KRUSKAL'S MINIMUM SPANNING TREE ALGORITHM

INTRODUCTION

- Kruskal's algorithm is used to find the minimum spanning tree of a weighted graph [1]
- The algorithm selects edges based on their weight on an increasing order and only adds them if they don't create a cycle.[2]
- Formally, Kruskal's algorithm maintains a forest—a collection of trees. When adding an edge, it merges two trees into one[1]
- The algorithm terminates when the number of edges equal to V-1, where V is the number of vertices. There is only one tree, and this is the minimum spanning tree[2]

EXAMPLE

Undirect weighted graph, where:
(1, 2, 6), (1, 3, 5), (2, 4, 2), (2, 5, 5), (3, 5, 6), (3, 6, 8), (4, 5, 9), (4, 6, 10) (5, 6, 12)
expect: V-1 = 5 edges

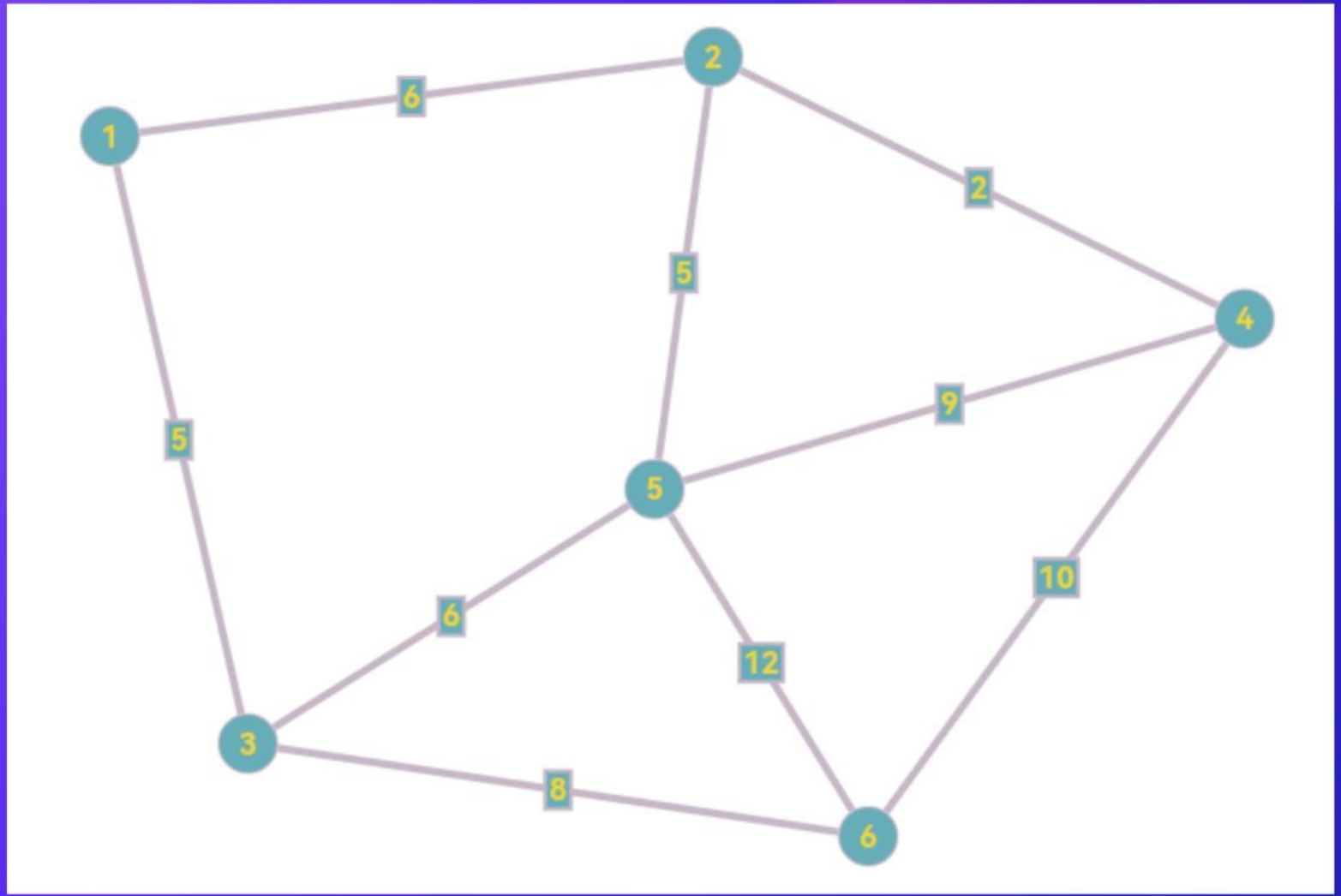


Fig 1. “An example of a weighted graph that consists of 6 nodes and 9 edges” [3]

STEP 1:

- Choose the smallest edge, which is edge 2 from node 4 to node 2.

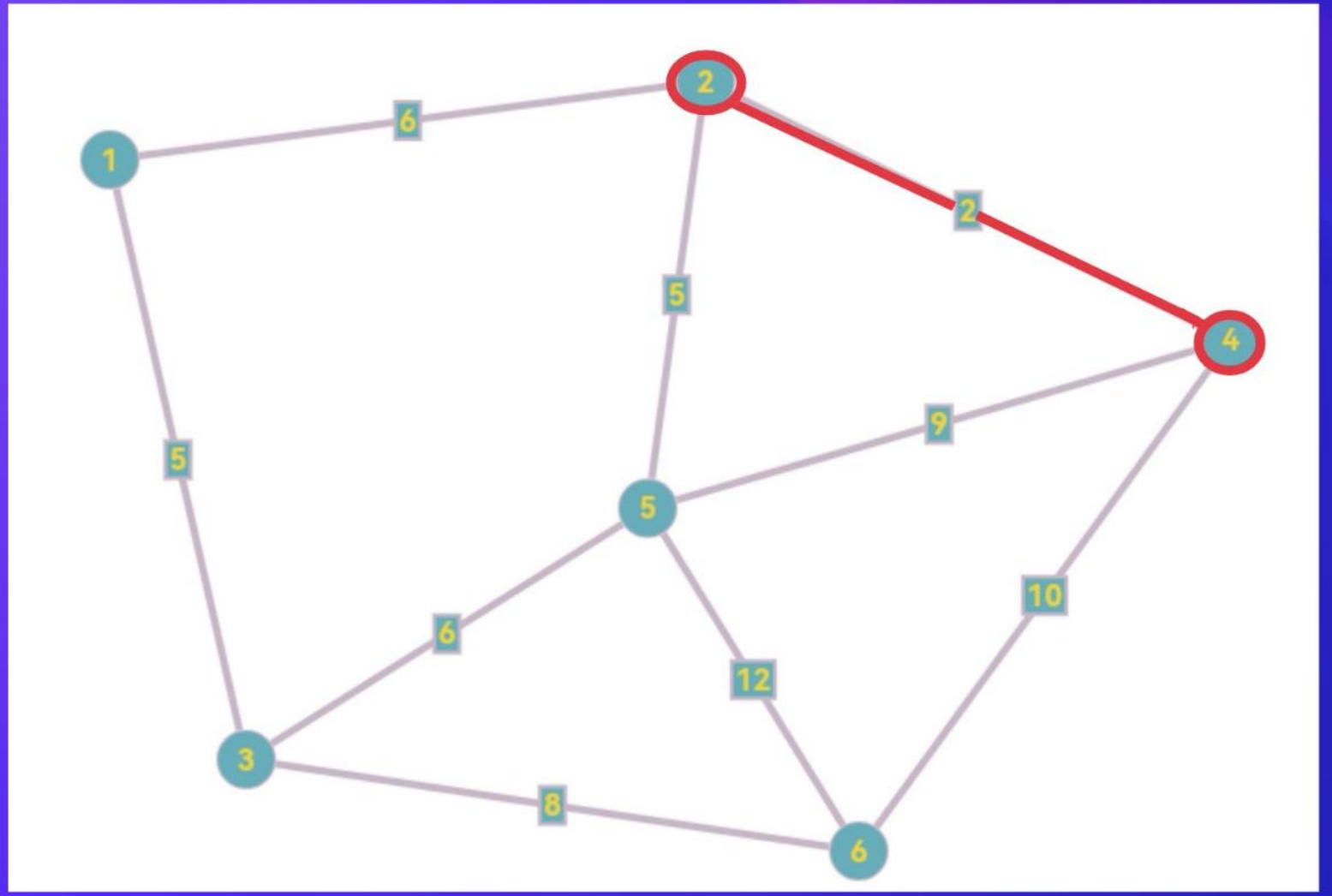


Fig 2. “Applying Kruskal’s MST step: 1 on the example graph” [3]

STEP 2 & 3:

- Choose the next smallest edge, which is edge 5
- In this case, there are two edges: from node 2 to node 5 and from node 1 to node 3

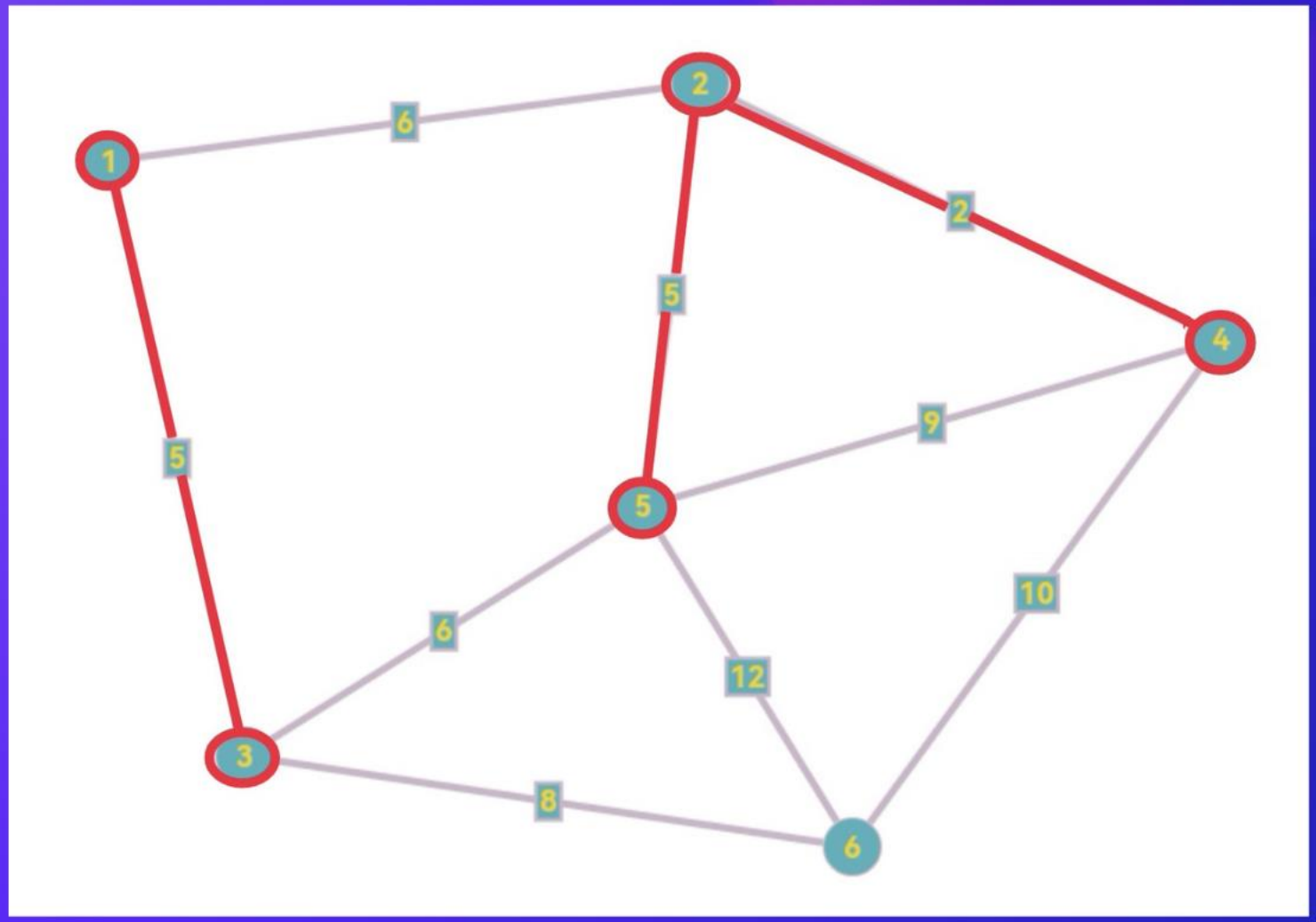


Fig 3. “Applying Kruskal’s MST step: 2 and 3 on the example graph” [3]

STEP 4:

- Choose the next smallest edge, which is edge 6.
- In this case, the algorithm randomly chooses the edge from node 2 to node 1
- Notice: it cannot choose both since a tree cannot have a cycle

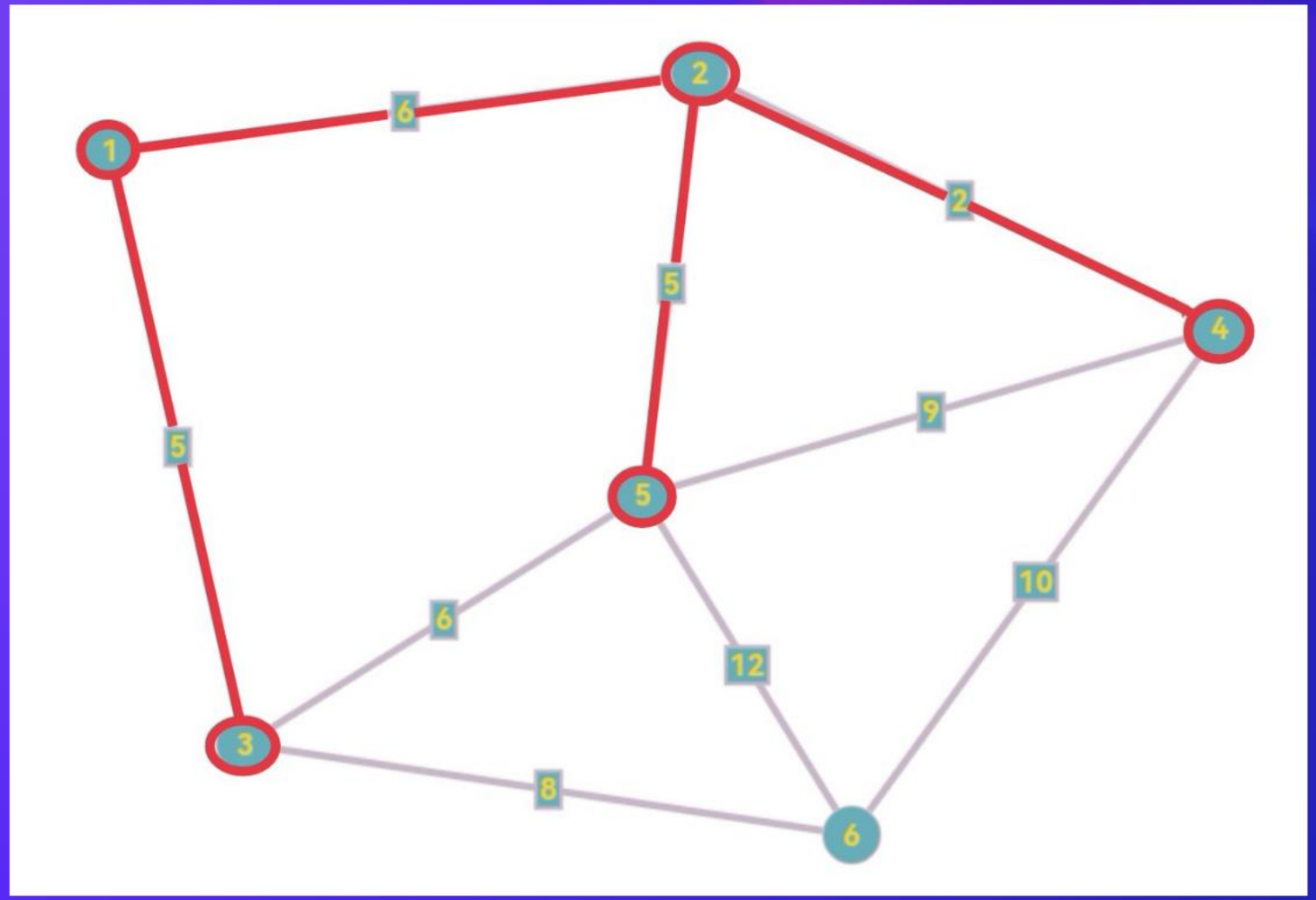


Fig 4. “Applying Kruskal’s MST step: 4 on the example graph” [3]

STEP 5:

- The next smallest edge is edge 8 from node 3 to node 6
- Now, there are a total of V-1 = 5 edges, so the program ends
- Or, it can keep selecting the next smallest edge, but the result will be the same

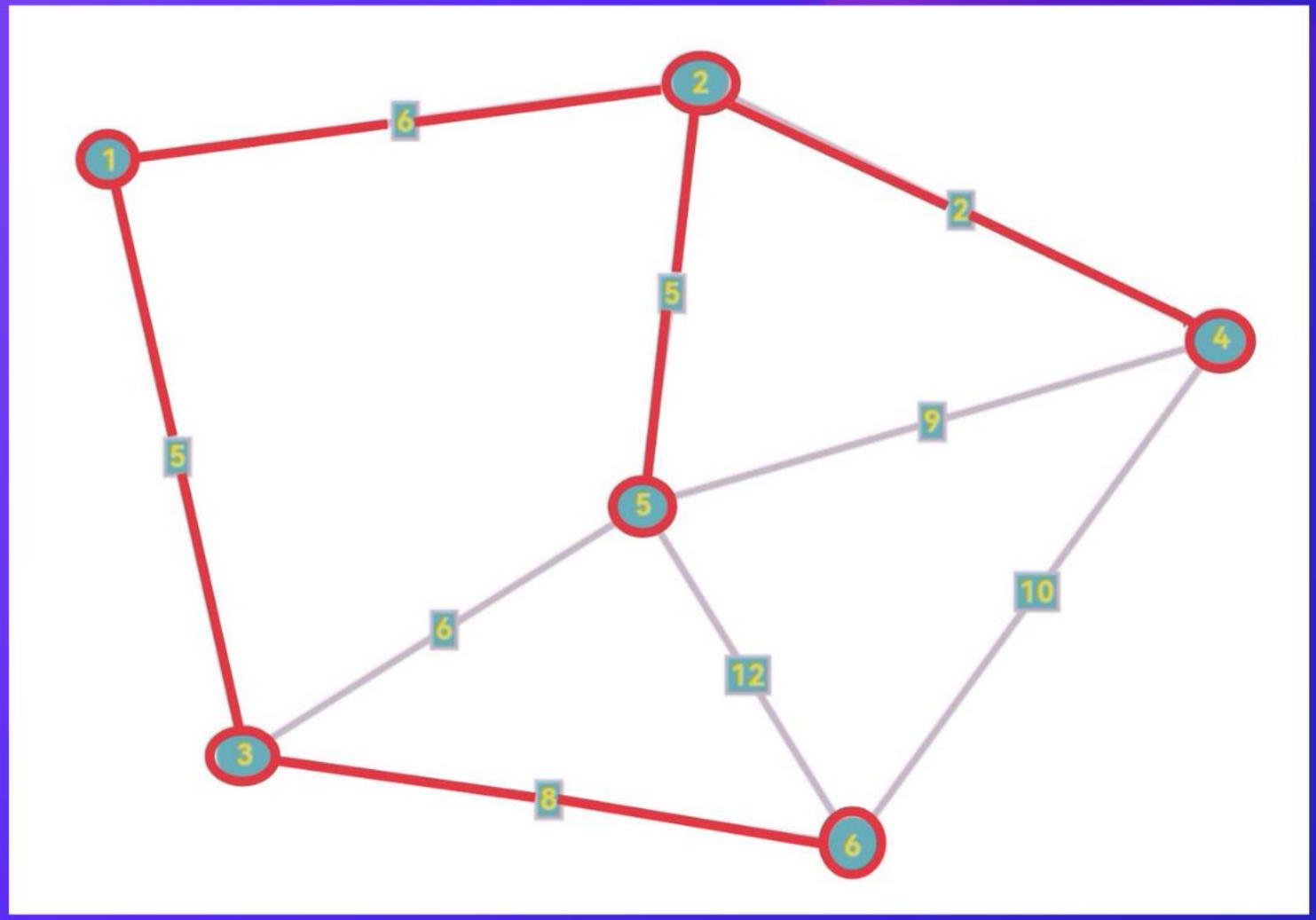


Fig 5. "Applying Kruskal's MST step: 5 on the example graph" [3]

APPLICATION FIELD

A Minimum Spanning Tree to solve a Transportation Problem

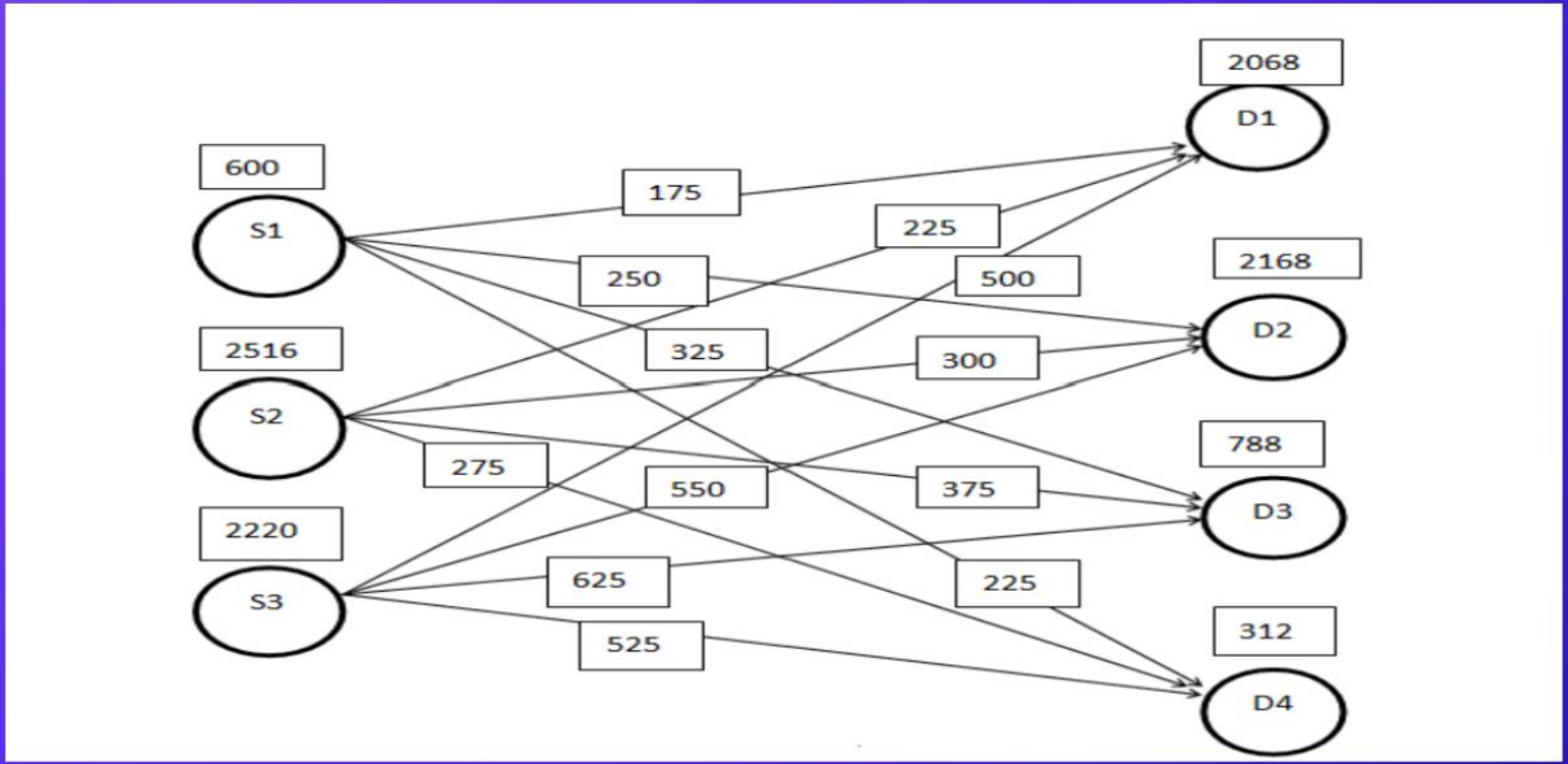


Fig 6. " representation of the transportation problem" [4]

Nodes S1, S2, and S3 represent Supply

Nodes D1, D2, and D3 represent Demand

The number associated with each node represent Capacity

Weighted values represent cost per unit in AED

APPLICATION FIELD

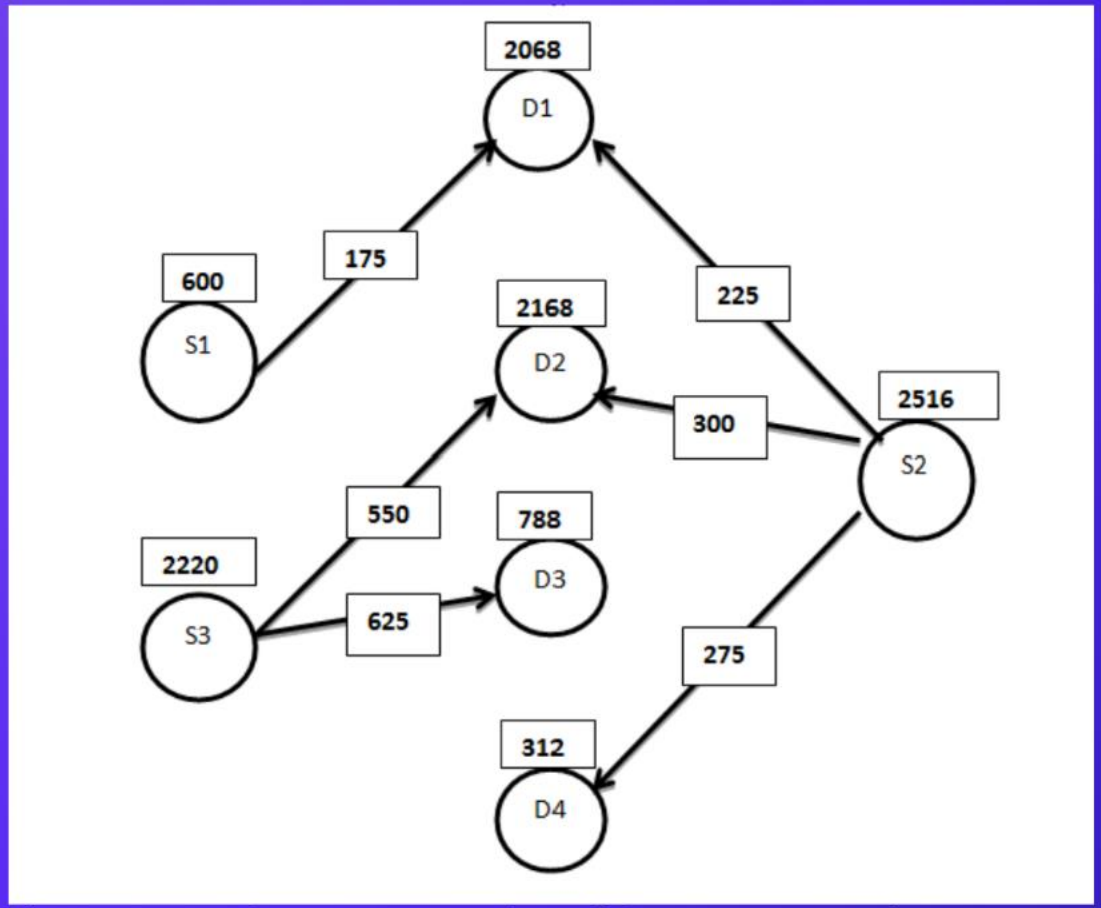


Fig 7. " a graph representation of the transportation problem after applying Kruskal's algorithm" [4]

This is a graph after Applying Kruskal's algorithm.

Because this is a directed graph, the direction of the edges are taken into consideration.

NEW TENTATIVE PERSPECTIVE

- Utilize Kruskal's minimum spanning tree algorithm to construct a public transportation system that spreads across LA County.

DATASET

- We use cities in Los Angeles County as a dataset, each city represents a vertex.

88 Cities, Incorporation and Population

City Name	Supervisorial District	Incorporation Effective	Class	Population*
Agoura Hills	3	Dec. 8, 1982	General Law	20,299
Alhambra	1	July 11, 1903	Charter	82,868
Arcadia	5	Aug. 5, 1903	Charter	56,681
Artesia	4	May 29, 1959	General Law	16,395
Avalon	4	June 26, 1913	General Law	3,460
Azusa	1	Dec. 29, 1898	General Law	50,000
Baldwin Park	1	Jan. 25, 1956	General Law	72,176
Bell	4	Nov. 7, 1927	Charter	33,559
Bell Gardens	4	Aug. 1, 1961	General Law	39,501
Bellflower	4	Sept. 3, 1957	General Law	79,190
Beverly Hills	3	Jan. 28, 1914	General Law	32,701
Bradbury	5	July 26, 1957	General Law	921

Fig 8. “Part of the 88 cities' incorporation and population table from the website of the County of Los Angeles [5]

THE PROCESS OF DETERMINING CITIES

01

- First, we divided Los Angeles County into regions
- The purpose of this project is to create a public transportation system that spans across the entire county.

02

- For each region, we selected some of the more popular cities.
- A city is considered popular if it has a high population or features attractive spots, such as famous beaches.

03

- For each region, we also selected some cities with lower populations
- We believe it will help balance the inequality gap and boost the economies by connecting them with the more popular ones.

DATASET

- We stored weighted values associated with each edge using the Measure Distance tool (in miles) on Google Maps.

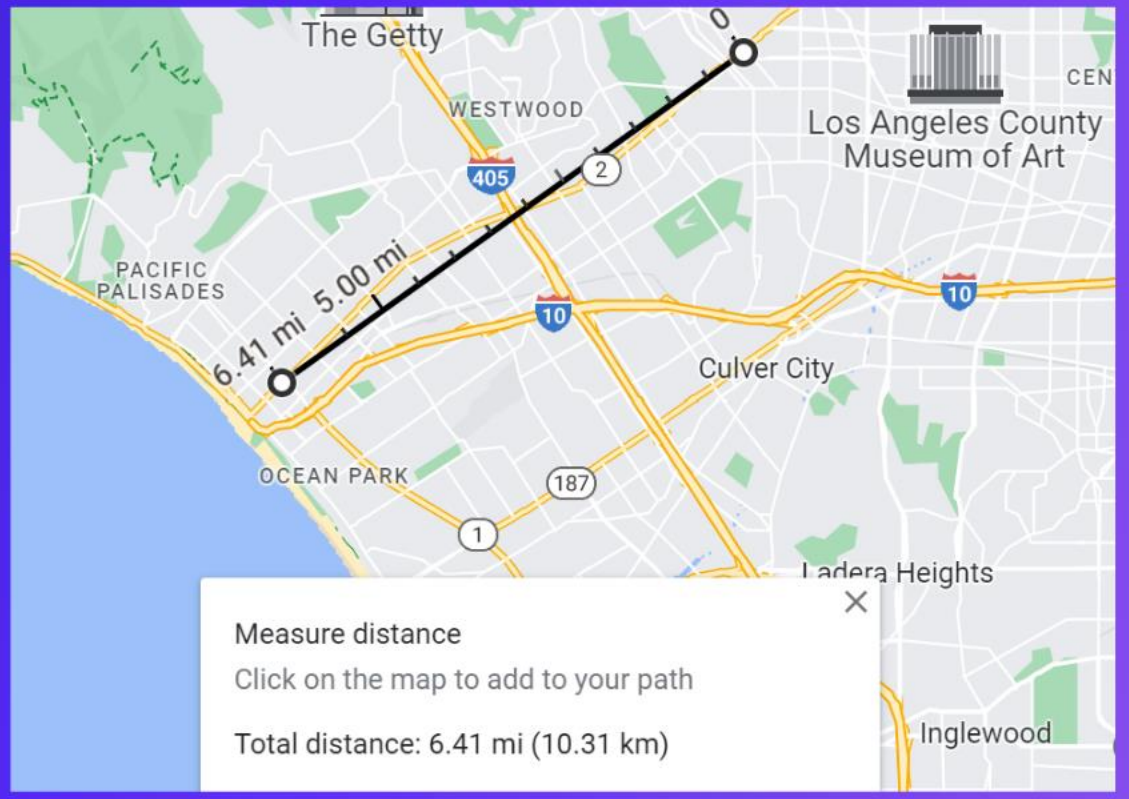


Fig 9. "An example of using the measure distance tool from Beverly Hills to Santa Monica in Google Map. [6]

NOTE

The error value for distance is +1 and -1 miles

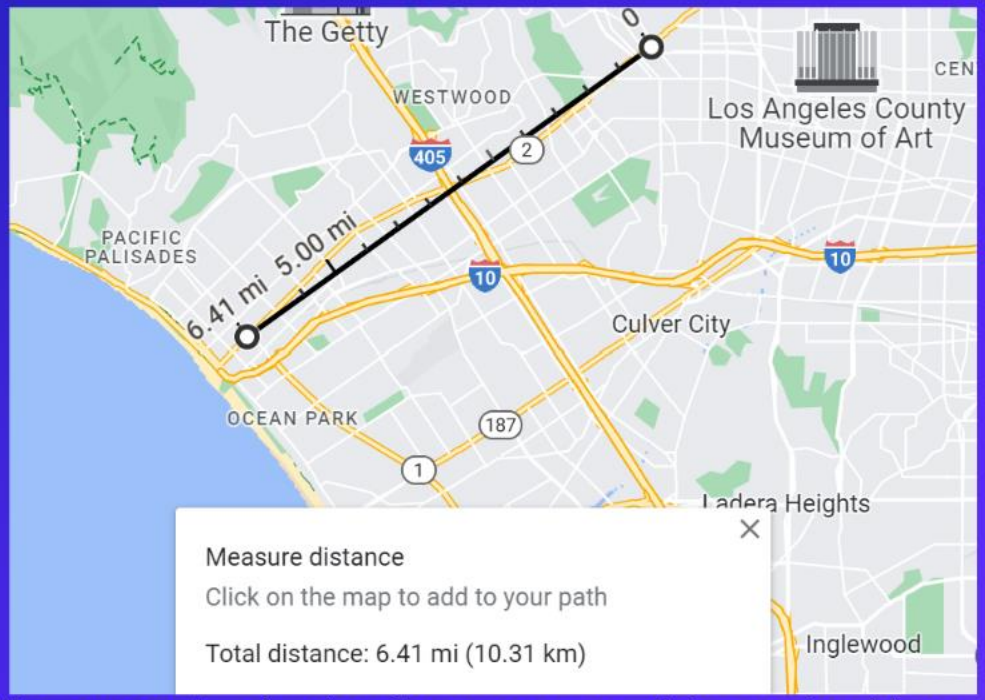


Fig 9. "An example of using the measure distance tool from Beverly Hills to Santa Monica in Google Map. [6]

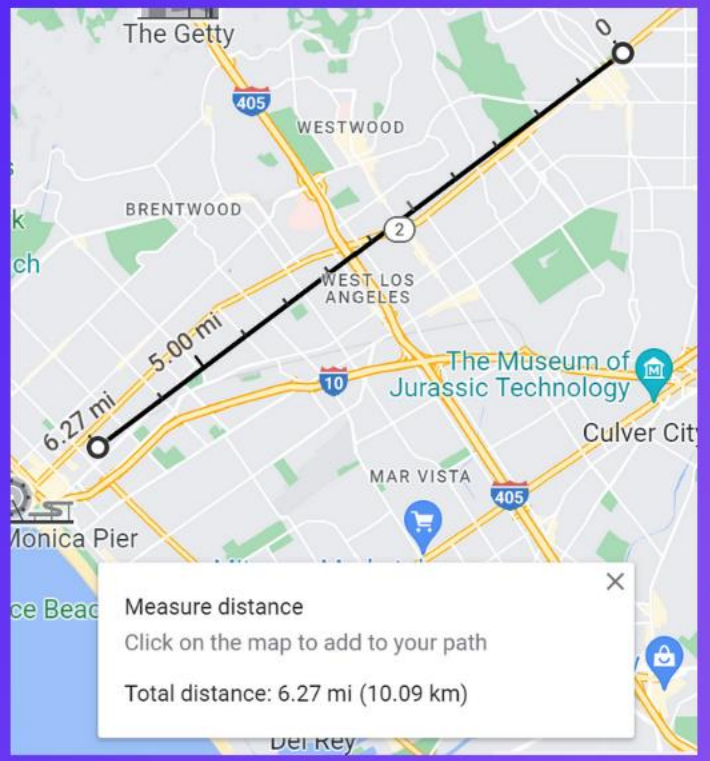


Fig 10. An example of second time measuring the distance by using the same method of figure 5. [6]

APPLYING KRUSKAL'S MST ALGORITHM

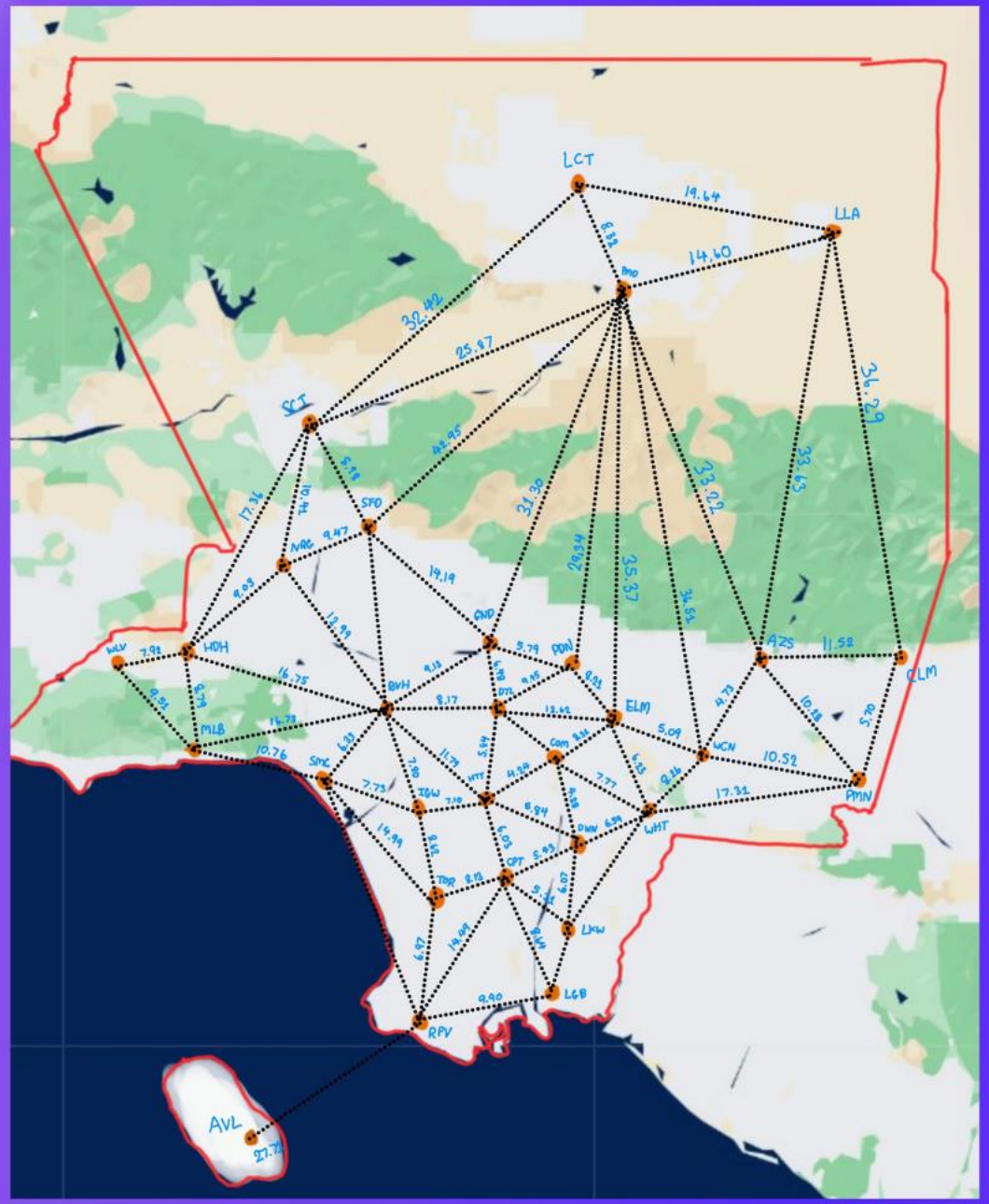
```
MST-KRUSKAL(G, w)
1  A = ∅
2  for each vertex v ∈ G.V
3      MAKE-SET(v)
4  sort the edges of G.E into nondecreasing order by weight w
5  for each edge (u, v) ∈ G.E, taken in nondecreasing order by weight
6      if FIND-SET(u) ≠ FIND-SET(v)
7          A = A ∪ {(u, v)}
8          UNION(u, v)
9  return A
```

Fig 11 "Kruskal's MST Algorithm Pseudocode". [7]

KRUSKAL'S MST ALGORITHM EXPLAINATION

- Initialize **A** as an empty set
- **Make a set** for each vertex. It creates a set containing only vertex 'v'.
- **Sort the edges** of G in non-decreasing order by weight:
- For each edge (u, v) Loop through the sorted edges.
- Check if the vertices of the edge belong to **different sets** (Check the cycle)
- If the vertices belong to different sets, **add this edge** to the MST represented by set A.
- **Union** the sets to merge them
- Finally, **return** the set of edges (A) that form **the minimum spanning tree.**

GRAPH OF INITIAL 30 CITIES



- Programming Language: Java
- Tool: VS Code
- Passing the total of 70 edges
- Shortest Distance: 4.24 miles
- Longest Distance: 42.95 niles

Fig 12. the weighted graph of initial 30 cities

Fig 13. The simulating result

PRELIMINARY RESULTS

COM - HTT weight 4.24
COM - DWN weight 4.28
AZS - WCN weight 4.73
ELM - WCN weight 5.09
CPT - LKW weight 5.11
DTL - HTT weight 5.28
DWN - CPT weight 5.43
CLM - PMN weight 5.7
GND - PDN weight 5.79
WHT - ELM weight 6.23
BVH - SMC weight 6.33
GND - DTL weight 6.48
WHT - DWN weight 6.59
LGB - LKW weight 6.59
TOR - RPV weight 6.97
HTT - IGW weight 7.1
SMC - IGW weight 7.73
HDH - WLV weight 7.92
TOR - CPT weight 8.13
LCT - PMD weight 8.32
HDH - MLB weight 8.79
SCT - SFD weight 8.98
NRG - HDH weight 9.03
NRG - SFD weight 9.47
AZS - PMN weight 10.28
MLB - SMC weight 10.76
LLA - PMD weight 14.6
PMD - SCT weight 25.87
AVL - RPV weight 27.71

RESULTS AFTER APPLYING THE SAME PROCESS FOR A FEW TIMES

- There are a total of 9 color lines: 74 stations
- Orange Line consists of 16 stations
- Yellow Line consists of 8 stations
- Orange Line connects all the regions together

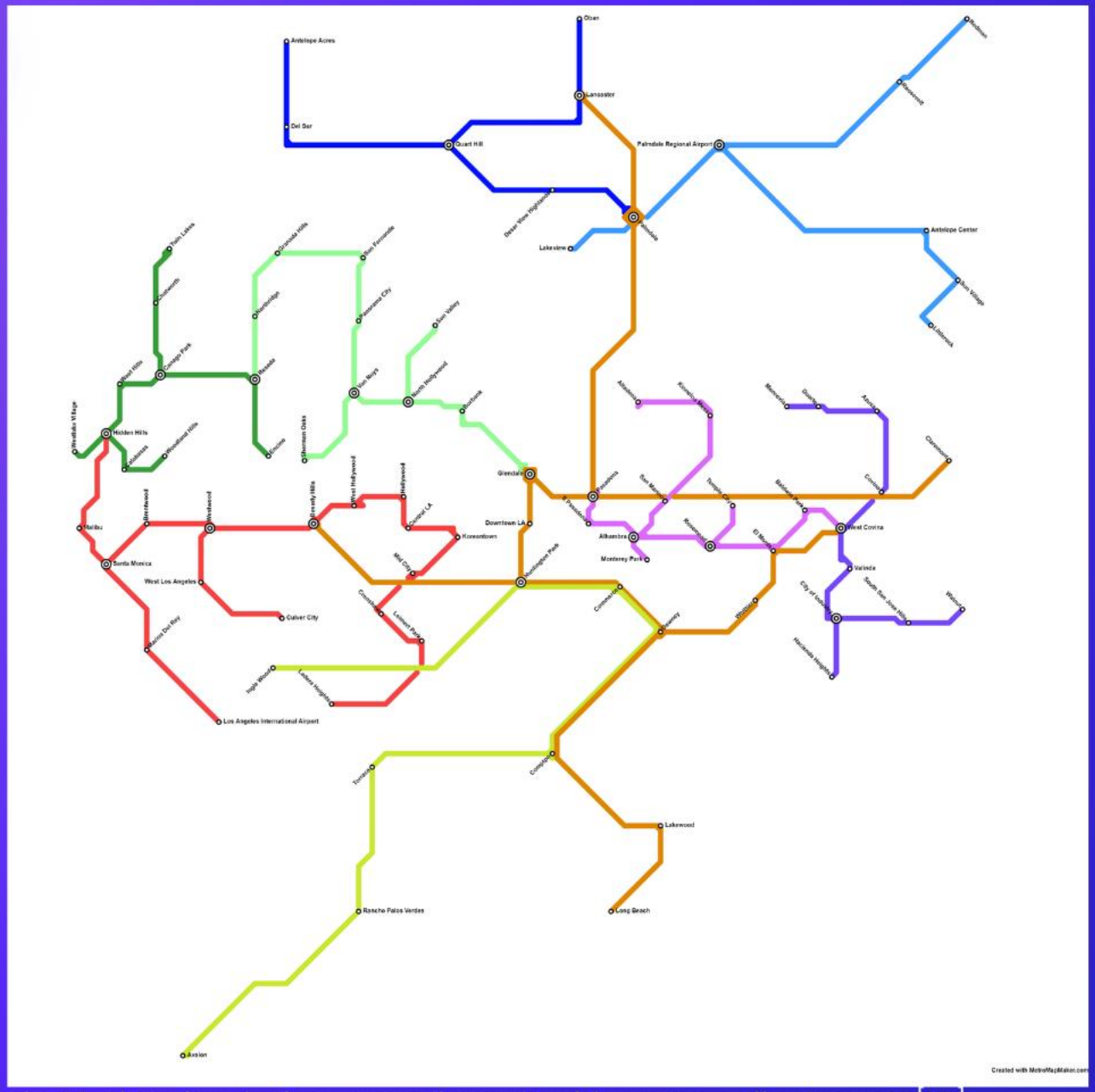


Fig 14. The simulating overall result with some adjustments[15]

Note: the map is not up to scale

Fig 15. The simulating of the West LA region[15]

ZOOM IN RESULT

- West LA region
- Consisting of 3 main lines:
- Light Green Line: 11 stations
- Dark Green Line: 10 stations
- Red Line: 18 stations

Note: the map is not up to scale

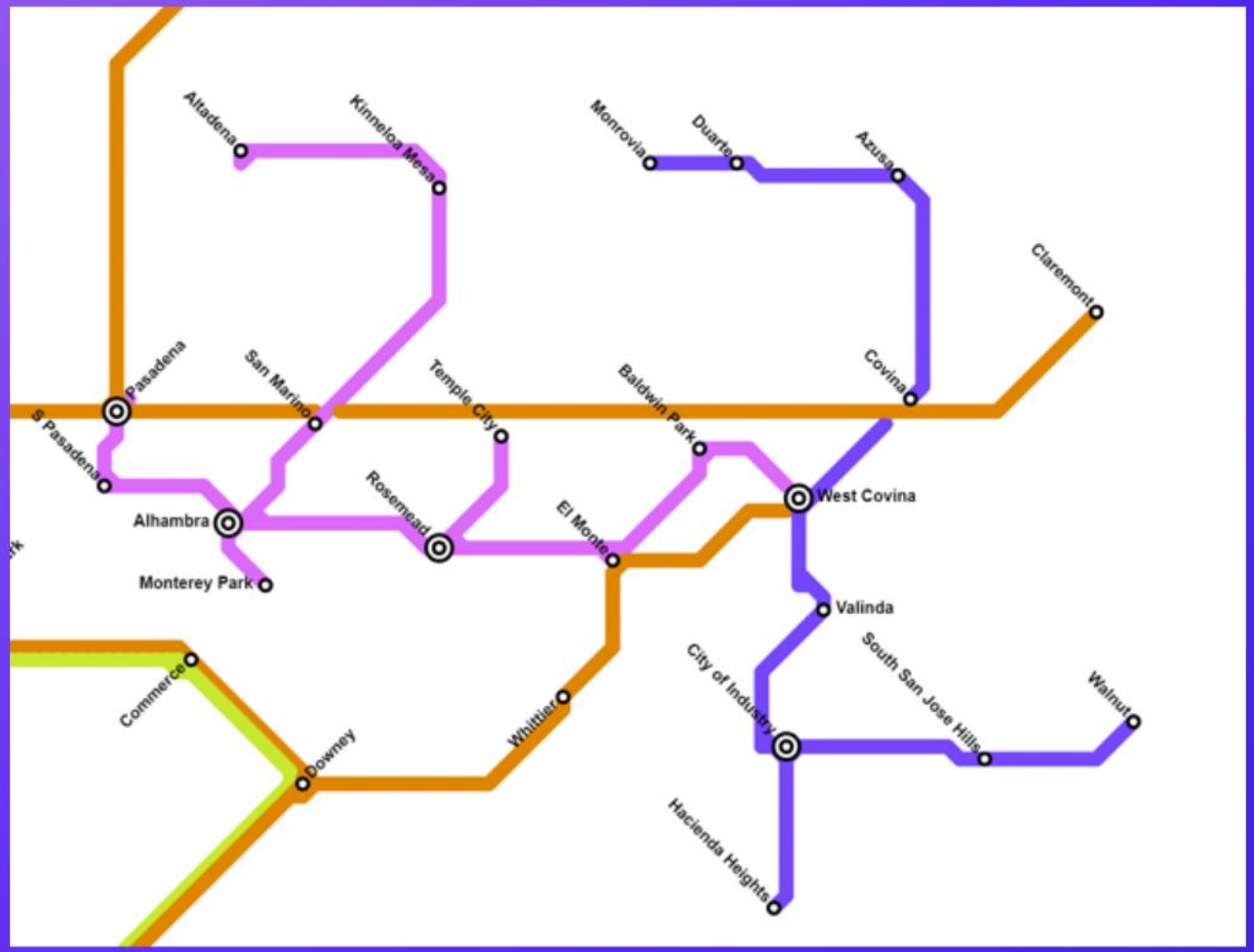


Fig 16. The simulating of the East LA region[15]

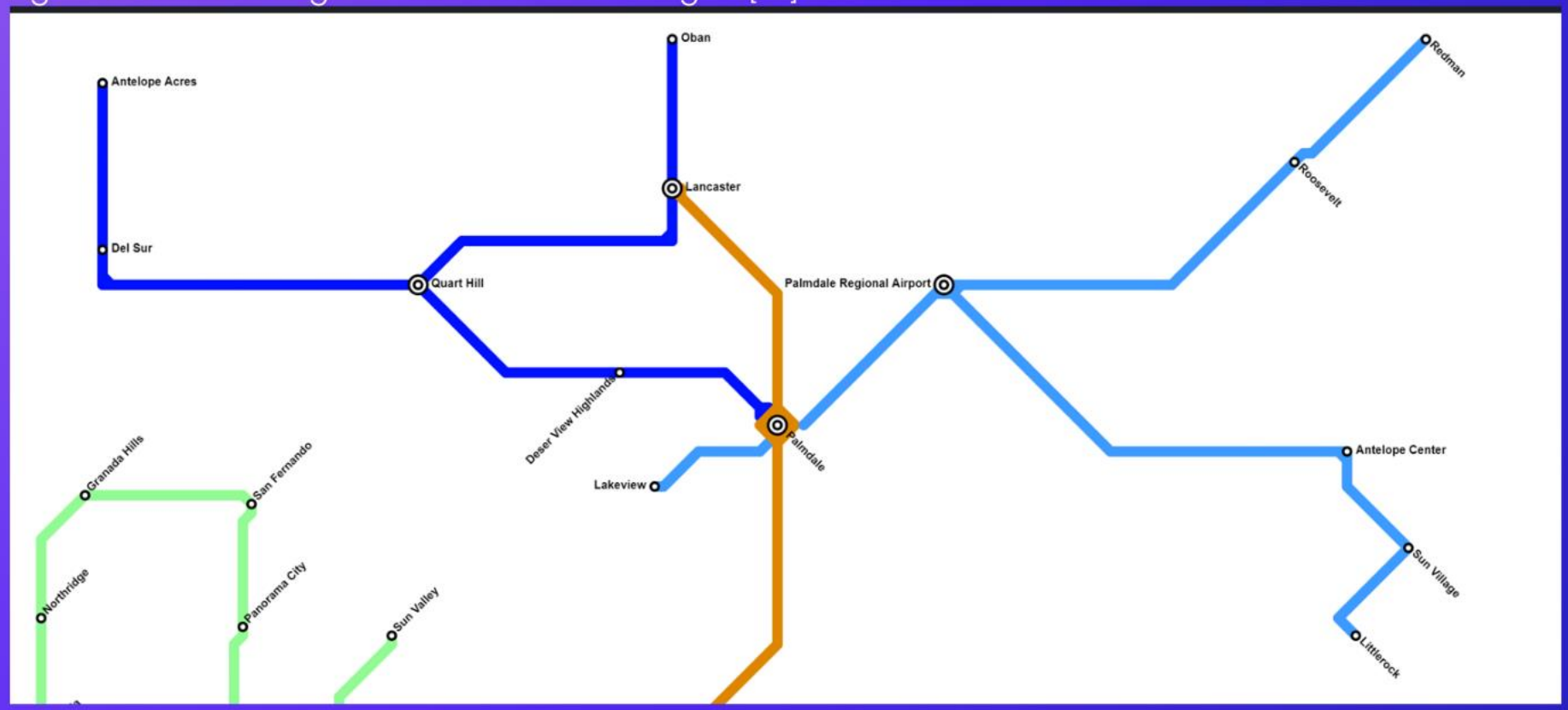
ZOOM IN RESULT

- East LA region
- Consist of 2 main lines:
- Pink Line: 12 stations
- Purple line: 10 stations

Note: the map is not up to scale

ZOOM IN RESULT

Fig 17. The simulating of the North East LA region[15]



- North East LA region
- Consist of 2 main lines: Light Blue Line: 8 stations
- Navy Blue line: 7 stations

Note: the map is not up to scale

CONTRIBUTIONS

ECONOMY

- Our purpose is to connect small cities with lower populations to the economic powerhouses of larger cities.
- It enhances economic opportunities for residents in smaller communities
- It also actively contributes to reducing economic disparities between these regions.

ACCESSIBILITY

- The system prioritizes accessibility and inclusivity, ensuring that a wide demographic can benefit from its services.
- Our commitment to accessibility extends to people with disabilities, low-income group, and people who are living in the remote areas

TRAFFIC

- Reduced traffic congestion leads to shorter commute times
- The aim is to improve productivity and the overall quality of life for LA County residents.
- The system prioritizes road safety to reduce the chances of accidents while driving.,

CONTRIBUTIONS

"A DEVELOPED COUNTRY IS NOT A PLACE WHERE THE POOR HAVE CARS. IT'S WHERE THE RICH USE PUBLIC TRANSPORTATION."[8]

-GUSTAVO PETRO-

COMPARISONS OF SIMILAR ALGORITHMS

BORUVKA'S ALGORITHM PSEUDOCODE

Algorithm MST:

Input: $G(V, E)$
Output: T, the **MST**

1: $T \leftarrow$ empty graph
2: **for** each v in V **do**
3: Let e be the minimum weight edge in **G** that is incident on v
4: $T \leftarrow T + \{e\}$
5: **end for**
6: $G' \leftarrow$ **G** with all edges in T contracted
7: $T' \leftarrow$ recursively compute the minimum spanning tree of G'
8: return $T + T'$

Fig 18 "Boruvka's MST Algorithm Pseudocode". [12]

BORUVKA'S ALGORITHM EXPLAINATION

- Initialize T as an empty graph
- Loop through each vertex v in V
- Find the minimum weight edge incident **on v**
- **Add** the minimum weight edge 'e' **to the MST T**
- **Repeat this process** for all vertices in the graph
- Contract edges in G to form a new graph G
- Continue to find the MST within the contracted graph G by using a recursive approach, treating G as a smaller subproblem.
- **Return the combination** of the union of T and the recursively computed MST of G

PRIM'S ALGORITHM

PSEUDOCODE

Prim-MST(G)
    Select an arbitrary vertex s to start the tree from.
    While (there are still nontree vertices)
        Select the edge of minimum weight between a tree and nontree vertex
        Add the selected edge and vertex to the tree T_{prim}.

Fig 19 "Kruskal's MST Algorithm Pseudocode". [9]

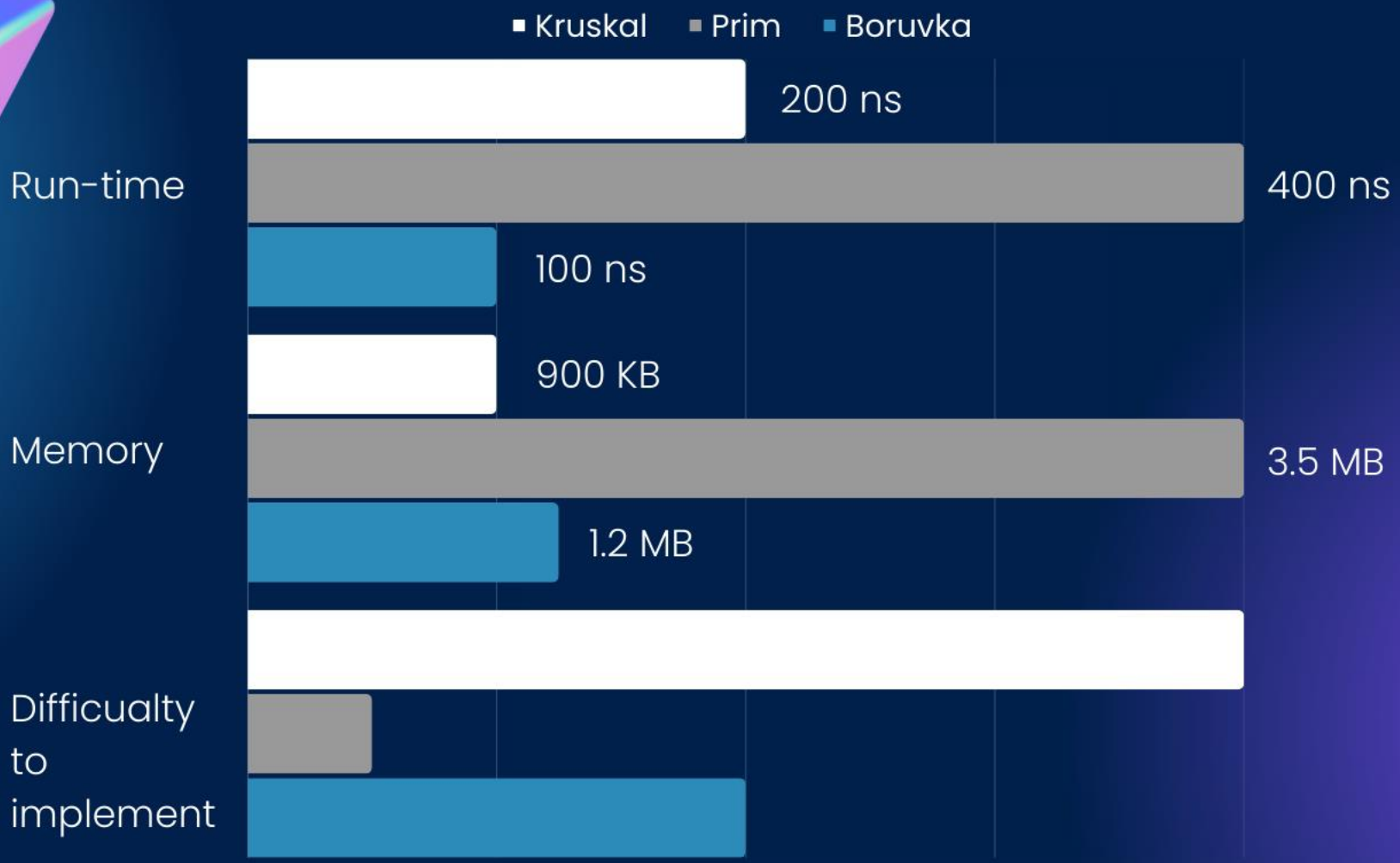
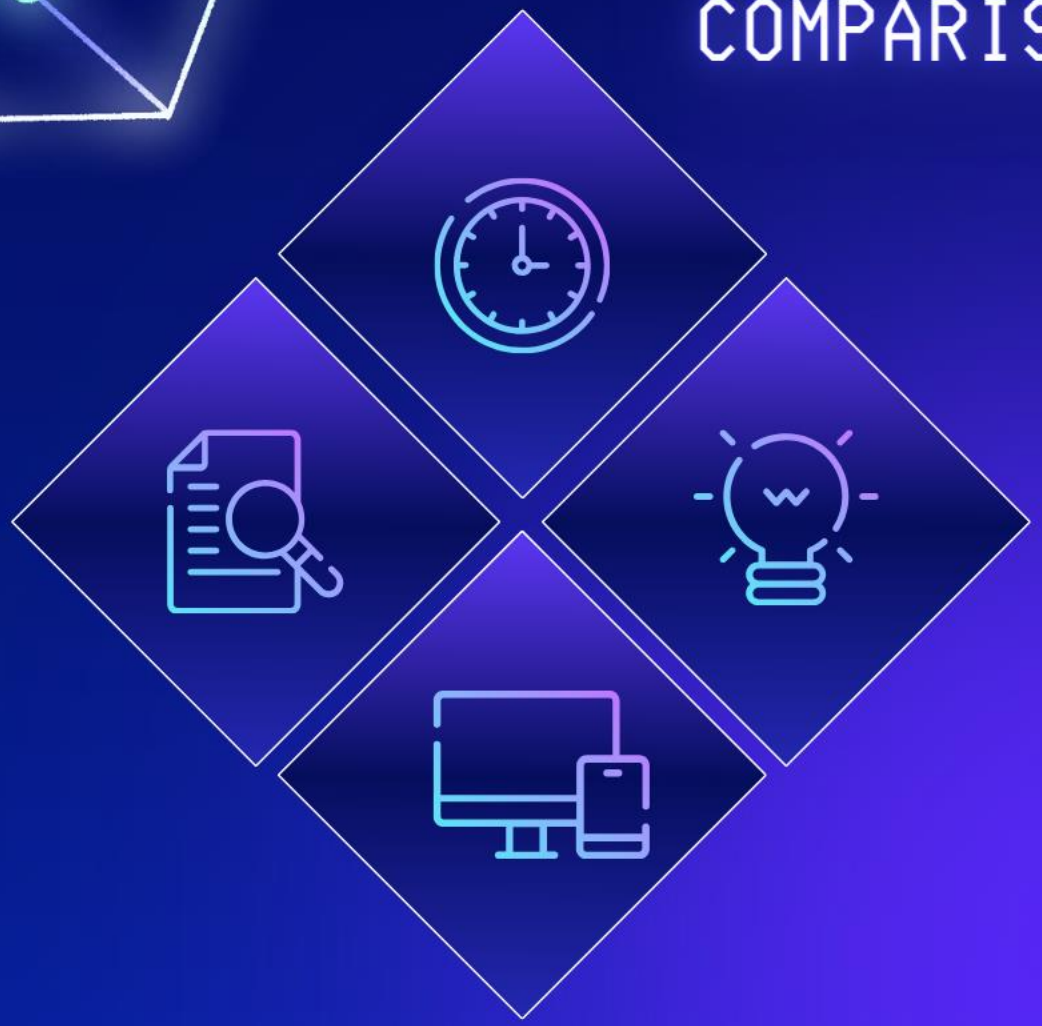


Table 1: Table of the comparison of three MST algorithms

COMPARISONS OF SIMILAR ALGORITHMS

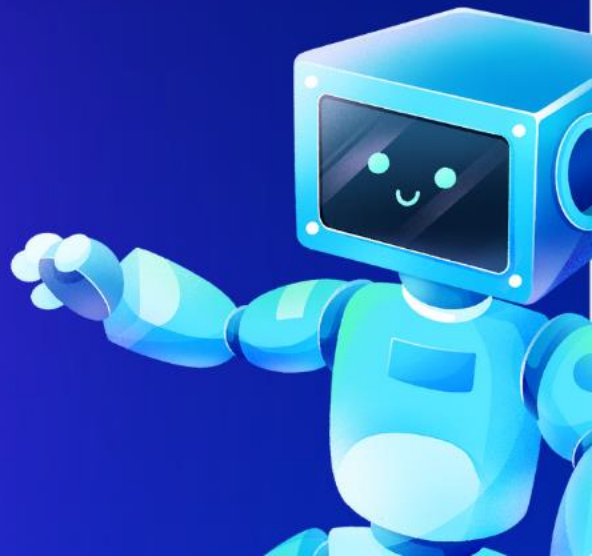


1. RUN-TIME ANALYSIS

2. MEMORY(SPACE) ANALYSIS

3. THE DIFFICUALTY OF IMPLEMENTATION

4. PROS AND CONS



COMPARISONS OF SIMILAR ALGORITHMS

RUN-TIME ANALYSIS

Kruskal's algorithm

- O(log V) [9]

---

our implementation:

- O(E log E)
- We passed list of edges and vertices, instead of graph

Prim's algorithm

- O(E + V log V) [11]

---

our implementation:

- O(V log E)
- We used priority queue

Boruvka's algorithm

- O(E log V) [10]

---

our implementation:

- O(E log V)
- We implemented Union-find data structure

ADVANTAGE & DISADVANTAGE OF KRUSKAL'S ALGORITHM

CON

- When it makes its computations, it changes the starting point, which leads to some adjustments of planning the transportation system
- It can be highly computationally expensive, especially for high number of stations (locations) [11]

PRO

- The algorithm is a greedy algorithm. That means its can be adapted and extended to various optimization problems.
- For example, in Karel's paper, he uses Kruskal's reverse MST algorithm for optimization ship transportation [13]

LIMITATIONS

01

DATA SIZE AND SCALABILITY

- The algorithm's efficiency can be affected by the size of the dataset
- It may lead to memory and processing limitations when dealing with extensive data.

02

SINGLE ALGORITHM APPROACH

- A combination of algorithms could better address the complex and multifaceted challenges in transportation planning [14].

03

GEOGRAPHICAL REGION

- The project does not encompass a detailed analysis or accounting for the unique geographical characteristics of Los Angeles County.

04

BUDGET CONSTRAINTS

- Building and maintaining public transportation infrastructure require substantial financial resources.
- The project may not address the financial feasibility of the proposed system.

FUTURE WORK

Expansion of Destination Points

- We can add more destination points, especially in **the South LA region**, that are spread across LA County.

Geographical Considerations

- Take into account geographical features and constraints when expanding the transportation network.

Interactive Map

- Incorporate an interactive map displaying the transportation network that users can easily see it in real time.

Q&A

REFERENCES

[1] R. Sedgewick and K. Wayne, "Algorithms," in *Kruskal's algorithm*, 4th ed., Addison-Wesley Professional, 2011, pp. 624–629.

[2] M. A. Weiss, "Data structures and algorithm analysis in Java," in *Kruskal's algorithm*, 3rd ed.,2012, pp.397–399.

[3] "Create Graph online and find shortest path or use other algorithm," *Graph Online*. https://graphonline.ru/en/

[4] "A Minimum Spanning Tree Approach of Solving a Transportation Problem," *AKPAN, N. P. & IWOK, I. A.*, vol. 5, no. 3, pp. 9–18, Mar. 2017.

[5] "Maps and Geography – COUNTY OF LOS ANGELES," *COUNTY OF LOS ANGELES*. https://lacounty.gov/government/about-la-county/maps-and-geography/ (accessed Oct. 01, 2023).

[6] "Google Maps," *Google Maps*. https://www.google.com/maps

[7] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, "Minimum Spanning Trees," in *Introduction to Algorithms*, 3rd ed., 2009, pp. 624–642. [Online]. Available: http://portal.acm.org/citation.cfm?id=1614191

[8] "Gustavo Petro quotes," *Quote.org*. https://quote.org/quote/a-developed-country-is-not-a-place-632780

[9] S. Skiena, *The Algorithm Design Manual*. 2008. doi: 10.1007/978-1-84800-070-4.

[10] M. Mareš, "Two linear time algorithms for MST on minor closed graph classes," Archivum Mathematicum, vol. 040, no. 3, pp. 315–320, Jan. 2002, doi: 10.3929/ethz-a-004354035.

[11] "Route Planning using The Kruskal's Algorithm: A Case of Lobels Bulawayo," Frederick M. DANDURE, Jun. 2018, [Online].
Available: https://www.researchgate.net/publication/333672845_Route_Planning_using_The_Kruskal%27s_Algorithm_A_Case_of_Lobels_Bulawayo

REFERENCES

[12] "MinimumSpanningTrees," West Virginia University, Lane Department of Computer Science and Electrical Engineering Directory. Accessed: Nov. 03, 2023. [Online]. Available: https://community.wvu.edu/~krsubramani/courses/fa01/random/lecnotes/lecture11.pdf

[13] "The Use of Minimal Spanning Tree for Optimizing Ship Transportation," Karel Antoš, pp. 81–85, Mar. 2016, doi: 10.17818/NM/2016/SI1.

[14] V. R. Vuchic, Urban Transit Systems and Technology. 2007. doi: 10.1002/9780470168066.

[15] "Metro Map Maker," Metro Map Maker. https://metromapmaker.com/