Introduction to Mathematical Modeling

Ramapo College of New Jersey

Instructor: Dr. Atul Anurag

Semester: Fall 2025 Date: September 15, 2025

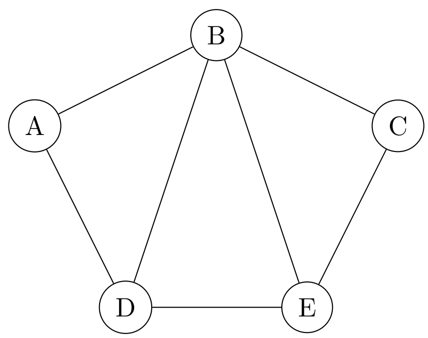
Definitions

Hamiltonian Path: A path that visits every vertex in a graph exactly once.

Hamiltonian Circuit: A circuit (closed loop) that visits every vertex exactly once and returns to the starting point.

Example Graph

Below is a simple graph with 5 vertices:



Hamiltonian Path Example

One possible Hamiltonian path in this graph is:

$$A \rightarrow B \rightarrow C \rightarrow E \rightarrow D$$

Hamiltonian Circuit Example

One possible Hamiltonian circuit is:

$$A \rightarrow B \rightarrow C \rightarrow E \rightarrow D \rightarrow A$$

What is the Traveling Salesman Problem (TSP)?

Problem Statement: Given a list of cities and the cost (or distance) between each pair of cities, find the shortest possible route that:

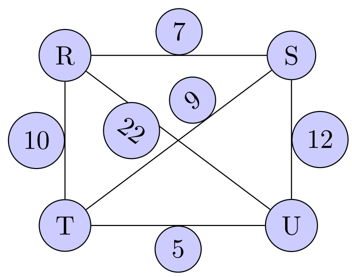
- Visits each city exactly once, and
- Returns to the starting city.

In TSP, we are interested in finding a Hamiltonian circuit with the **minimum total cost**.

Problem Statement

The graph below shows the time (in minutes) to travel between the vertices R, S, T, and U. Find the least-cost Hamiltonian circuit starting and ending at vertex T.

Graph Representation



Possible Hamiltonian Circuits Starting at T

We consider all permutations of the other three vertices after T and return to T.

- $\mathbf{T} \rightarrow \mathbf{R} \rightarrow \mathbf{S} \rightarrow \mathbf{U} \rightarrow \mathbf{T}$ Cost $= 10 + 7 + 12 + 5 = 34$
- $\mathbf{T} \rightarrow \mathbf{R} \rightarrow \mathbf{U} \rightarrow \mathbf{S} \rightarrow \mathbf{T}$ Cost $= 10 + 22 + 12 + 9 = 53$
- $\mathbf{T} \rightarrow \mathbf{S} \rightarrow \mathbf{R} \rightarrow \mathbf{U} \rightarrow \mathbf{T}$ Cost $= 9 + 7 + 22 + 5 = 43$
- $\mathbf{T} \rightarrow \mathbf{S} \rightarrow \mathbf{U} \rightarrow \mathbf{R} \rightarrow \mathbf{T}$ Cost $= 9 + 12 + 22 + 10 = 53$
- $\mathbf{T} \rightarrow \mathbf{U} \rightarrow \mathbf{R} \rightarrow \mathbf{S} \rightarrow \mathbf{T}$ Cost $= 5 + 22 + 7 + 9 = 43$
- $\mathbf{T} \rightarrow \mathbf{U} \rightarrow \mathbf{S} \rightarrow \mathbf{R} \rightarrow \mathbf{T}$ Cost $= 5 + 12 + 7 + 10 = 34$

Answer: Least-Cost Hamiltonian Circuit

The two circuits with the minimum total time are:

$$\boxed{\begin{array}{l} \mathrm{T} \rightarrow \mathrm{R} \rightarrow \mathrm{S} \rightarrow \mathrm{U} \rightarrow \mathrm{T}, \quad \text{Cost} = 34 \\ \mathrm{T} \rightarrow \mathrm{U} \rightarrow \mathrm{S} \rightarrow \mathrm{R} \rightarrow \mathrm{T}, \quad \text{Cost} = 34 \end{array}}$$

Complete Graphs

Definition

A **complete graph** is a graph in which **every pair of distinct vertices is connected by exactly one edge**. There are no missing connections between vertices.

The complete graph with n vertices is denoted as K_n.

Number of Edges in a Complete Graph

Each edge connects a pair of distinct vertices. To find the total number of edges, we count the number of such pairs.

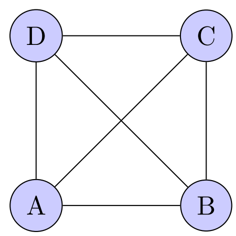
$$\text{Number of edges in } K_n = \frac{n(n-1)}{2}$$

Key Points

- A complete graph has an edge between *every* pair of vertices.
- Complete graphs are commonly used in problems like the **Traveling Salesman Problem (TSP)**, where every location (vertex) must be reachable from every other.

Complete Graph Examples

Complete Graph K_4 (4 vertices, 6 edges)



Heuristic Algorithms

A **heuristic algorithm** is an algorithm that is designed to solve a problem quickly, but does not guarantee an optimal solution. Instead, it uses practical methods or "rules of thumb" to produce a solution that is *good enough* within a reasonable amount of time.

Greedy Algorithms

A **greedy algorithm** builds up a solution piece by piece, always choosing the option that looks best at the moment. It makes a series of choices, each of which is the *locally optimal* choice in the hope that the overall solution will also be optimal.

Key characteristic: At each step, the algorithm selects the best immediate option, without considering the global consequences.

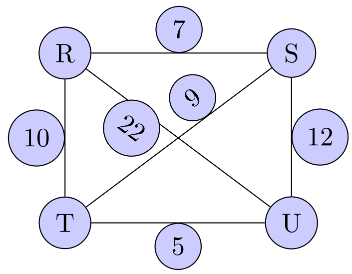
Nearest Neighbor Algorithm

The **Nearest Neighbor Algorithm (NNA)** is a heuristic algorithm often used to find a short path in problems such as the Traveling Salesperson Problem (TSP).

1. Start at the **home city**.
2. Among all cities that have not yet been visited, travel to the **nearest unvisited city**.

3. Repeat Step 2 until all cities have been visited.
4. Return to the **home city** to complete the circuit.

Graph



Applying Nearest Neighbor Algorithm (Starting at R)

Step 1: Start at city R.

From R, distances to other cities are:

$$R \to S = 7, \quad R \to T = 10, \quad R \to U = 22.$$

Nearest city from R is S (distance 7).

Step 2: From city S:

Remaining unvisited cities: T, U.

Distances from S are:

$$S \to T = 9, \quad S \to U = 12.$$

Nearest city is T (distance 9).

Step 3: From city T:

Remaining unvisited city: U.

Distance:

$$T \to U = 5.$$

Travel to city U.

Step 4: Return to starting city R:

Distance:

$$U \to R = 22.$$

Resulting Route and Total Distance

The route found by the Nearest Neighbor Algorithm is:

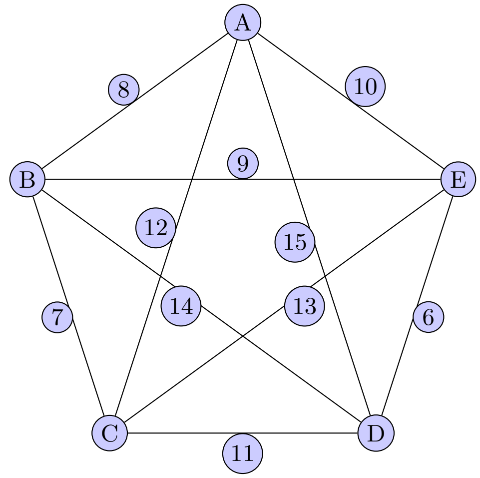
$$R \to S \to T \to U \to R.$$

The total distance traveled is:

$$7 + 9 + 5 + 22 = 43.$$

Total distance: $\boxed{43}$.

Complete Graph with 5 vertices



Applying Nearest Neighbor Algorithm starting at B

Step 1: Start at city B.

Distances from B to others:

$$B \to A = 8, \quad B \to C = 7, \quad B \to D = 14, \quad B \to E = 9.$$

Nearest city: C (distance 7).

Step 2: From city C:

Remaining unvisited: A, D, E.

Distances:

$$C \to A = 12, \quad C \to D = 11, \quad C \to E = 13.$$

Nearest city: D (distance 11).

Step 3: From city D:

Remaining unvisited: A, E.

Distances:

$$D \to A = 15, \quad D \to E = 6.$$

Nearest city: E (distance 6).

Step 4: From city E:

Remaining unvisited: A.

Distance:

$$E \to A = 10.$$

Travel to A.

Step 5: Return to start city B:

Distance:

$$A \to B = 8.$$

Resulting Route and Total Distance

Route:

$$B \to C \to D \to E \to A \to B.$$

Total distance:

$$7 + 11 + 6 + 10 + 8 = 42.$$

Total distance: $\boxed{42}$.

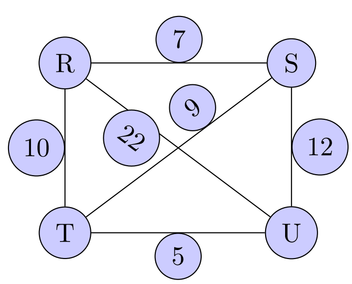
Sorted Edges Algorithm

The **Sorted Edges Algorithm** (also known as the *Cheapest Link Algorithm*) is a method for approximating the solution to the Traveling Salesperson Problem (TSP). The goal is to find a Hamiltonian circuit of minimal total cost in a weighted complete graph.

Steps of the Algorithm

1. **List all edges** in the complete graph along with their weights (costs or distances).
2. **Sort the edges** in ascending order based on their weights.
3. **Select edges one at a time** from the sorted list, starting with the lowest weight, under the following conditions:
   - **No vertex** can be included in more than **two edges**. (This ensures a simple circuit without branches.)
   - **Do not form a circuit** until all vertices are included. (Premature cycles are not allowed.)
4. **Repeat the selection process** until a Hamiltonian circuit is formed (a cycle that visits each vertex exactly once and returns to the starting vertex).

Find the shortest Hamiltonian circuit using the sorted edges algorithm for the graphs below.



Sorted Edges Algorithm Steps

Step 1. Sort edges by increasing weight:

(a) T–U: 5

(b) R–S: 7

(c) S–T: 9
(d) R–T: 10
(e) S–U: 12
(f) R–U: 22

Step 2. Build the circuit by adding edges according to the rules:

- Add T–U (5)
- Add R–S (7)
- Add S–T (9)
- Skip R–T (10): would give vertex T three edges
- Skip S–U (12): would give vertex S three edges
- Add R–U (22): completes the Hamiltonian circuit

Result

- **Selected edges:** T–U, R–S, S–T, R–U
- **Hamiltonian Circuit:** R → S → T → U → R
- **Total Cost:**

$$5 + 7 + 9 + 22 = \boxed{43}$$

Types of Special Matrices

This section introduces different types of special square matrices. We use examples of 2×2 and 3×3 matrices to help understand each concept.

1. Symmetric Matrix

A matrix is called **symmetric** if it is equal to its transpose, i.e., $A = A^\top$. This means the elements are mirrored across the main diagonal.

Examples:

$$\begin{bmatrix} 1 & 2 \\ 2 & 3 \end{bmatrix}, \quad \begin{bmatrix} 4 & 1 & -2 \\ 1 & 5 & 3 \\ -2 & 3 & 6 \end{bmatrix}$$

2. Skew-Symmetric Matrix

A matrix is **skew-symmetric** if it is equal to the negative of its transpose, i.e., $A = -A^\top$. All diagonal elements must be zero.

Examples:

$$\begin{bmatrix} 0 & -2 \\ 2 & 0 \end{bmatrix}, \quad \begin{bmatrix} 0 & -1 & 3 \\ 1 & 0 & -4 \\ -3 & 4 & 0 \end{bmatrix}$$

3. Upper Triangular Matrix

A matrix is **upper triangular** if all the elements below the main diagonal are zero.
Examples:

$$\begin{bmatrix} 1 & 3 \\ 0 & 2 \end{bmatrix}, \quad \begin{bmatrix} 5 & -1 & 4 \\ 0 & 2 & 6 \\ 0 & 0 & 3 \end{bmatrix}$$

4. Lower Triangular Matrix

A matrix is **lower triangular** if all the elements above the main diagonal are zero.
Examples:

$$\begin{bmatrix} 7 & 0 \\ 2 & 5 \end{bmatrix}, \quad \begin{bmatrix} 3 & 0 & 0 \\ 1 & -2 & 0 \\ 4 & 6 & 8 \end{bmatrix}$$

5. Diagonal Matrix

A matrix is **diagonal** if all the non-diagonal elements are zero.
Examples:

$$\begin{bmatrix} 4 & 0 \\ 0 & 9 \end{bmatrix}, \quad \begin{bmatrix} 2 & 0 & 0 \\ 0 & -5 & 0 \\ 0 & 0 & 7 \end{bmatrix}$$

Note: A diagonal matrix is both upper and lower triangular. It is also symmetric if all diagonal elements are real numbers.

End of Lecture #4