Traveling Salesman Problem (TSP)

- Visit every city and then go home.
A Hamiltonian Path is a path that goes through each vertex exactly once.

Note: Not all edges have to be used.
Does this graph have a Hamiltonian path?
A Hamiltonian Circuit is a Hamiltonian path that starts and ends at the same vertex.

This is the traveling salesman problem.
For any given graph, finding an Euler path will be easier than finding a Hamiltonian path.
Pop Quiz!

An Euler path visits every _____ once

1) vertex
2) edge
3) something else
PROBLEM SOLVING

The Splitting-Hairs Principle

Recall the Splitting-Hairs Principle in Section 1.1. Although the definitions of Hamilton path and Euler path sound similar, they are not the same. In producing a Hamilton path, you do not have to trace every edge, as with an Euler path.

Euler path/circuit = cross each edge once

Hamiltonian path/circuit = cross each vertex once
**Definition** A complete graph is one in which every pair of vertices is joined by an edge. A complete graph with $n$ vertices is denoted by $K_n$. *

- **Examples:**

![Graphs $K_3$, $K_4$, $K_5$](image)
Let's find $K_6$
Since every vertex is visited, it doesn't matter where you start.

Let's start from A.
There are three choices for the second vertex.

Two choices remain.

There is one choice for the last vertex.
Tracing through the six branches of the tree, we see that $K_4$ has six Hamilton circuits:

- ABCDA, ABDCA, ACBDA, ACDBA, ADBC, ADCBA

There are three choices for the second vertex.

Two choices remain.

There is one choice for the last vertex.

reverse circuits
For $K_5$ we have 5 vertices A, B, C, D, E

Since we visit every vertex, we can start at A.

There are 4 vertices left. All can be reached.

From there 3 are left (all reachable). Then 2 are left, then one.
Then we take the edge back to A.

So there are $4 \times 3 \times 2 \times 1 = 24$ possible paths.
Finding Hamilton Circuits

The number of Hamilton circuits in $K_n$ is $K_n$ has $(n-1)(n-2)(n-3) \cdots 3 \times 2 \times 1$ Hamilton circuits. This number is written $(n-1)!$ and is called $(n-1)$ factorial.

<table>
<thead>
<tr>
<th>$n$</th>
<th>Number of Hamilton Circuits in $K_n$</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>2! = 2 · 1 = 2</td>
</tr>
<tr>
<td>4</td>
<td>3! = 3 · 2 · 1 = 6</td>
</tr>
<tr>
<td>5</td>
<td>4! = 4 · 3 · 2 · 1 = 24</td>
</tr>
<tr>
<td>10</td>
<td>9! = 362,880</td>
</tr>
<tr>
<td>15</td>
<td>14! = 87,178,291,200</td>
</tr>
<tr>
<td>20</td>
<td>19! = 121,645,100,408,832,000</td>
</tr>
</tbody>
</table>
Which Hamiltonian circuit is better?
In a Traveling Salesman Problem, not only do you want to visit each city exactly once, but you also want to do so as \textit{efficiently} as possible.

This will be measured by the total distance for the trip.
The **weight** of an edge is a number assigned to the edge. (Think distance between cities.)

A graph is a **weighted graph** if all of its edges have weights.

What is the weight of
a) A,B
b) C,B
c) C,D - no edge, no weight
The weight of a path is the sum of the weights of the edges along the path.

What is the weight of the path B,A,D,E?
Q: What is the relation between the weight of a path and the weight of its reverse path? Why?

A:
Q: What is the relation between the weight of a path and the weight of its reverse path? Why?

A: The are the same.

The path and its reverse path use the same edges.

(So direction doesn't matter for finding the weight of a path.)
Which is the circuit with the smaller weight?

That one is the solution to the Traveling Salesman Problem.
Solving the TSP by Brute Force

• Brute Force means trying all possible outcomes and then picking the best one.
• In solving a TSP problem by brute force, we consider all possible Hamiltonian circuits.
Solving the TSP by Brute Force

• Example: Use the weighted graph to find the sequence of cities for Danielle to visit that will minimize her total travel cost.
Solving the TSP by Brute Force

• Solution: Use brute force to explore all possible circuits:

<table>
<thead>
<tr>
<th>Hamilton Circuit</th>
<th>Weight ($)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PACMNP</td>
<td>1,280</td>
</tr>
<tr>
<td>PACNMP</td>
<td>1,460</td>
</tr>
<tr>
<td><strong>PAMCNP</strong></td>
<td><strong>1,200</strong></td>
</tr>
<tr>
<td>PAMNCP</td>
<td>1,480</td>
</tr>
<tr>
<td>PANCMP</td>
<td>1,350</td>
</tr>
<tr>
<td>PANMCP</td>
<td>1,450</td>
</tr>
<tr>
<td>PCAMNP</td>
<td>1,400</td>
</tr>
<tr>
<td>PCANMP</td>
<td>1,550</td>
</tr>
<tr>
<td>PCMANP</td>
<td>1,290</td>
</tr>
<tr>
<td>PCNAMP</td>
<td>1,470</td>
</tr>
<tr>
<td>PMACNP</td>
<td>1,300</td>
</tr>
<tr>
<td>PMCANP</td>
<td>1,270</td>
</tr>
</tbody>
</table>
Solving the TSP by Brute Force

**THE BRUTE FORCE ALGORITHM FOR SOLVING THE TSP**

- **Step 1:** List all Hamilton circuits in the graph.
- **Step 2:** Find the weight of each circuit found in step 1.
- **Step 3:** The circuits with the smallest weights tell us the solution to the TSP.
Paths are:

ABCDA (reverse ADCBA)

ACBDA (reverse ADBCA)

ABDCA (reverse ACDBA)
The Nearest Neighbor Algorithm

- There are algorithms that give good approximations to the TSP.

**The Nearest Neighbor Algorithm for Solving the TSP**

1. Start at any vertex $X$.
2. Of all the edges connected to $X$, choose any one that has the smallest weight. (There may be several with smallest weight.) Select the vertex at the other end of this edge. This vertex is called the nearest neighbor of $X$.
3. Choose subsequent new vertices as you did in step 2. When choosing the next vertex in the circuit, choose one whose edge with the current vertex has the smallest weight.
4. After all vertices have been chosen, close the circuit by returning to the starting vertex.
Start from A

B,C,D remain, B is closest.

C,D remain, C is closest.

D remains.

Lastly go back to A.
The Nearest Neighbor Algorithm

- Example: Use the nearest neighbor algorithm to schedule Danielle’s trip.
- Solution:
The Best Edge Algorithm

Step 1: Begin by choosing any edge with the smallest weight.
Step 2: Choose any remaining edge in the graph with the smallest weight.
Step 3: Keep repeating step 2; however, do not allow a circuit to form until all vertices have been used. Also, because the final Hamilton circuit cannot have three edges joined to the same vertex, never allow this to happen during the construction of the circuit.
2 is the smallest weight, add it.

3 is the next smallest, add it.

4 is the next smallest, add it.

5 is next, can't add, triple at C.

6 is next, can't add, triple at A.

7 is next, add it and get the circuit!
The Best Edge Algorithm

• Example: Use the best edge algorithm to schedule Danielle’s trip.

• Solution:

\[ 210 + 230 + 170 + 240 + 350 = 1200 \]

Notice that this circuit has a weight of 1,200, which also makes it the best solution to Danielle’s problem.