The Mathematics of Touring

Math 105

Section 003 - Prasad

December 1, 2010

Reading Quiz

1. Define **all** of the following terms **in your own words**.

   Hamilton path
   complete graph
   tour

2. How many edges are in a complete graph with 5 vertices?
3. How many distinct Hamilton circuits are there in a complete graph with 5 vertices?

Finding an optimal route

What is the cheapest way for this traveling salesman to visit all five cities only once? Assume he lives in city A and wants to get back there when he's done.

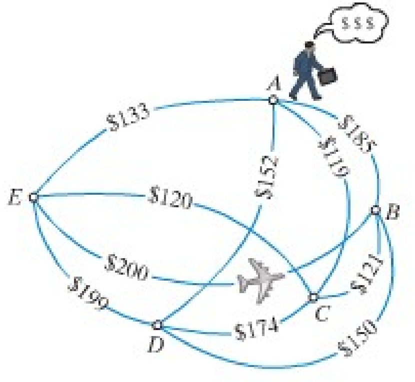


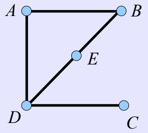
FIGURE 6-7

Hamilton Paths and Circuits

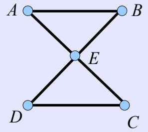
Definitions:

- A **Hamilton path** is a path that includes each vertex of the graph once and only once.
- A **Hamilton circuit** is a circuit that includes each vertex of the graph one and only once. (If a graph has a Hamilton circuit, it automatically has a Hamilton path!)

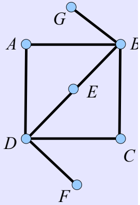
Which of the following graphs have Hamilton paths or circuits? Which of them have **Euler** paths or circuits?



Graph 1



Graph 2



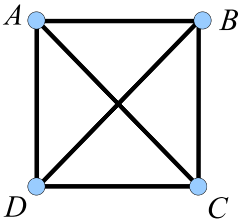
Graph 3

Different Hamilton circuits

How many **different** Hamilton circuits does a particular graph have?

Two Hamilton circuits are different if they reach the vertices of a graph in different relative orders.

For example, describe two **different** Hamilton circuits in the following graph:



Different Hamilton circuits

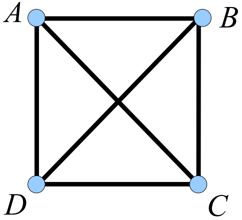
How many **different** Hamilton circuits does a particular graph have?

This can be a very easy question, or a very difficult one, depending on the graph. We can answer this question for complete graphs.

A **complete graph** is a graph where every vertex is connected to every other vertex. How many edges does such a graph have?

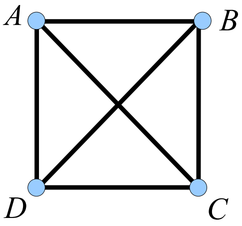
Hamilton circuits in a complete graph

The complete graph with four vertices (called K_4) looks like this:



Find all the **different** Hamilton circuits in K_4 starting at point A.

Hamilton circuits in a complete graph



Note that starting at point A, we have three different choices of vertex to head to next: B, C, or D. Once we pick one of them, we have two choices (since we cannot go back to a vertex we have already left) and then there is only one choice. Each choice defines a different Hamilton circuit.

Hamilton circuits in a complete graph

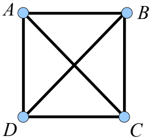
What about the complete graph with N vertices (K_N)? How many different Hamilton circuits does it have?

Number of Hamilton circuits in K_N

There are $(N-1)!$ distinct Hamilton circuits in K_N.

Finding an optimal route - revisited

Consider the complete graph K_4 again, this time with each route weighted.

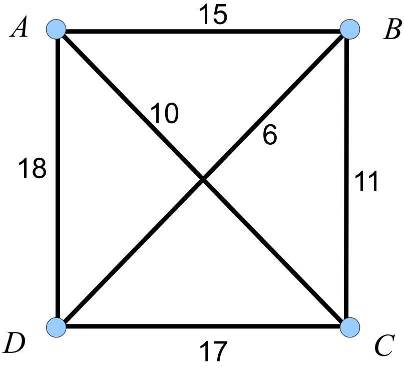


Consider the following table of weights:

	A	B	C	D
A	–	15	10	18
B	15	–	11	6
C	10	11	–	17
D	18	6	17	–

Find the weights of every Hamilton circuit. Do some have the same weights? Which one(s) is/are optimal?

Finding an optimal route - revisited



Brute-force algorithms

In the previous example, we found **all** the Hamilton circuits and calculated their weights (or costs), then picked the lowest one. This is called a **brute-force** algorithm.

This is not great, since this graph is a complete graph with five vertices - that's 24 distinct Hamilton circuits! However, we only need to calculate the weights of 12 of those circuits...why?

Nearest-Neighbor algorithm

The **nearest-neighbor** algorithm is simple and intuitive. At any vertex, pick the cheapest route to travel next to a vertex you have **not yet** visited.

Nearest-Neighbor algorithm

What is the route generated by the nearest-neighbor algorithm starting at point A? Is it different if you start at point B?

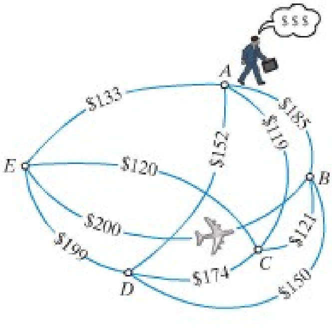


FIGURE 6-7

Nearest-Neighbor algorithm

Since we get different weights for routes with different starting points under this algorithm, the **repetitive nearest neighbor** algorithm repeats the nearest neighbor algorithm starting at different vertices, so that you have a few choices of routes, and can pick the cheapest.

Repetitve Nearest-Neighbor algorithm

Find the weights of routes generated by this algorithm starting at points C, D, and E. Which is optimal?

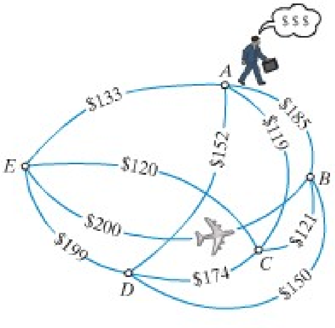


FIGURE 6-7

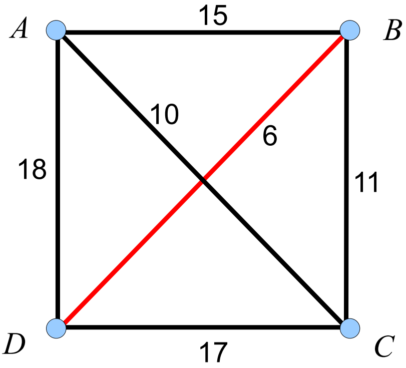
Cheapest-Link algorithm

The cheapest link algorithm for a graph with N vertices is as follows:

- Pick the **cheapest link**, which is the edge with the smallest weight (wherever this is on the graph). Mark it.
- Then, pick the next cheapest link, and mark it.
- Keep picking the next cheapest links, **unless** they close a circuit too soon or they create three edges coming out of one vertex.
- When you have $N - 1$ edges marked, close the circuit by adding the Nth edge.

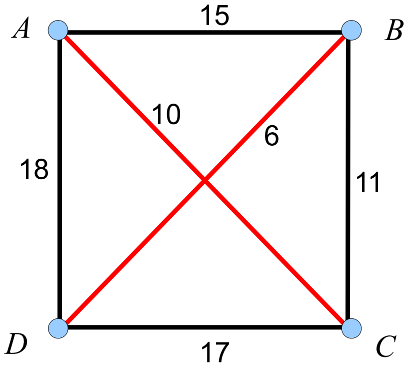
Cheapest-Link algorithm

Step one: Mark the cheapest link:



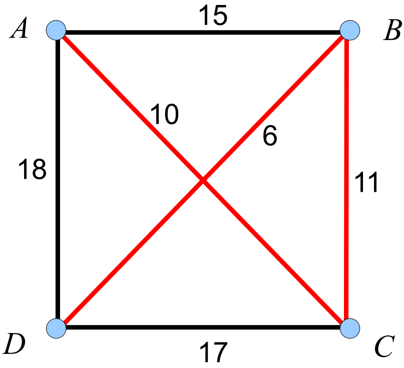
Cheapest-Link algorithm

Step two: Mark the next cheapest link:



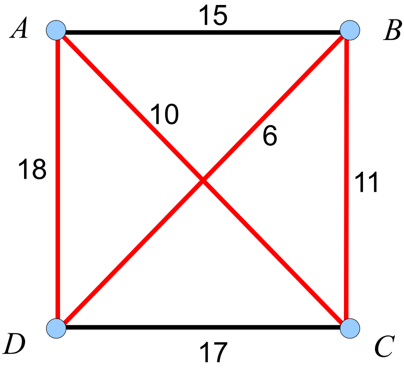
Cheapest-Link algorithm

Step three:



Cheapest-Link algorithm

Step four: close the circuit.



Cheapest-Link algorithm

What route is generated by the cheapest-link algorithm?

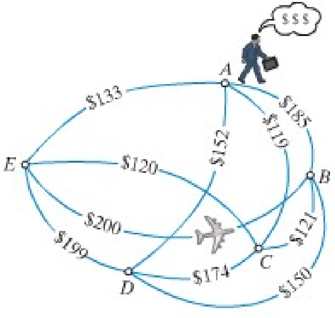


FIGURE 6-7