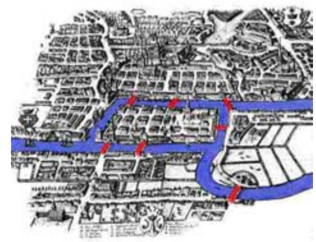
Algorithm Design: Graph Theory & Algorithms

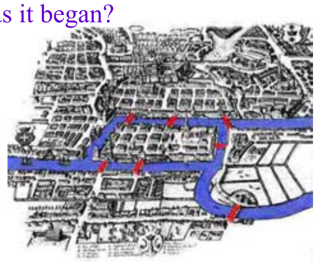
The 7 Bridges of Königsberg

- In the former city of Königsberg, East Prussia, the river Preger had 7 bridges:

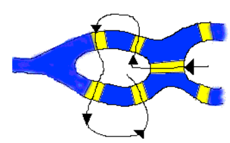


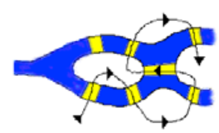
The 7 Bridges of Königsberg

- Can the 7 bridges be traversed in a single tour without doubling back, so that the tour ends in the same place as it began?



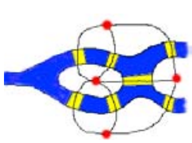
Some brave attempts…

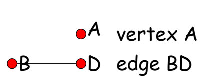




The 7 Bridges of Königsberg

- Leonard Euler's solution, 1736

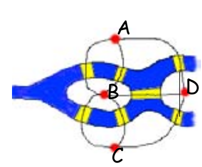


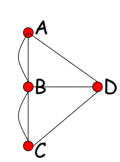


edges are bridges that connect landmasses (vertices)

The 7 Bridges of Königsberg

- Leonard Euler's solution, 1736



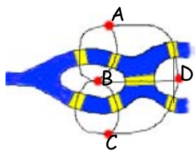


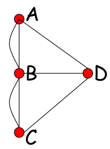
graph

A graph is a set of vertices and edges connecting these vertices

Intuition

- When traveling along a tour, whenever you enter a landmass (vertex) by a bridge (edge), you must exit by another bridge (edge)

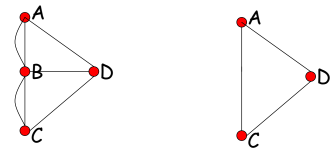




graph

Vertex Degree

- The *degree* of a vertex is the number of edges incident to it

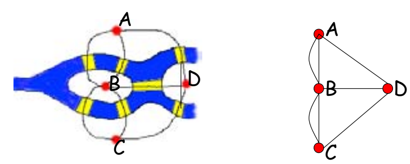


Euler's Theorem

- ***If a graph has an Euler tour then each vertex must be of even degree***

- *Proof: For each vertex that you enter during the tour via an edge, you must also be able to exit via an edge. Since you have to finish at the same vertex where you started the tour, the same holds for the start/finish vertex. Thus, if the graph has an Euler tour, each vertex has as many edges incident to it that are used to enter this vertex as are used to exit. Thus, the degree of each vertex is even.*

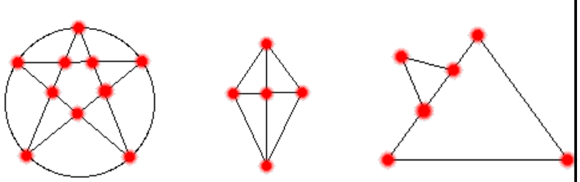
Euler's Theorem



- So, in Königsberg, there is NO Euler tour of the 7 bridges! (as there are vertices with odd degree)

Euler Tours

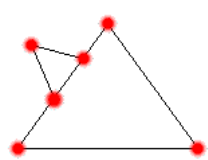
- Do the graphs below have Euler Tours?

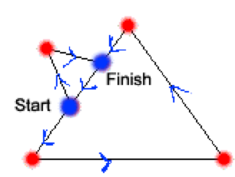


Yes No No

Euler Paths

- Less restrictive problem: must visit all edges, but can start and finish at *different* vertices
- Can this graph have an Euler path?



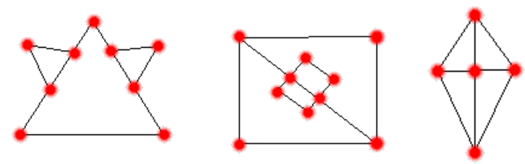


Euler Paths

- ***If a graph has an Euler path, then all vertices except the start and end vertices must of even degree***
- *Proof: As we traverse our graph along an Euler path, for each vertex that we enter via an edge, there must be another unused edge incident to this vertex to exit the vertex. As the start and finish vertices need not be the same, the degree of these vertices may be odd, while the degrees of all other vertices must be even.*

Euler Paths

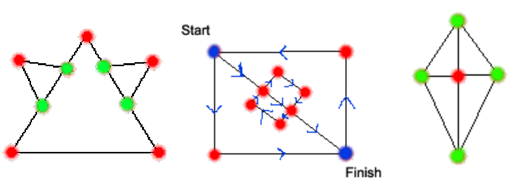
- Do the graph below have Euler paths?



No Yes No

Euler Paths

- Do the graph below have Euler paths?



4 vertices of odd degree Yes 4 vertices of odd degree

Graph Theory

- The 7 bridges of Königsberg was the first problem solved by using of a *graph* representation
- Graphs serve as powerful abstract models representing relationships between pairs of entities
- Models of:
  - transportation networks
  - decision-making process

Paving Smallville

Pave roads such that all intersections are reached and overall cost minimized

Paving Smallville

- Naïve solution:

  Try all possibilities (each of 20 roads either paved or not paved) and check if all intersections reached, while also noting cost

  Number of possible "pavings" to consider with 20 roads: 2^{20} = 1,048,576 possibilities

- This kind of algorithm is called *brute-force* because it exhaustively considers all possibilities

Paving Smallville

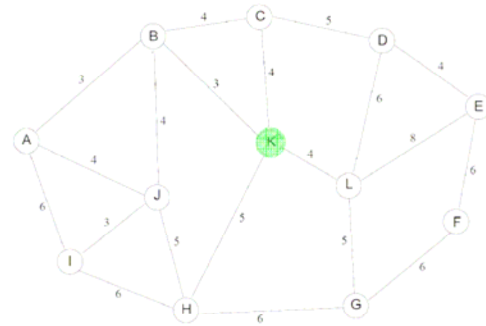
- Prim's Algorithm:

  1. Start at arbitrary intersection
  2. Pave cheapest road leaving this intersection
  3. Keep paving cheapest road that adds a new intersection to the paved set of roads

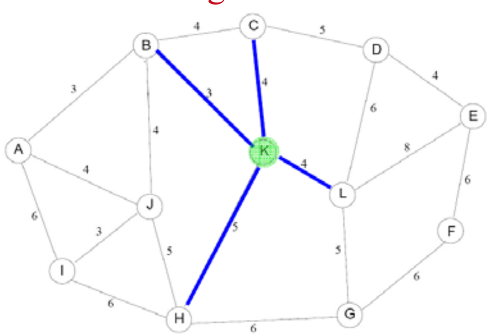
  For each of the 12 intersections added, consider at most 20 edges, so don't make more than 12 x 20 = 240 decisions

- This kind of algorithm is called *greedy* because it makes the most appealing decision at each step and never reconsiders its choice

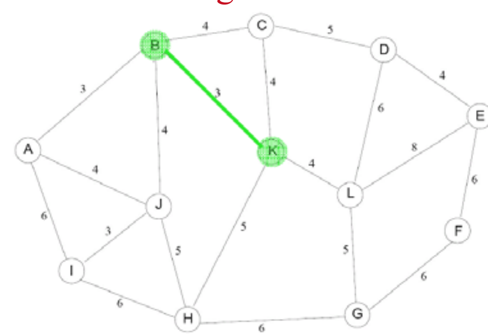
Paving Smallville



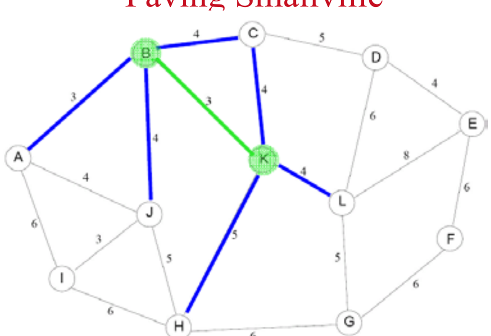
Paving Smallville



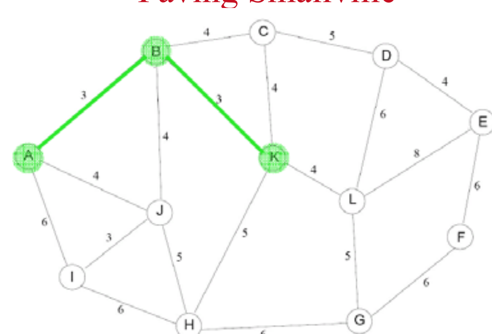
Paving Smallville



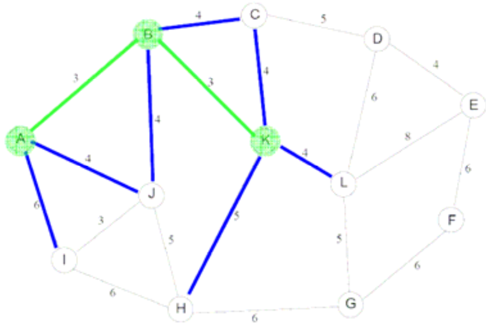
Paving Smallville



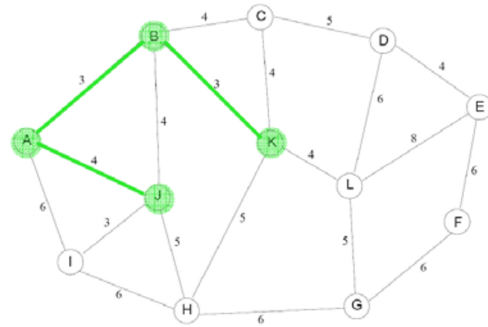
Paving Smallville



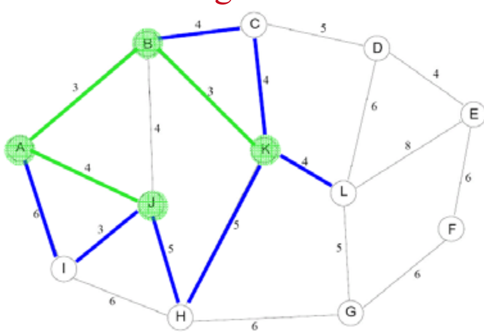
Paving Smallville



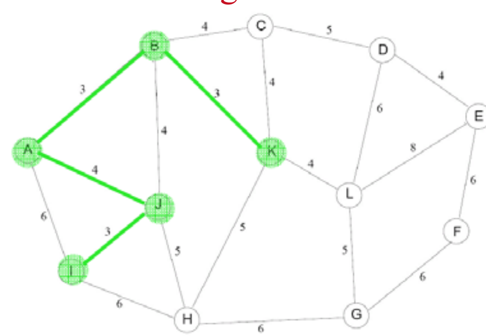
Paving Smallville



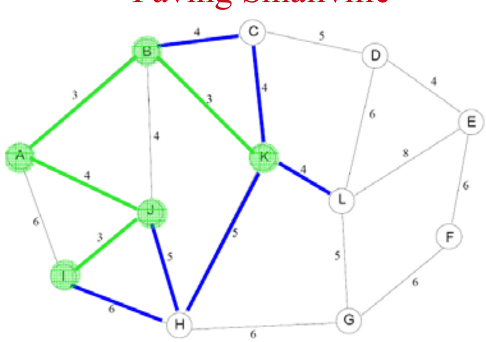
Paving Smallville



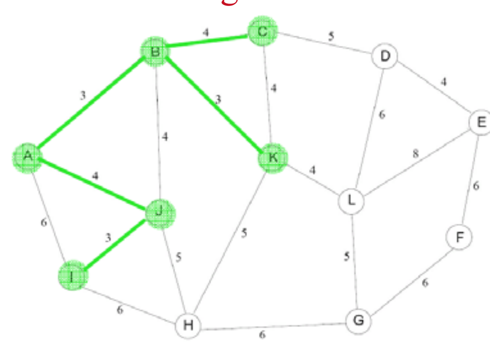
Paving Smallville



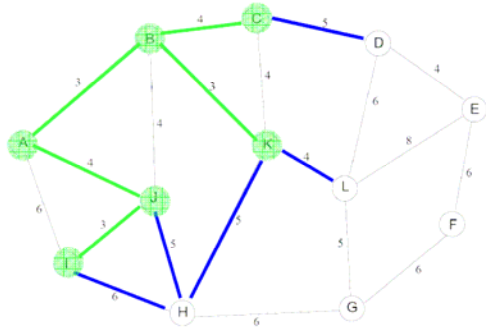
Paving Smallville



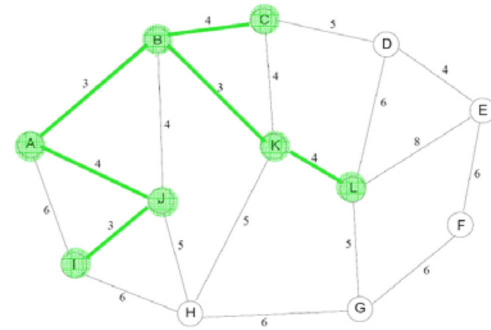
Paving Smallville



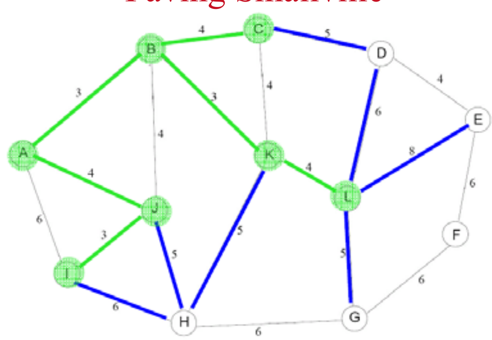
Paving Smallville



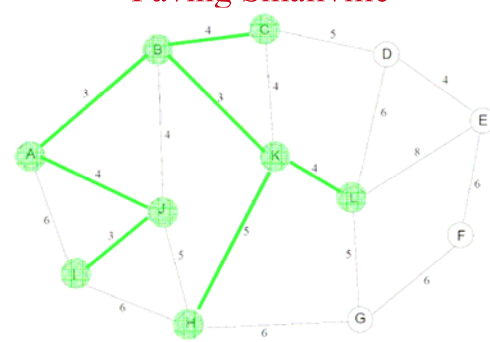
Paving Smallville



Paving Smallville

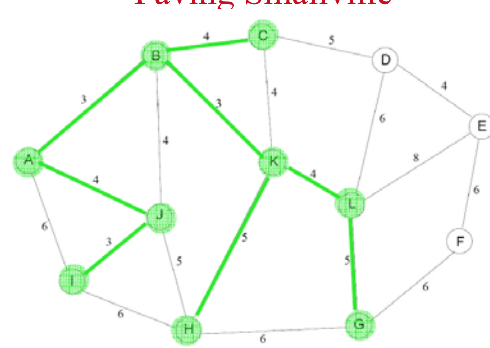


Paving Smallville

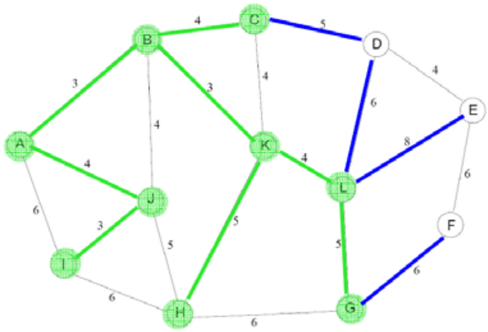


Paving Smallville

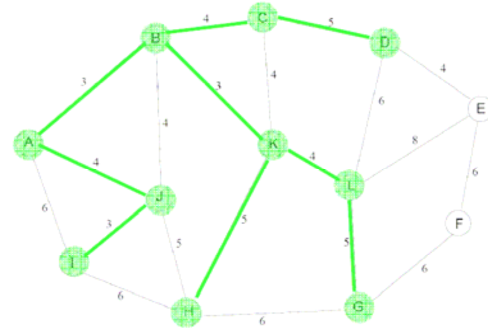
Paving Smallville



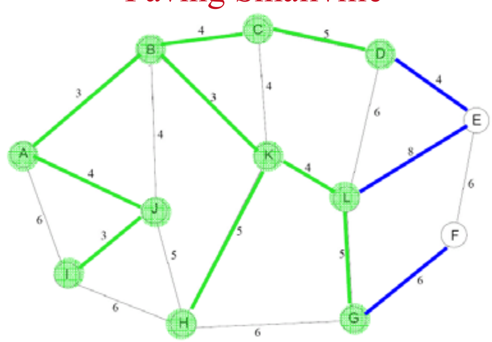
Paving Smallville



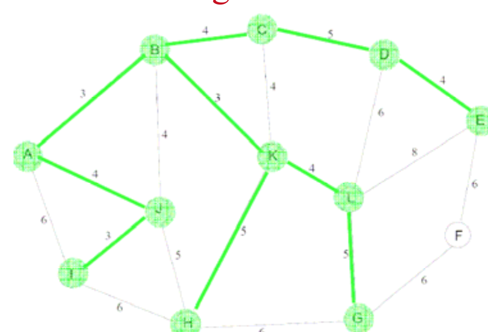
Paving Smallville



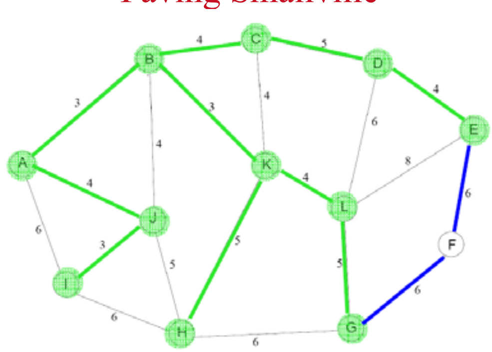
Paving Smallville



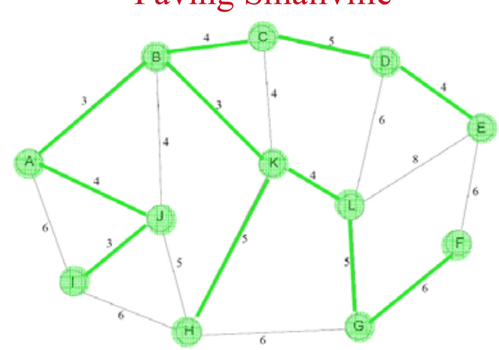
Paving Smallville



Paving Smallville



Paving Smallville

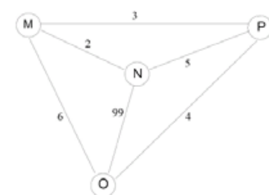


Paving Smallville

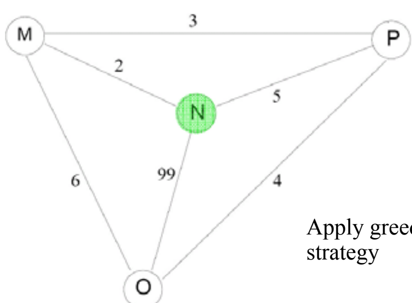
- The *greedy* strategy chooses the best road to pave at each step of the algorithm
- The resulting "paving" is optimal (no cheaper paving is possible by paving different roads)
- Does the greedy strategy work well on all problems?

Traveling Salesman Problem

- Traveling salesman must travel to every city along the cheapest route
- But he cannot visit a city more than once and he must come back where he started

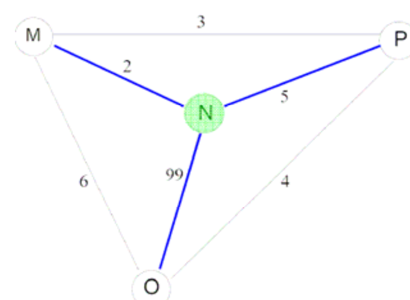


Traveling Salesman Problem

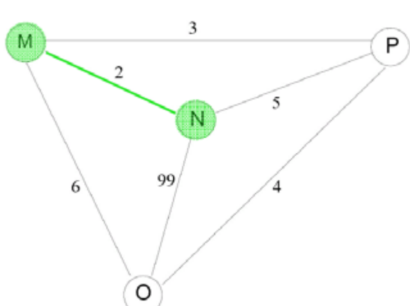


Apply greedy strategy

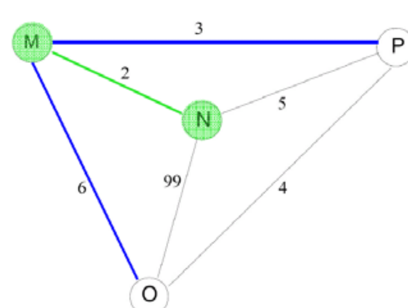
Traveling Salesman Problem



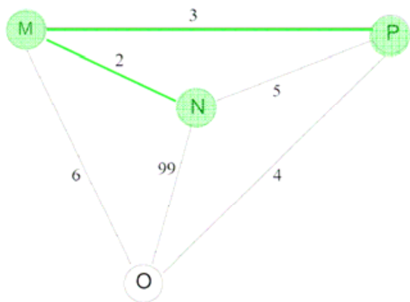
Traveling Salesman Problem



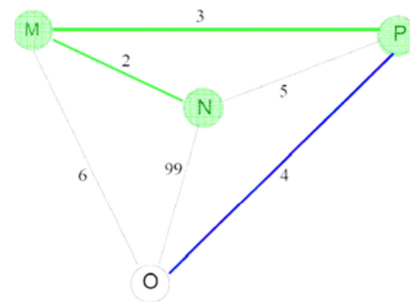
Traveling Salesman Problem



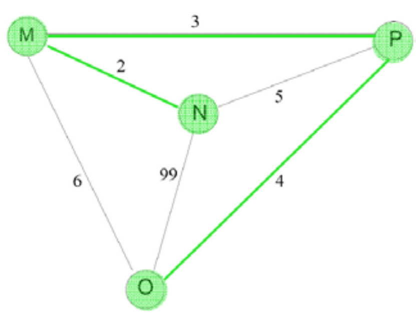
Traveling Salesman Problem



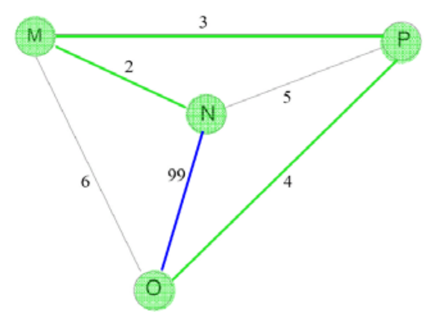
Traveling Salesman Problem



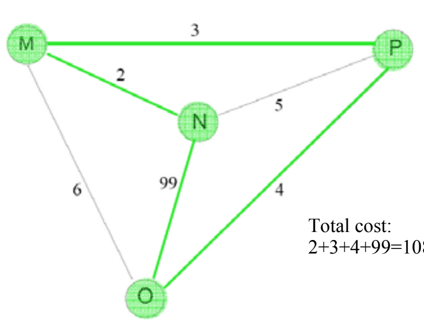
Traveling Salesman Problem



Traveling Salesman Problem

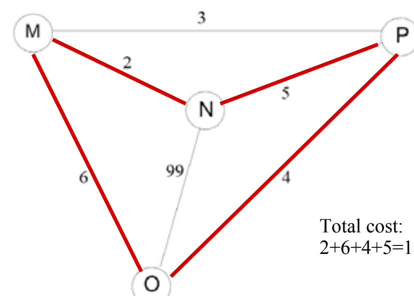


Traveling Salesman Problem



Total cost:
2+3+4+99=108

Traveling Salesman Problem



Total cost:
2+6+4+5=17

Traveling Salesman Problem (Trivia)

- This is a significantly more difficult problem than "Paving Smallville"
- A TSP tour for 15,112 German cities was found using parallel processors that would take a single 500 MHz CPU 22.6 years to complete.
- Finding a fast algorithm for TSP would be revolutionary to computer science as it would show a certain fundamental conjecture false
- Humans quite good at TSP