Week 11 - Monday

COMP 2100

Last time

- What did we talk about last time?
- Euler practice
- Network flow
- Started B-trees

Questions?

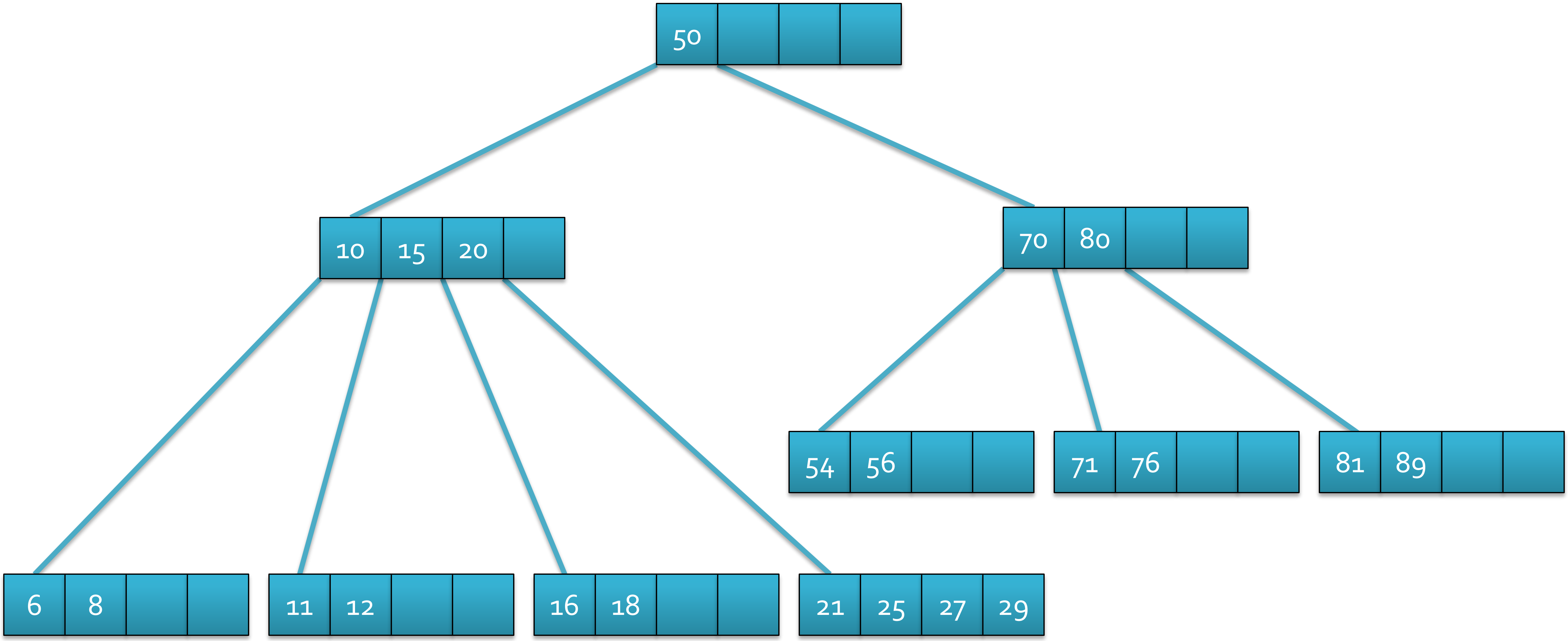
Project 3

B-trees

B-tree definition

- A B-tree of order ***m*** has the following properties:
  1. The root has at least two subtrees unless it is a leaf
  2. Each nonroot and each nonleaf node holds ***k*** keys and ***k*** + 1 pointers to subtrees where $m/2 \leq k \leq m$
  3. Each leaf node holds ***k*** keys where $m/2 \leq k \leq m$
  4. **All leaves are on the same level**

B-tree of order 4



B-tree practice

- Insert the following numbers:
  - 86 69 81 15 100 94 8 27 56 68 92 89 38 53 88

Deletions

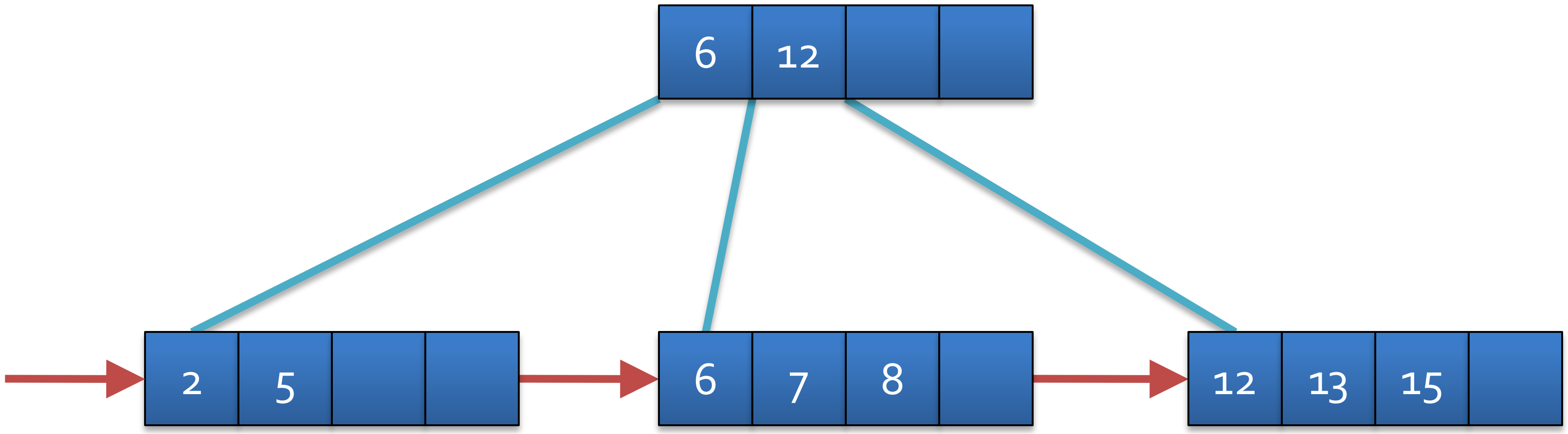
- When the list of keys drops below half ***m***, we have to redistribute keys
- In the worst case, we have to delete a level

B*-tree

- Instead of requiring every non-root node to be half full, every non-root node must be at least 2/3 full
- Key redistribution becomes more complex
- However, the tree is fuller

B⁺-tree

- Essentially, make a B-tree such that all the leaves are tied together in a linked list
- It's also necessary that all keys in a B⁺-tree appear as leaves
- Some other variations are possible, but we'll end the list here



Hamiltonian Style

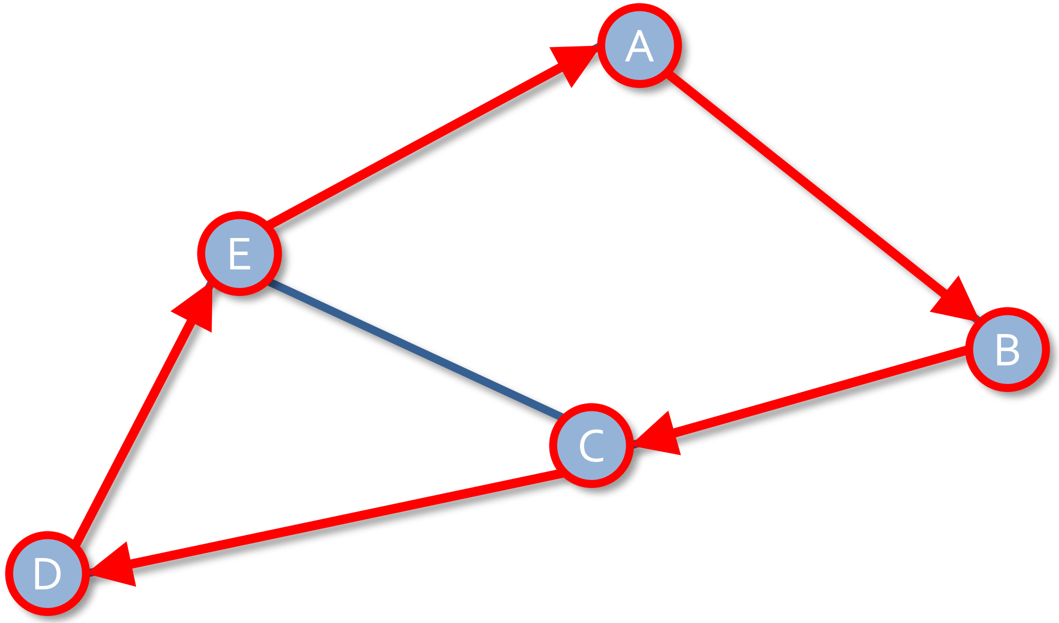
Are you ready to get Hamiltonian with it?

Hamiltonian cycle

- The **Eulerian Cycle** problem asked if you could start at some node, cross every **edge** once, and return to the start
- The **Hamiltonian Cycle** problem asks if you can start at some node, visit every **node** only once, and return to the start
- In other words, find a **tour**
- Sounds easy, right?

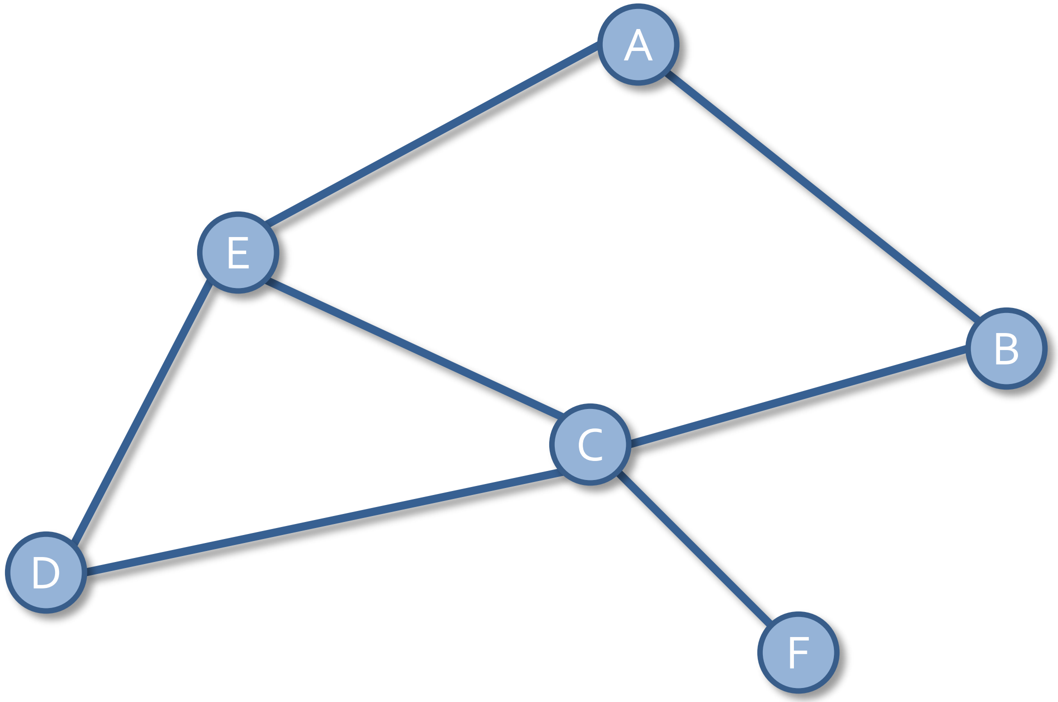
Find the Hamiltonian Cycle

- On this graph:



Find the Hamiltonian cycle

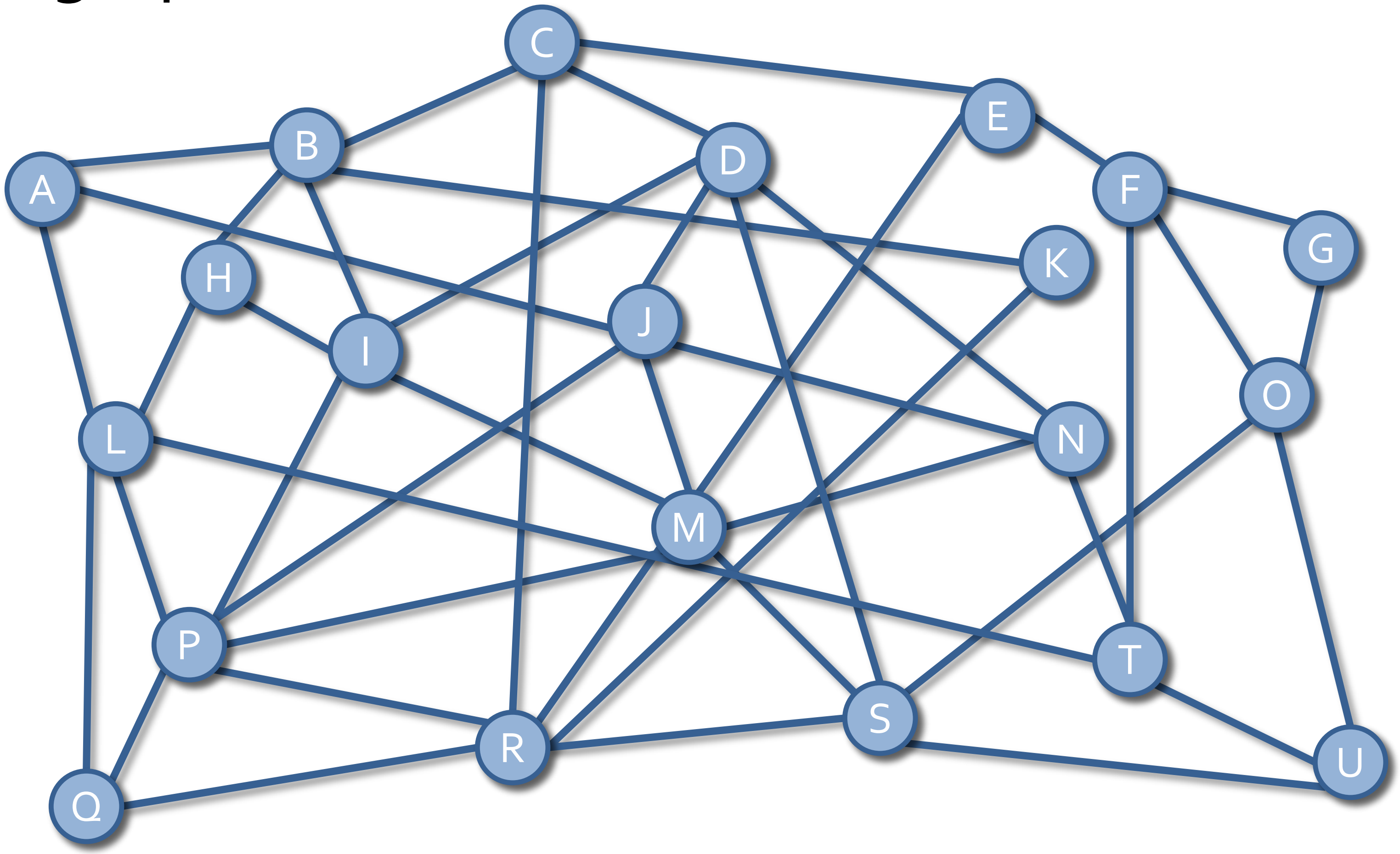
- Now, on this graph:
- There isn't one!



Find the Hamiltonian cycle

- Now, on this graph:

Hmmm…



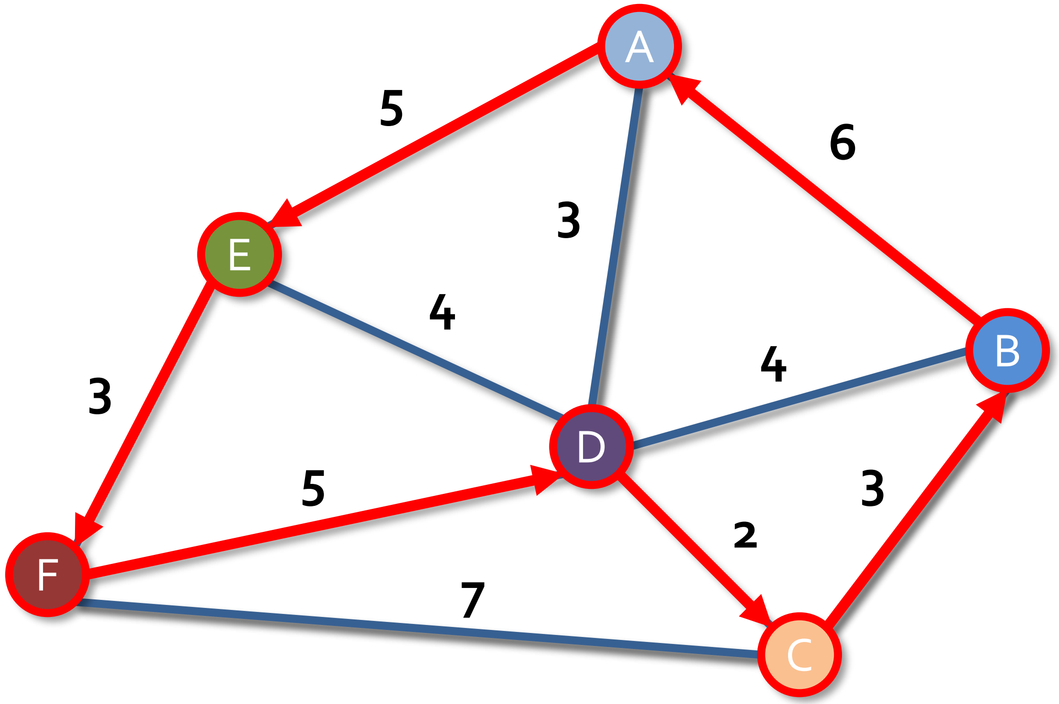
TSP

Traveling salesman problem:
Hamiltonian cycle meets shortest path

- Given a graph, find the shortest tour that visits every node and comes back to the starting point
- Like a lazy traveling salesman who wants to visit all possible clients and then return to his home

What's the shortest tour?

- Find the shortest TSP tour:
  - Starts anywhere
  - Visits all nodes
  - Has the shortest length of such a tour



How can we always find the shortest tour?

- Strategies:
  - Always add the nearest neighbor
  - Randomized approaches
  - Try every possible combination

- Why are we only listing strategies?

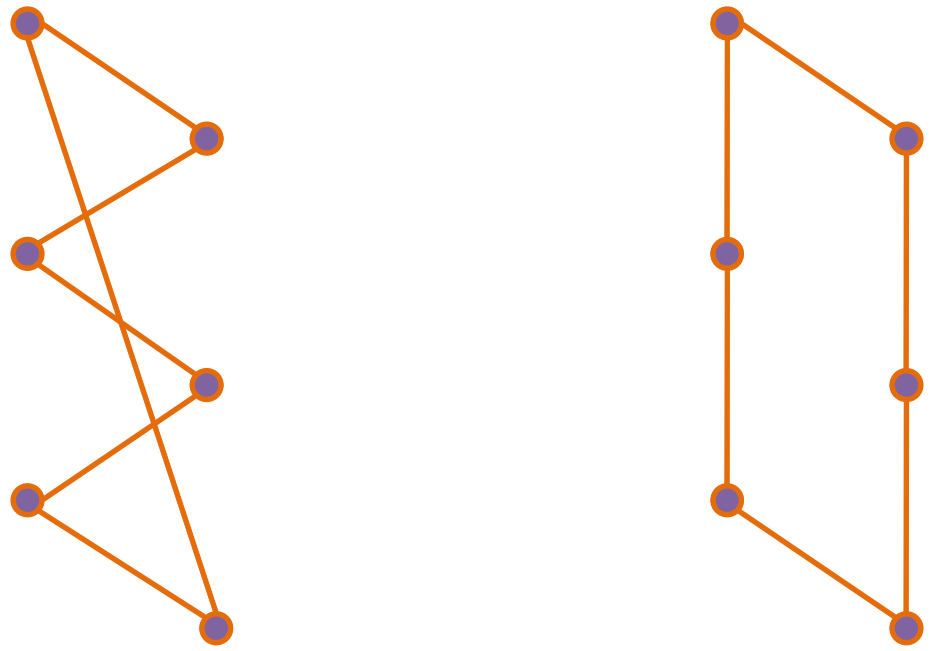
Best Solution to TSP

- So far we haven't given a foolproof solution to TSP
- Let's look at two possibilities:
  - Greedy Solution: Pick the closest neighbor
  - Brute Force: Try all possibilities

Greedy Doesn't Work

- We are tempted to always take the closest neighbor, but there are pitfalls

Greedy Optimal



Brute force is brutal

- In a completely connected graph, we can try any sequence of nodes
- If there are n nodes, there are (n – 1)! tours
- For 30 cities, 29! = 8841761993739701954543616000000
- If we can check 1,000,000,000 tours in one second, it will only take about 20,000 times the age of the universe to check them all
- We will (eventually) get the best answer!

What's the best time for TSP?

- No one knows how to find the best solution to TSP in an efficient amount of time
- For a general graph, no good approximation even exists
- For a graph with the triangle inequality, there is an approximation that yields a tour no more than 3/2 the optimal
- Some variations on the problem are easier than others

Hard problems

- Is TSP the hardest problem there is?
- Are there other problems equally hard?
- How do we compare the difficulty of one problem to another?

NP-completeness

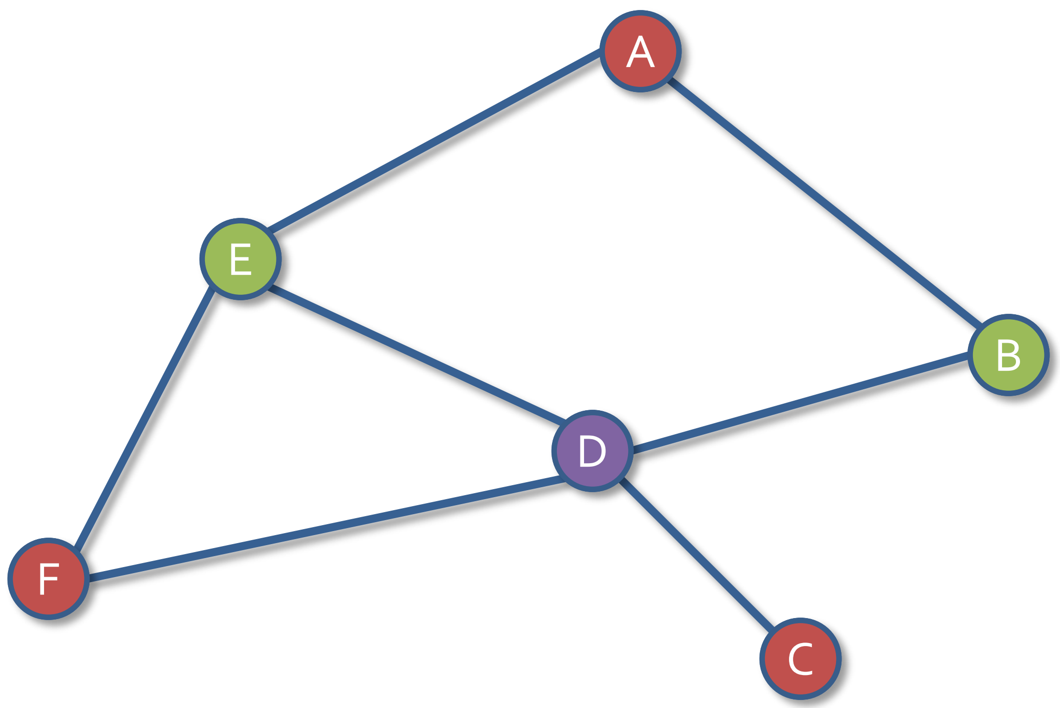
- TSP is one of many problems which are called **NP-complete**
- All of these problems share the characteristic that the only way we know to find best solution uses brute force
- All NP-complete problems are reducible to all other NP-complete problems
- An efficient solution to one would guarantee an efficient solution to all

NP-complete problems on graphs

- Traveling Salesman Problem
- Hamiltonian Cycle (and Path)
- Let's see just a couple more...

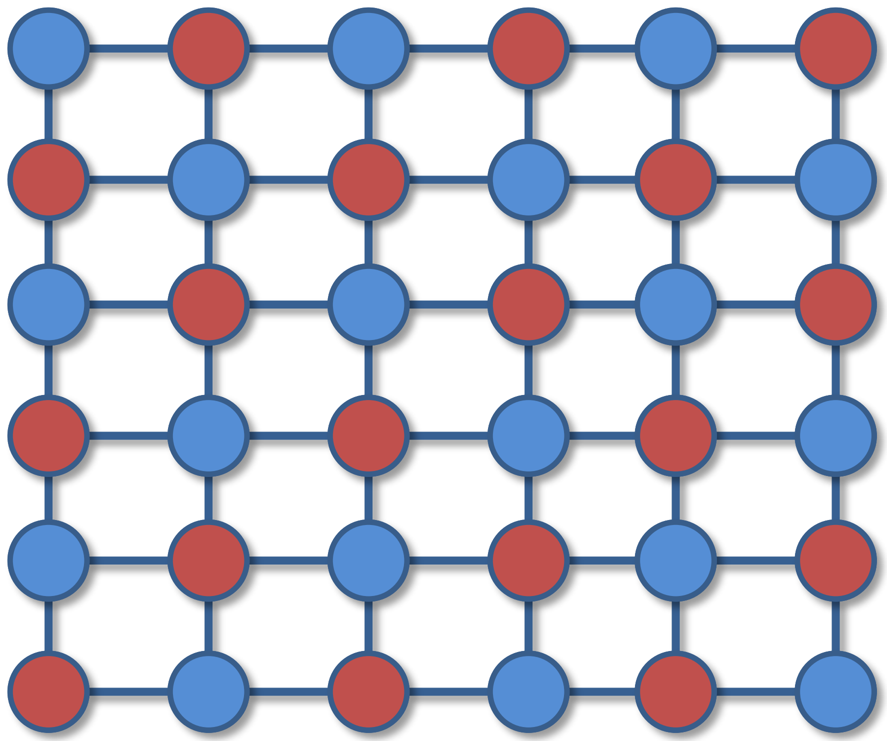
Graph coloring

- Find the smallest number of colors for coloring nodes such that no two adjacent nodes have the same color



Graph coloring

- What about this graph?

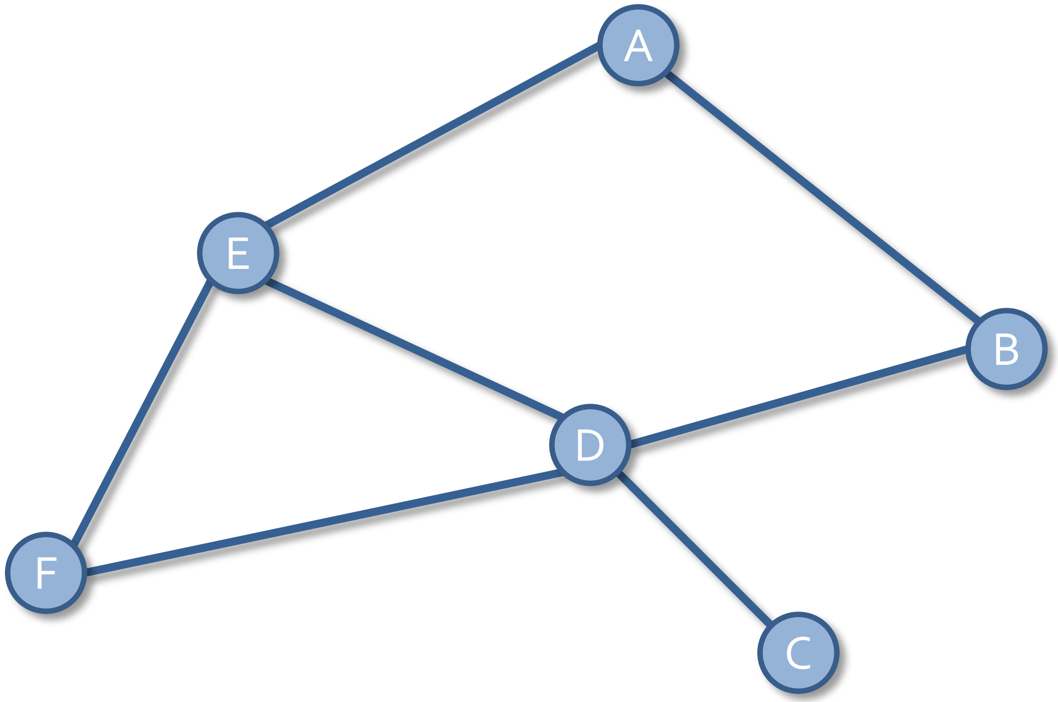


Graph coloring

- Like TSP and Hamiltonian Cycle, large graphs get really hard to find the smallest coloring for
- It might not be obvious, but graph coloring has practical applications
- Perhaps the most common example is register allocation inside of a compiler

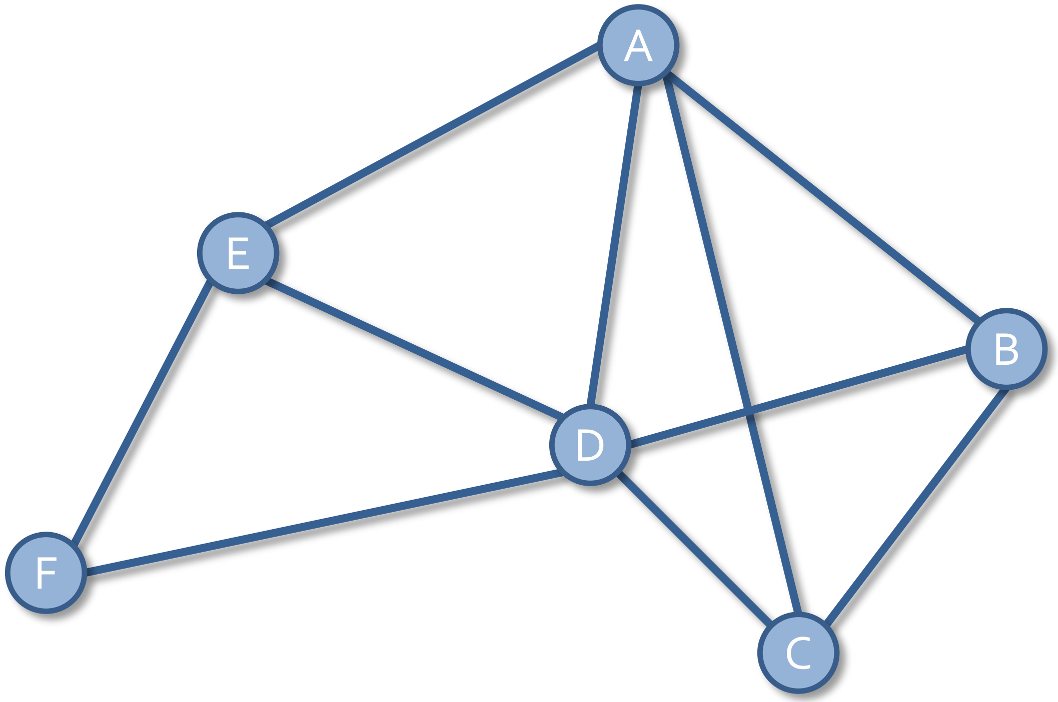
Maximum clique

- Find the largest complete subgraph in a given graph
- This graph has a clique of size 3



Maximum clique

- What about this graph?
- This graph has a clique of size 4
- As before, a large graph can be very difficult



Knapsack problem

- Not all NP-complete problems are graph problems
- The knapsack problem is the following:
  - Imagine you are Indiana Jones
  - You are the first to open the tomb of some long-lost pharaoh
  - You have a knapsack that can hold ***m*** pounds of loot, but there's way more than that in the tomb
  - Because you're Indiana Jones, you can instantly tell how much everything weighs and how valuable it is
  - You want to find the most valuable loot that weighs less than or equal to ***m*** pounds

Upcoming

Next time…

- Finish NP-completeness and intractability

Reminders

- Keep working on Project 3
  - Due Friday by midnight
- Review chapters 3 and 4 for Exam 2
  - **Next Monday!**
  - We'll review for Exam 2 on Friday