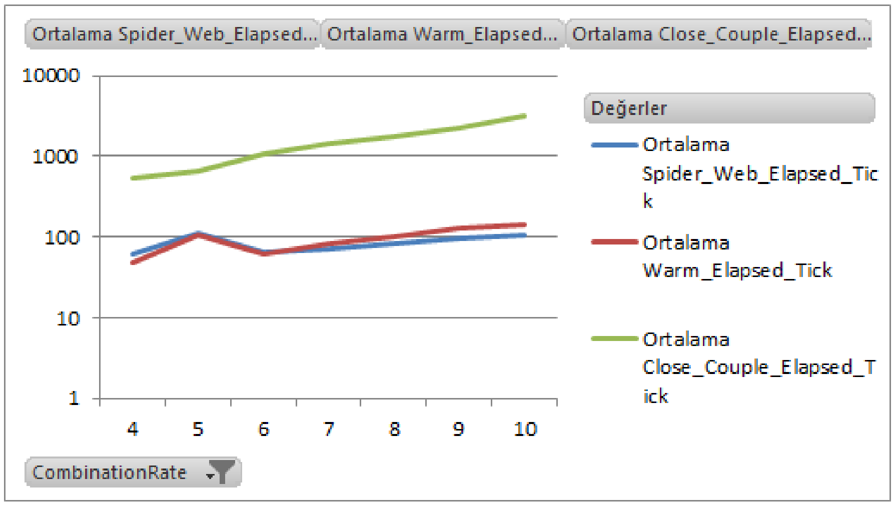


Figure 4 Time tick graphical representation of the algorithms

Spider Web Algorithm has been observed faster than other algorithms according to the Cities Data Set.

The Measurement of the Algorithms Distance Values

Satır Etiketleri	Ortalama Result_Distance_Rate_Spider_Web_Brute_Force	Ortalama Result_Distance_Rate_Warm_Brute_Force	Ortalama Result_Distance_Rate_Close_Couple_Brute_Force
4	1,0027943	1,0225237	1,003988
5	1,0163799	1,0501504	1,0140388
6	1,0282154	1,0696755	1,02313
7	1,0424771	1,0890806	1,0305808
8	1,0552811	1,1065926	1,0410604
9	1,0604559	1,1226016	1,0535207
10	1,0692467	1,1264102	1,0581866
Genel Toplam	**1,039264343**	**1,083862086**	**1,032072186**

Table 10 Distance Rate Result Table of the Algorithms

The distances that the algorithms produced have been measured according to Brute Force Distances. Within this context Distance Rate has been calculated.

$$Distance\ Rate\ =\ Algorithm\ Distance\ /\ Brute-Force\ Distance$$

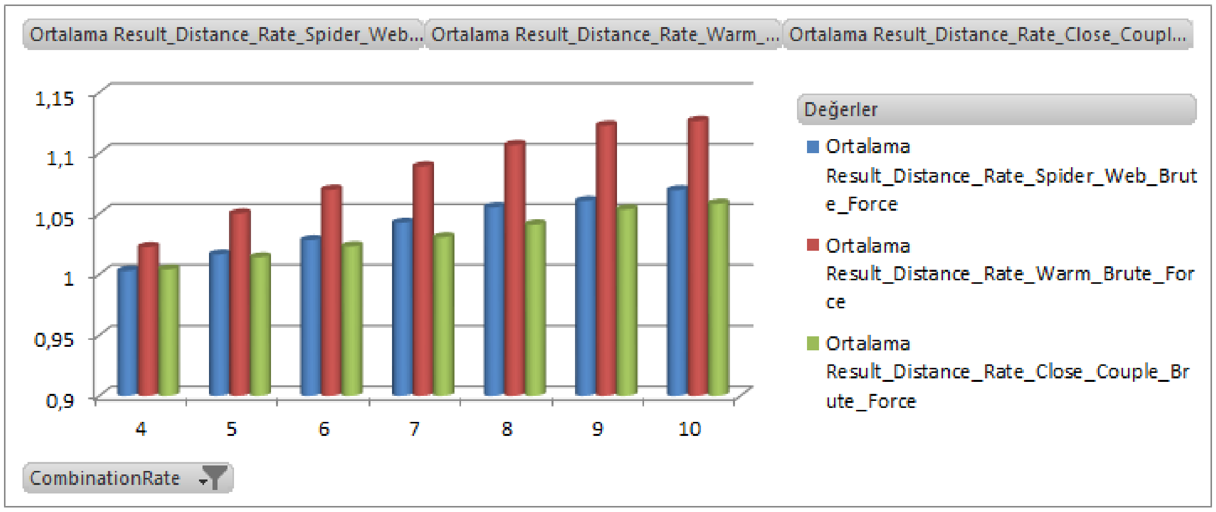


Figure 5 Distance rate graphical representation of the algorithms

When Distance Rate values compared, the values that the Close Couple Algorithm produced it has been observed that they are better than the other algorithms.

The Measurement of the Algorithms Target Values

Satır Etiketleri	Toplam Result_Target_Value_Spider_Web_Brute_Force	Toplam Result_Target_Value_Warm_Brute_Force	Toplam Result_Target_Value_Close_Couple_Brute_Force
4	945	571	914
5	736	384	757
6	601	287	627
7	513	206	520
8	443	155	470
9	393	127	351
10	330	96	293
Genel Toplam	**3961**	**1826**	**3932**

Table 11 Result Table of the Algorithms Target Values

The Target values produced by algorithms have been calculated. For every point number the distance values produced by Brute Force have been compared with the values produced by algorithms and for the values that are same Target Value has been increased by 1. The value of being the same has been found for 1000 sample.

$$Target\ Value = COUNT\ (IF\ (Algorithm\ Distance = Brute\ Force\ Distance))$$

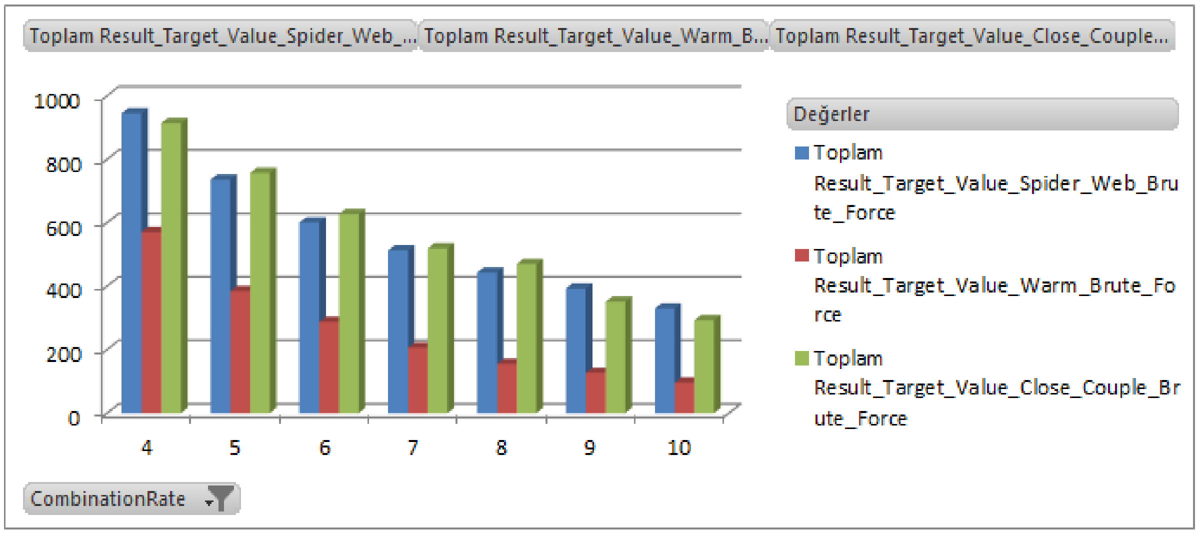
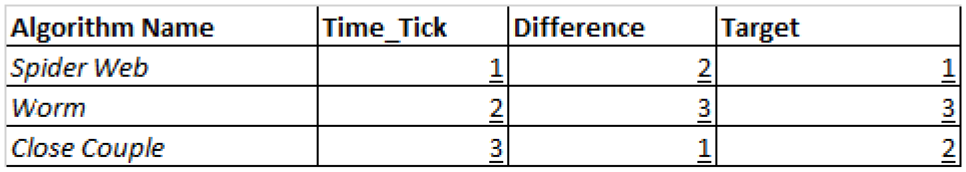


Figure 6 Target values graphical representation of the algorithms

According to the Target Values it has been observed that the values of Spider Web Algorithm are better than other algorithms.

The Evaluation of the Algorithms According to the Brute-Force Method



Algorithm Name	Time_Tick	Difference	Target
Spider Web	1	2	1
Worm	2	3	3
Close Couple	3	1	2

Table 12 The Table of the Evaluation of the Algorithms With Data Set

When the values obtained by algorithms are compared with the values obtained by Brute Force Method;

A. In terms of Distance Values, it has been observed that Close Couple Algorithm produced better result when compared with other algorithms,

B. In terms of Target and Speed Values, it has been observed that Spider Web Algorithm produced better result when compared with other algorithms.

Within the scope of these results it has been decided to work through Spider Web and Close Couple Algorithms for the evaluation process via Random data set.

4.4.2 The evaluation of the algorithms with random data set

The duration and the results of Spider Web and Close Couple Algorithms have been compared using random matrix that has been produced.

The Comparison of the Algorithms Durations

Satır Etiketleri	Ortalama Spider_Web_Elapsed_Time	Ortalama Close_Couple_Elapsed_Time
4	0,144	0,479
8	0,031	0,179
16	0,034	8,318
32	0,005	113,723
64	0,322	1757,434
128	4,292	25490,964
Genel Toplam	**0,804666667**	**4561,8495**

Table 13 Elapsed Time Result Table of the Algorithms

Spider Web and Close Couple Algorithms have been compared in terms of the passing time. The average duration of the Close Couple Algorithm has been identified as $O\ (n^4)$ according to the values obtained as a result of the comparison.

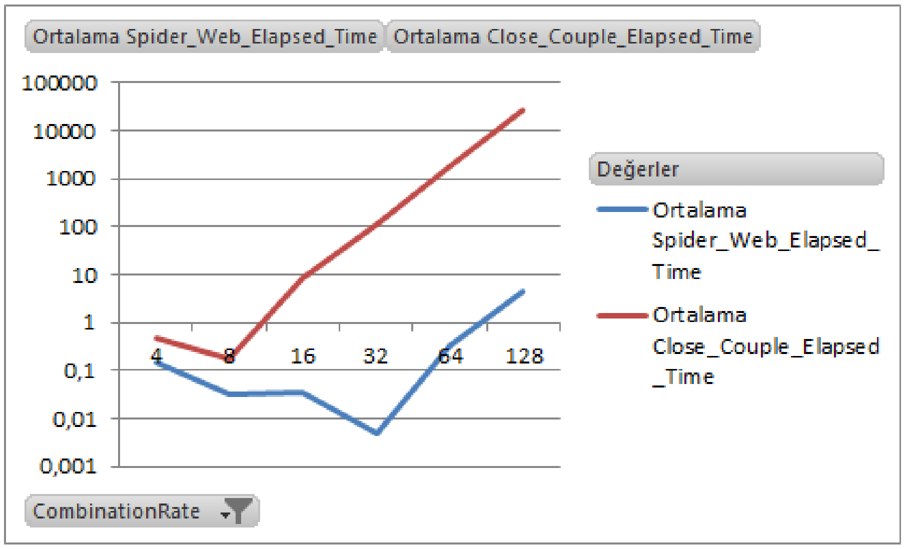


Figure 7 Elapsed time graphical representation of the algorithms

When it is evaluated in terms of the passing time, it has been observed that Spider Web Algorithm is faster than Close Couple Algorithm.

The Comparison of Algorithms Distances

Satır Etiketleri	Ortalama Spider_Web_Distance	Ortalama Close_Couple_Distance
4	1890,832	1890,965
8	2773,508	2760,974
16	3867,129	3814,249
32	5545,529	5444,902
64	8194,974	7822,868
128	12094,782	11290,398
Genel Toplam	**5727,792333**	**5504,059333**

Table 14 Distance Value of the Algorithms Result Table

Algorithms have been compared according to the distance values which have been obtained according to the point numbers.

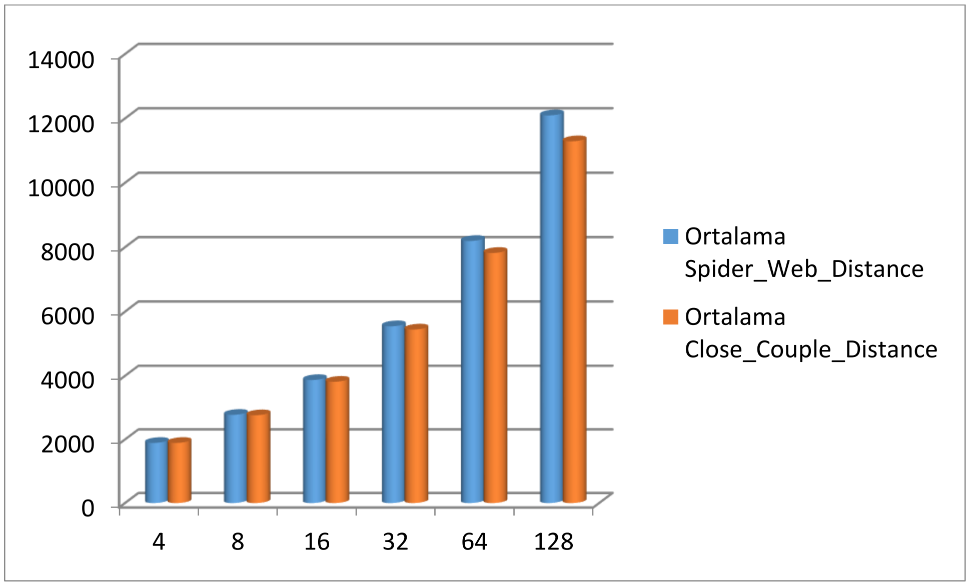


Figure 8 Distance values graphical representation of the algorithms

According to the distance that has been obtained, it has been observed that Close Couple Algorithm has produced better results.

The Evaluation of the Algorithms with Random Data Set

When the values obtained by algorithms compared with each other on Random Data Set, it has been observed that;

A- In terms of speed values Spider Web Algorithm was 5700 times faster,

B- In terms of distance values that have been produced, Close Couple Algorithm % 4 better values than Spider Algorithm.

According to these results for development and revision of the algorithm it has been decided to work on Spider Web Algorithm.

4.4.3 The development of the Spider Web Algorithm and evaluation of the results

Spider Web Algorithm has been evaluated again in terms of speed and values and algorithm has been improved by various changes and updates. Within this context Spider Web Second edition (SW2) and Spider Web third edition (SW3) algorithms have been developed. The newly developed algorithms have been compared.

The Comparison of the Algorithms Duration with Brute - Force Method

Satır Etiketleri	Ortalama Spider_Web_Elapsed_Tick	Ortalama Spider_Web_Second_Elapsed_Tick	Ortalama Spider_Web_Third_Elapsed_Tick
4	61,655	127,589	159,139
5	112,756	58,749	71,678
6	64,289	61,97	79,918
7	70,487	66,82	90,121
8	81,037	86,091	102,275
9	96,862	85,46	126,771
10	106,358	108,01	123,703
Genel Toplam	**84,77771429**	**84,95557143**	**107,6578571**

Table 15 Time Tick Result Table of the Spider Web Algorithm

The passing time Spider Web Algorithm has been compared via Time Tick values.

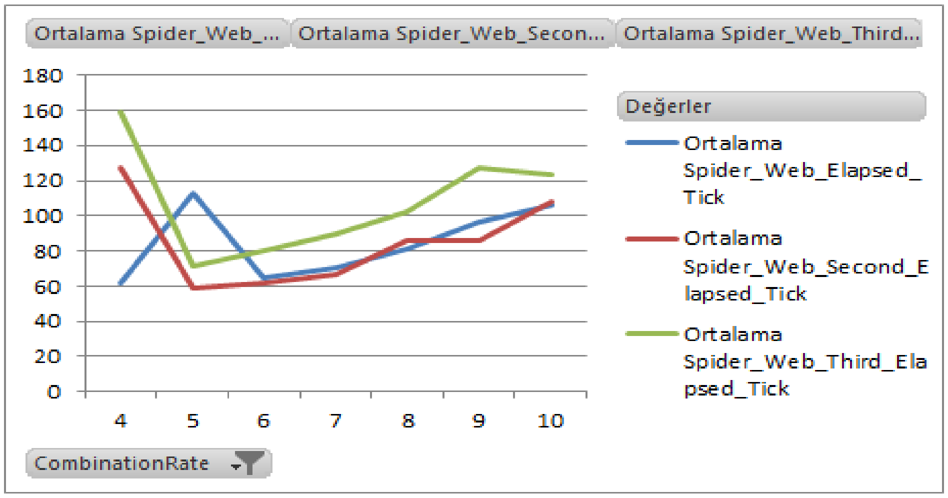
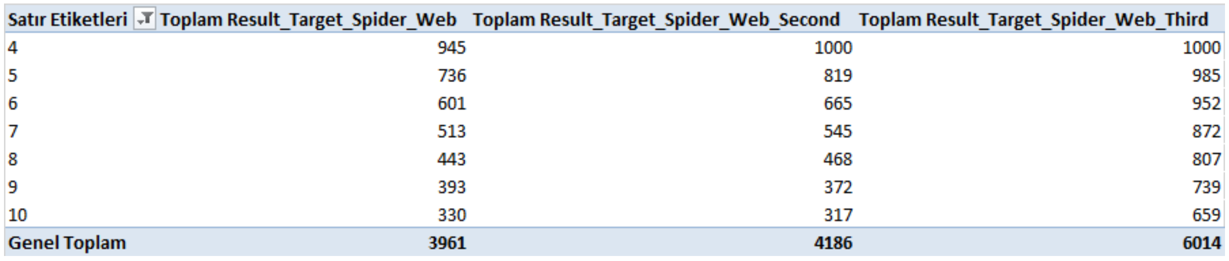


Figure 9 Graphical representations of the Spider Web Algorithm time tick values

According to Bruce Force Data Set it has been observed that Spider Web Algorithm is faster than other algorithms. It has been observed that the developed Spider Web third edition algorithm is % 27 slower.

The Comparison of the Algorithms Target Values via Brute – Force Method



Satır Etiketleri	Toplam Result_Target_Spider_Web	Toplam Result_Target_Spider_Web_Second	Toplam Result_Target_Spider_Web_Third
4	945	1000	1000
5	736	819	985
6	601	665	952
7	513	545	872
8	443	468	807
9	393	372	739
10	330	317	659
Genel Toplam	**3961**	**4186**	**6014**

Table 16 Spider Web Algorithm Target Resut Table

The developed agorithms have been compared according to Target values.

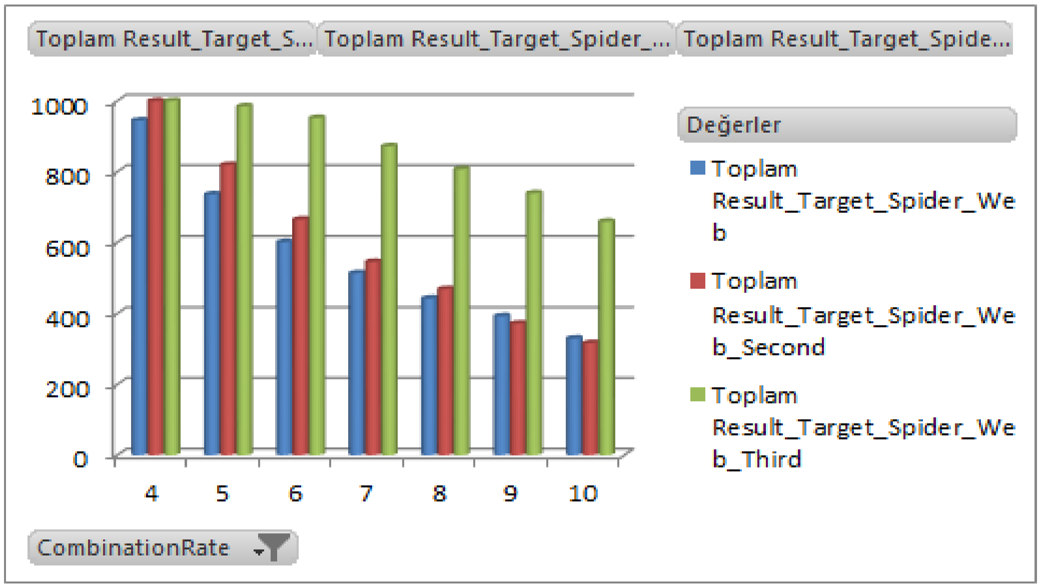


Figure 10 Graphical representation of the Spider Web Algorithm target values

% 51 improvement has been observed on the Target values of the developed Spider Web Third Edition Algorithm.

The Comparison of the Algorithms Distance Rate Values via Brute – Force Method

Satır Etiketleri	Ortalama Result_Distance_Rate_	Ortalama Result_Distance_Rate_Spider_Web_Second_	Ortalama Result_Distance_Rate_Spider_Web_Third_Brute_Force
4	1,0027943	1	1
5	1,0163799	1,0055671	1,0000562
6	1,0282154	1,0096207	1,0008291
7	1,0424771	1,0154497	1,0037574
8	1,0552811	1,0204048	1,0063955
9	1,0604559	1,0229865	1,0089291
10	1,0692467	1,0273579	1,0121528
Genel Toplam	**1,039264343**	**1,014483814**	**1,004588586**

Table 17 Distance Rate Result Table of Spider Web Algorithm

The developed algorithms have been compared according to the Distance Rate values.

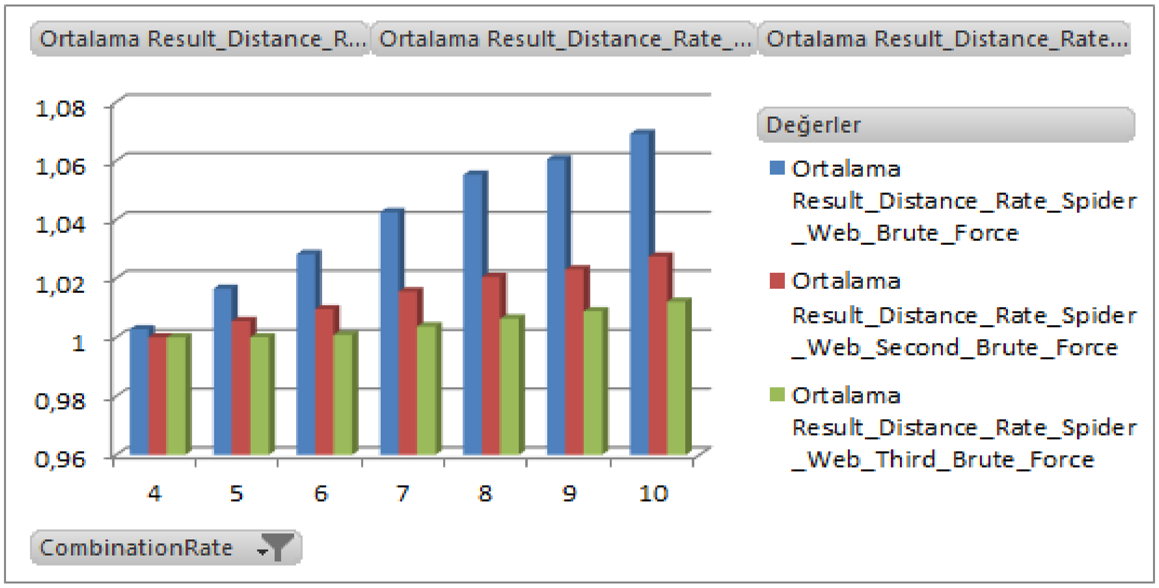


Figure 11 Graphical representation of the Spider Web Algorithm distance rate values

It has been observed that Spider Web Third Edition Distance Rate Value has reached 1.004 over the average values.

The Comparison of the Algorithms Duration

Satır Etiketleri	Ortalama Spider_Web_Elapsed_Tick	Ortalama Spider_Web_Second_Elapsed_Tick	Ortalama Spider_Web_Third_Elapsed_Tick
4	115,632	22474,135	86,201
8	73,141	71,399	123,562
16	169,981	158,074	270,909
32	517,775	491,42	792,18
64	1808,381	1444,248	2392,525
128	7192,05	6201,322	9440,814
256	39332,755	31111,618	41171,021
512	182187,708	153119,128	180938,472
Genel Toplam	**28924,67788**	**26883,918**	**29401,9605**

Table 18 Spider Web Algorithm Elapsed Tick Result Table

The developed algorithms have been compared with Time Tick values via Random Data Set. The average duration of the Spider Web Algorithms has been identified as $O\ (n^2)$ as a result of the comparison according to the acquired values.

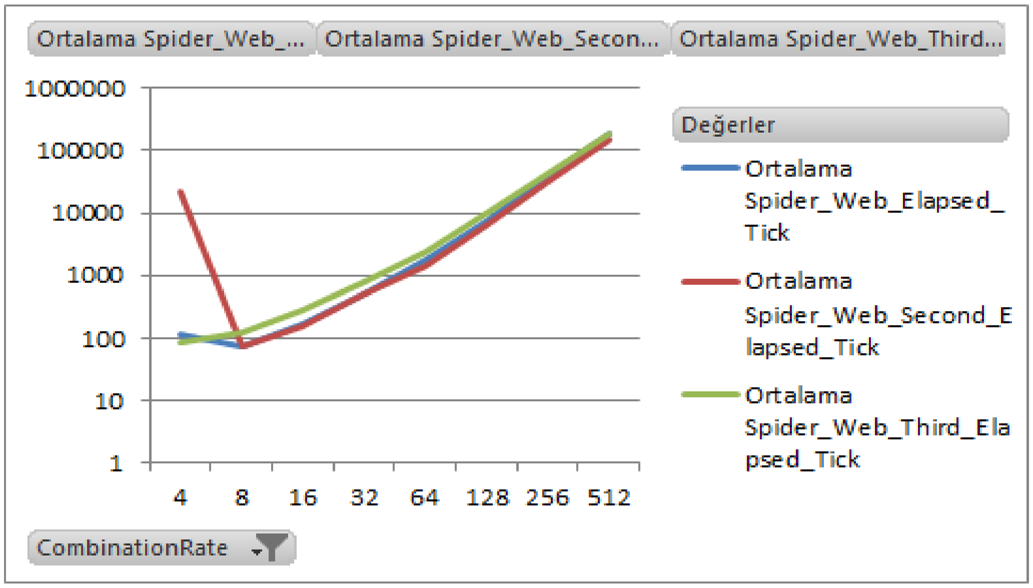
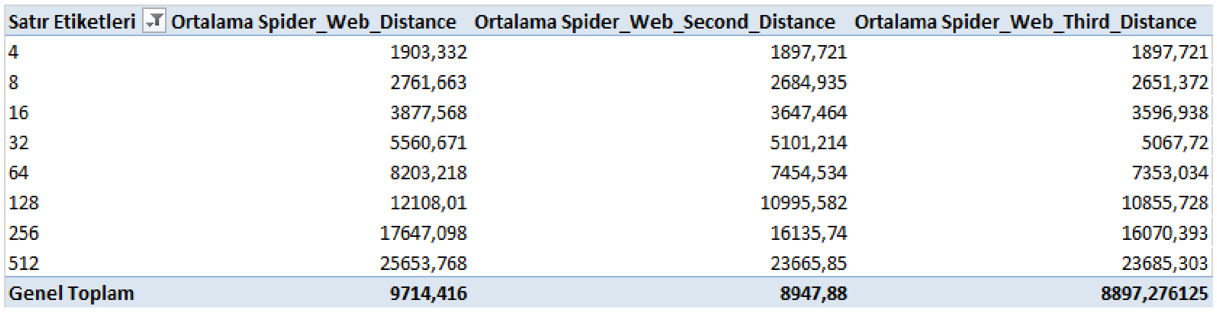


Figure 12 Graphical representation of the Spider Web Algorithm elapsed tick

It has been observed that Spider Web Third Edition algorithm is % 1 slower than other algorithms.

The Comparison of the Algorithms Distance



Satır Etiketleri	Ortalama Spider_Web_Distance	Ortalama Spider_Web_Second_Distance	Ortalama Spider_Web_Third_Distance
4	1903,332	1897,721	1897,721
8	2761,663	2684,935	2651,372
16	3877,568	3647,464	3596,938
32	5560,671	5101,214	5067,72
64	8203,218	7454,534	7353,034
128	12108,01	10995,582	10855,728
256	17647,098	16135,74	16070,393
512	25653,768	23665,85	23685,303
Genel Toplam	**9714,416**	**8947,88**	**8897,276125**

Table 19 Spider Web Algorithm Distance Result Table

The algorithms developed have been compared according to the distances that they have produced.

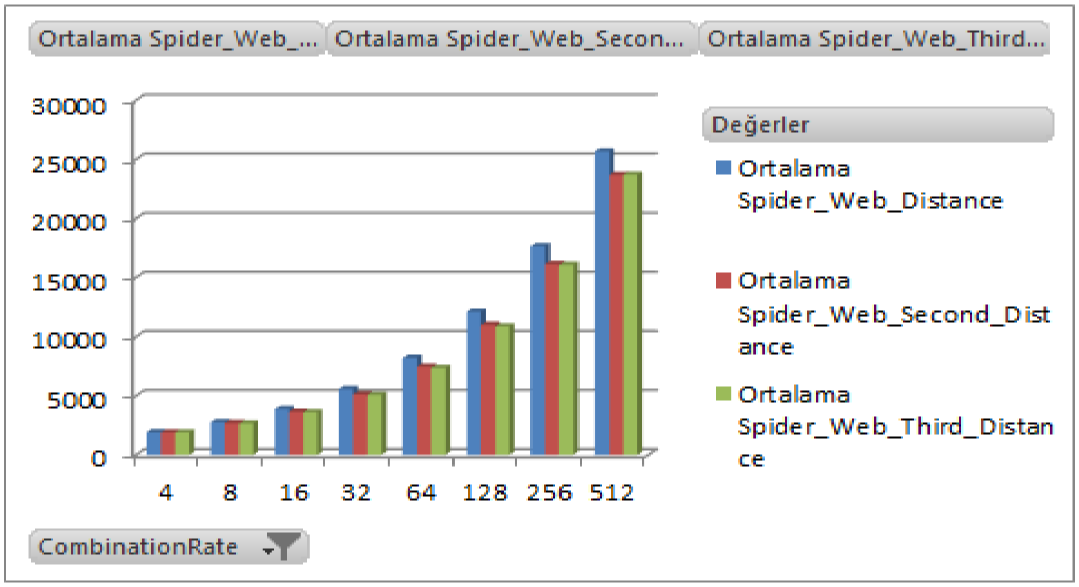


Figure 13 Graphical representation of the Spider Web Algorithm distance

It has been observed that % 9 better results have been acquired with Spider Web Third Edition algorithm over distance values.

CHAPTER 5

CONCLUSION AND FUTURE WORK

5.1 Conclusion

This thesis aimed to find the optimum solution of TSP problem. Three different algorithms are proposed and developed with different perspectives. Both Close Couple and Worm Algorithms search the optimum solution, starting from the closest points. Spider Web Algorithm tries to reach the optimum solution with the reference data that it has acquired from the farthest points.

The developed algorithms have been evaluated with two different data sets. In the first data set (cities in Turkey) the distances between them have been used as database. The second data set has been formed by using random value intervals within identified matrix size and the speeds of the algorithms and the quality of the results that produced have been compared. Also finding the optimum and the fastest result of the symmetrical TSP problem are achieved.

A progressive evaluation has been used for improvement and revision of the algorithms. In the first stage, the results of all algorithms have been compared by using City Data Set and Brute-Force Method. As a result of this comparison, in terms of Distance Rate Close Couple algorithm and in terms of speed Spider Web Algorithm gained better score than Worm Algorithm. After this stage Worm Algorithm has been eliminated.

In the second stage, using Random Data Set. The performance of Close Couple and Spider Web Algorithms has been measured via high matrix and point numbers .Within this context while Close Coupe produces %4 better result, Spider Web Algorithm has been produced 5700 times faster than Close Couple Algorithm. After this stage Close Couple Algorithm has been eliminated.

In the third stage optimization of Spider Web Algorithm (SW1) has been done. Within this context (SW2) and (SW3) algorithms have been tested with Spider Web algorithms. In this stage the developed Spider Web algorithms have been evaluated through both Cites Data Set and Random Data Set. At this point, Spider Web Third Edition (SW3) with its low distance rate and high Target Value on the Random Data Set has given better results than other versions.

According to the Random Data Set results Spider Web Third Edition Algorithm (SW3) has produced 9% better results compared with Spider Web First Edition Algorithm (SW1) but only   1% slower than Spider Web First Edition Algorithm (SW1). Spider Web Third Edition algorithm has been formed with different data sets and gradual evaluation.

5.2. Future Work

We plan to:

- Use the algorithm in a Discrete and Combinatorial Optimization Library (TSPLIB).
- Implement same algorithm on Asymmetric TSP.
- Adopt for the Time-Dependent-Distance problems in order to be use for the purpose of solving the problems like day time deliveries and traffic jam in transportation sector.

REFERENCES

1. **Renaud J., Boctor F. F., Ouenniche J., (2000),** *"A Heuristic for the Pickup and Delivery Traveling Salesman Problem"*, Computers & Operations Research vol 27, pp. 905-916.

2. **Lin S., Kernighan B. W., (1973),** *"An Effective Heuristic Algorithm for the Traveling-Salesman Problem"*, Operation Research., vol 52, pp. 498-516.

3. **Menger K., (1932),** "*Das Botenproblem"*, Ergebnisse Eines Mathematischen Kolloquiums. Teubner, Leipzig, pp.11–12.

4. **G. Gutin Yeo A., Zverovich A., (2002),** *"Traveling Salesman should not be Greedy: Domination Analysis of Greedy-type Heuristics for the TSP"*, Discrete Applied Mathematics, pp. 81–86.

5. **Lawler E.L., Lenstra J.K., Rinnooy Kan A.H.G., Shmoys D.B., (1985),** "*The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*", Wiley, New York, pp. 87-143.

6. **Helsagun K., (2000),** *"Theory and Methodology An Efective Implementation of the Lin-Kernighan Traveling Salesman Heuristic"*, European Journal of Operational Resarch, vol 126, pp. 106-130.

7. **Padberg, M.W., Rinaldi G., (1991),** *"A Branch-and-cut Algorithm for the Resolution of Large-scale Symmetric Traveling Salesman Problems"*, SIAM Review 33, pp. 60-100.

8. **Applegate D., Bixby R. Chvatal V., Cook W., (1995),** "*Finding cuts in the TSP (A preliminary report)*", DIMACS, Tech. Report 95-05, pp. 15-26.

9. **Bentley J.L., (1990),** *"Experiments on Traveling Salesman Herustics",* SIAM Philadelphia, PA, pp. 91-99.

10. **Nilsson C., (2003)***, "Heuristics for the Traveling Salesman Problem"*, Techical Report, Linköping University, Sweden, pp. 2-4.

11. **Jünger M., Reinelt G., Rinaldi G., (1994),** *"The Traveling Salesman Problem",*Wiley,New York, pp 199-221.

12. **Hoffman K. L., Padberg M., Rinaldi G., (2001),** *"Traveling Salesman Problem*", Kluwer Academic Publishers, pp. 5-6.

APPENDICES A

CURRICULUM VITAE

PERSONAL INFORMATION

Surname, Name: Kaya, Ahmet Sedat

Date and Place of Birth: 23 March 1978, Ankara

Marital Status: Married

Phone: +90 533 205 41 89

Email: ahmetsedatkaya@gmail.com

EDUCATION

Degree	**Institution**	**Year of Graduation**
M.Sc.	Çankaya University, Computer Engineering	2013-…
B.A.	Anadolu University, Public Administration	2005
Two Year Degree	Kırıkkale University, Computer Programming	1998

WORK EXPERIENCE

Year	**Place**	**Enrollment**
2014- Present	TÜBİTAK.	Specialist
2011-2014	Natek	Software and Quality Manager
2009-2011	Stratek	Software Engineer
2007-2009	PDI-Erkom.	Software Manager
2004-2007	Birsistem.	IT Director
1999-2004	Siemens Business Services.	Software Specialist

FOREIN LANGUAGES

English

PROJECTS

1. Common Criteria EAL Level 3 for Natek NAC Product, 2014.
2. CMMI-Dev v.1.3 Maturity Level 3 for Natek, 2012.
3. Ext JS Framework, 2011.
4. Management Project, (Ministry of Finance), 2010
5. SGB.Net, (Ministry of Finance), 2010
6. Seed Project, (European Union Project, Ministry of Agriculture), 2009
7. E-Sign Implementation (EJBCA Enterprise class PKI Certificate Authority), 2008
8. Data Center Project (Ministry of Industry), 2006
9. BI2PRS (Business Intelligence for Product Related Services), 2004
10. WRS(Web Reporting System), 2003
11. RMS(Request Management System), 2002
12. Halkbank VOIP Project, 2000

HOBBIES

Reading, Research, Travel