International Journal of Engineering and Information Systems (IJEAIS)
ISSN: 2643-640X
Vol. 7 Issue 12, December - 2023, Pages: 133-138

Some solution method of TSP

Aseel Sami Hamd [1], Zainb Mahmood Shakeer[2], Ali Hasan Abdoul Khaleq[3], Ahmed Hadi Hussain[4]

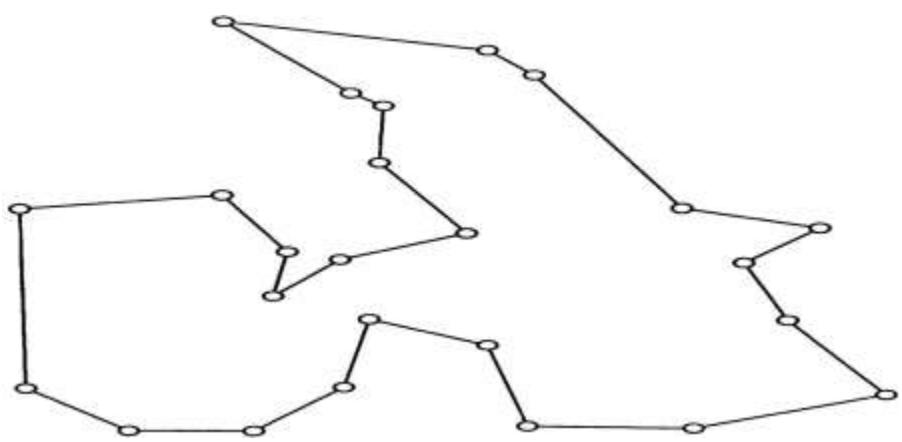
1Mathematics Department, College of Education for pure Sciences, University of Babylon. Babylon, Iraq.
pure.aseel.najm@uobabylon.edu.iq
3Mathematics Department, College of Education for pure Sciences, University of Babylon. Babylon, Iraq.
edu716.ali.hasan@uobabylon.edu.iq
2Mathematics Department, College of Education for pure Sciences, University of Babylon. Babylon, Iraq.
pure.zainab.hashim@uobabylon.edu.iq,
4Department of Automobile Engineering, College of Engineering Al-Musayab, University of Babylon. Babylon, Iraq
met.ahmed.hadi@uobabylon.edu.iq

Abstract: *traveling salesman problem one of the complex algorithmic problems that has attracted the interest of many computer scientists and mathematicians because of its importance in our real lives and for increasing the profitability of companies. Many algorithms have been developed to find the best possible route with the least time and cost for its route when it passes through a number of cities and returns to the city from which it started.*

Keywords. traveling salesman problem, approximation, Brute Force Method, Branch and Bound, nearest neighbor, Greedy.

1. Introduction.

The TSP aroused the interest of scientists, as it was developed by the scientist William Hamilton in the nineteenth century, who is considered the inventor of the icosian game 1857. It was represented by a board hung with twenty holes, based on visiting each peak once while not passing any edge twice or more. where The end point is the same as the starting point. It was developed with the scientist Thomas kikman , and karl menger and Harvard studied it in the 1930s. Then scientists solved the TSP problems in different cities, including {Applegate, Bixby, Chvatal and Cook 1994 (7397 cities), in 1998 (solved 13509 cities), in 2001 (15112 cities in Germany), in 2004 (Problem 24978 has been solved for all cities in Sweden) } until the largest TSP was found, amounting to 72,500 cities. [4], [3]



2. Exact Solution.

2.1. Brute Force Method.

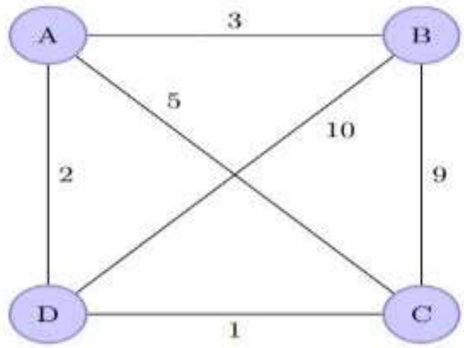
A direct method based on identifying the problem and defining its relevant concepts. It be useful for solving small-sized problems. Also known as naive approach.

To solve TSP by using brute force method :

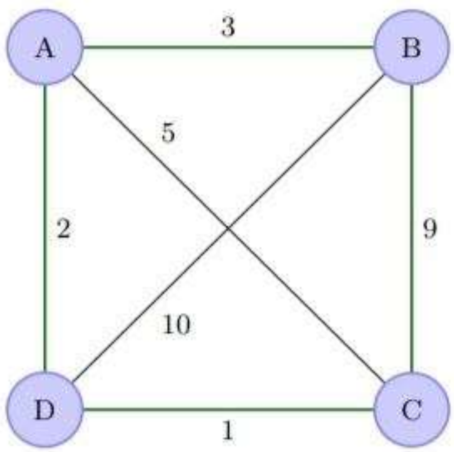
It is based on determining the size of the input with a number of elements, let it be n, calculate and compare the possible path permutations to determine the shortest solution and the total number of paths, draw all possible paths and include them calculate the distance of each path and choose the shortest one, to be the optimal solution.

International Journal of Engineering and Information Systems (IJEAIS)
ISSN: 2643-640X
Vol. 7 Issue 12, December - 2023, Pages: 133-138

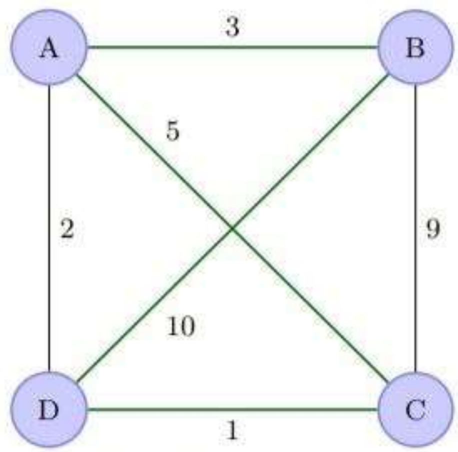
2.1.1. Example for Brute Force Method. [1]



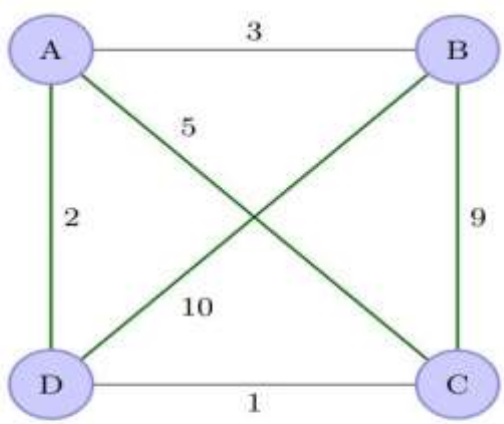
It contains 4 nodes, the following 3 paths will be possible .



$A \rightarrow B \rightarrow C \rightarrow D \rightarrow A = 15$



$A \rightarrow B \rightarrow D \rightarrow C \rightarrow A = 19$



$A \rightarrow C \rightarrow B \rightarrow D \rightarrow A = 26$

The best distance path is $A \rightarrow B \ \rightarrow C \rightarrow D \rightarrow A$, of value 15.

2.2. Branch and Bound.

International Journal of Engineering and Information Systems (IJEAIS)
ISSN: 2643-640X
Vol. 7 Issue 12, December - 2023, Pages: 133-138

This algorithm is one of the effective methods for tackling NP hard optimization problems, including the TSP, Its applications have been noted in several references [9],[10],[2],[6]

To solve TSP by using Branch and Bound:

It begins by creating an initial path, to be from the starting point to the first vertex or node in the set of cities, and then systematically explores the path for different permutations to expand the path to one node at a time, while calculating the algorithm. The length of the current path compared to the optimal path that was found previously. If the length of the current path is observed to be more than the optimal path, then this branch is reduced because it does not harm the optimal solution. This reduction represents the key to making the algorithm effective by investigating ineffective paths to narrow the search space while continuing the process to reach the shortest path That represents the optimal solution.

2.2.1.Example for Branch and Bound

Cost Matrix =

∞	**20**	**30**	**10**	**11**
15	∞	**16**	**4**	**2**
3	**5**	∞	**2**	**4**
19	**6**	**18**	∞	**3**
16	**4**	**7**	**16**	∞

- draw the space with the cost of ideal reduction.
- We derive the cost of the path from node i to j by assigning all entries in row i^{th} and column j^{th} as ∞.

$$M\,[j][i] = \infty$$

- the cost of the corresponding node N =

sum of optimal cost + reduction cost + $M\,[j][i]$,

- for not reducing the given matrix, We first find a reduced matrix for the row and then a reduced matrix for the column if we need to.
- We reduce the cost matrix by subtracting the minimum value from each row and column,
-

∞	**20**	**30**	**10**	**11**	→ **10**
15	∞	**16**	**4**	**2**	→ **2**
3	**5**	∞	**2**	**4**	→ **2**
19	**6**	**18**	∞	**3**	→ **3**
16	**4**	**7**	**16**	∞	→ **4**

$\Longrightarrow$

∞	**10**	**20**	**0**	**1**	
13	∞	**14**	**2**	**0**	
1	**3**	∞	**0**	**2**	**= $\acute{M}_1$**
16	**3**	**5**	∞	**0**	
12	**0**	**3**	**12**	∞	

↓ ↓ ↓ ↓ ↓

1 0 3 0 0

$\boldsymbol{M_1'}$

- so as not to reduce it as well, we subtract the minimum value from the corresponding column,

∞	**10**	**17**	**0**	**1**	
12	∞	**11**	**2**	**0**	
0	**3**	∞	**0**	**2**	= M_1
15	**3**	**12**	∞	**0**	
11	**0**	**0**	**12**	∞	

Cost of $M_1 = C(1)$.

= row reduction cost + column reduction cost.

- then the length of all rounds is at least 25 and this is the ideal cost. For the track.

3. Approximation solution.

As a result of it taking a long time to optimally solve the TSP problem, the use of the approximation algorithm gives us a guarantee of obtaining solutions. The Arora [8] algorithm is considered the best approximation algorithm within approximation$(1 + 1 = c\ ), \forall c > 1$ and it seeks geometric division, although c is very large in theory, but it has a negative impact on the time Turn it on $(O\left(n\ (\log_2 n\ )^{O(c)}\right)$ for two- dimensional problem instances).

3.1. nearest neighbor.

This algorithm be most clear inference for TSP, as it always depends on visiting the nearest city. It is based on decision-making, as the seller must, during his entire trip, make only two decisions before starting the trip, one of which is specifying which city he will start with, so that its importance appears in the different results that the beginning of each trip leads to. One, and the second must move from one city to another nearest city without visiting the next city.

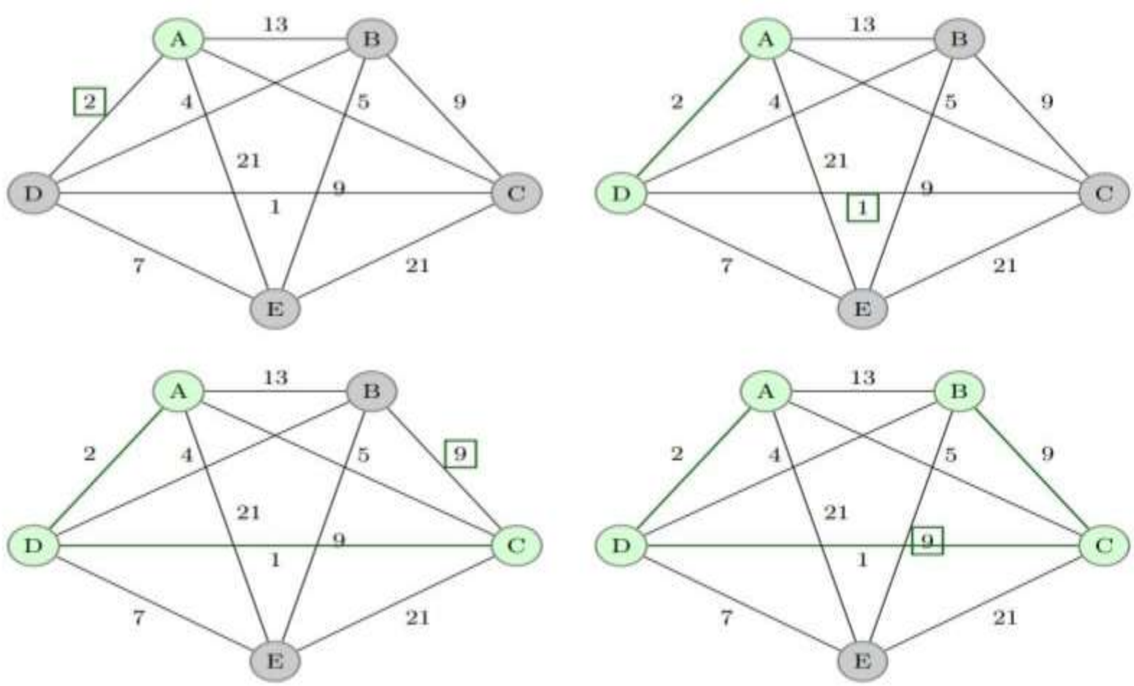
Nearest Neighbor, $O(n^2)$.

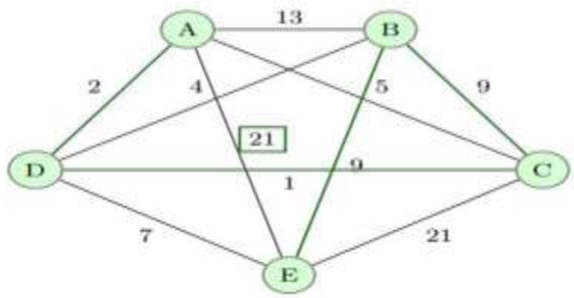
3.1.1. example of nearest neighbor. [1]

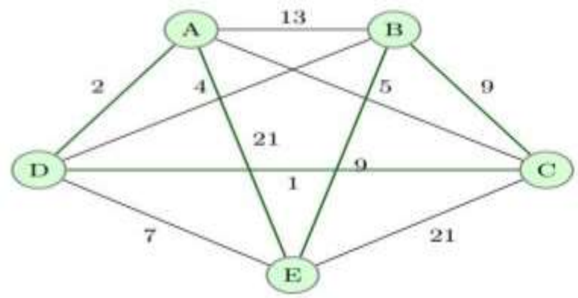
The approximate solution is represented using this method.

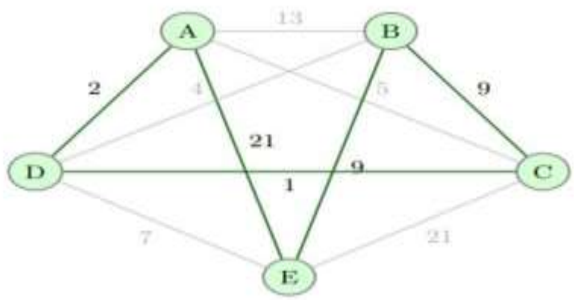
Here we have 5 nodes. we start with the node A to implement this

algorithm.









The total distance of the path $A \rightarrow D \rightarrow C \rightarrow B \rightarrow E \rightarrow A$, obtained

using this algorithm is 42.

3.2. Greedy.

It be considered the simplest optimization algorithm. The algorithms depend on making a local optimal choice to reach the globally optimal option. Most of the time, it may not lead to the optimal solution, but it does lead to locally optimal solutions.

Greedy, $O(n^2 \log_2(n))$.

Algorithm. [5]

1: set $s_1 \in V$ as the first center (arbitrary)

2: $S \leftarrow \{s_1\}$

3: for $i = 1,2, \dots, k$

. $s_i \leftarrow arg\ max_{u \in V}\ (min_{s \in \{s_1, \dots, s_{i-1}\}}\ d(s,u))$

. $S \leftarrow S \cup \{s_i\}$

4. Conclusion.

In conclusion, we noticed the multiplicity of algorithms to reach the optimal solution in the shortest distance, least time, and cost to increase the benefit of consuming companies and facilitate the seller's movement process with the least possible effort, as each algorithm appeared unique with its own steps. It was noted that exact algorithms guarantee finding an accurate and optimal solution to the problem regardless of time, unlike The greedy algorithm is designed to work quickly, choosing a locally optimal solution each time to get closer to the globally optimal solution. Finding an optimal solution for NP hard heuristically will require comparing solutions using computational experiments.

References.

[1] Amanur Rahman Saived, The Traveling Salesman problem, Indiana State University Terre Haute, IN 47809, USA ,April 11, 2012

[2] Archit Rastogi, Ankur kumar shrivastava,et.all, A Proposed Solution to Travelling Salesman Problem using Branch and Bound, International Journal of Computer Applications (0975 –8887) Volume 65–,pp. 44-49 2013.

[3] Hoffman, Karla L., Manfred Padberg, and Giovanni Rinaldi. "Traveling salesman problem." Encyclopedia of operations research and management science 1 (2013): 1573-1578.

[4] Jünger, Michael, Gerhard Reinelt, and Giovanni Rinaldi. "The traveling salesman problem." Handbooks in operations research and management science 7 (1995): 225-330.

[5] Karmel Shehadeh, Viswanath Nagarajan, greedy Algorithm:TSP and K- center, (2017).

[6] Leo Liberti, Branch- and Bound for the Travelling Salesman Problem, France, March, (2011)

[7] Nogales Giné, Roger. Different approaches to Travelling Salesman Problem. (2022).

[8] S. Arora, Polynomial Time Approximation Schemes for Euclidian Traveling Salesman and Other Geometric Problems, Journal of the ACM, Vol. 45, No. 5, September 1998, pp. 753-782.

[9] Tucker, A.. Applied Combinatorics. Wiley, New York, 2002

[10] Winston, W. Operations Research_ Applications and Algorithms. Duxbury, Belmont, CA. 2004.