LEC 21

CSE 373

Traveling Salesperson Problem (TSP)

BEFORE WE START

No practice problems today! Just Q&A!

Post questions about TSP (or anything) in Zoom chat.

Music: Whitney

Instructor Hunter Schafer

TAs
Ken Aragon
Khushi Chaudhari
Joyce Elauria
Santino Iannone
Leona Kazi
Nathan Lipiarski
Sam Long
Amanda Park
Paul Pham
Mitchell Szeto
Batina Shikhalieva
Ryan Siu
Elena Spasova
Alex Teng
Blarry Wang
Aileen Zeng

Learning Objectives

After this lecture, you should be able to...

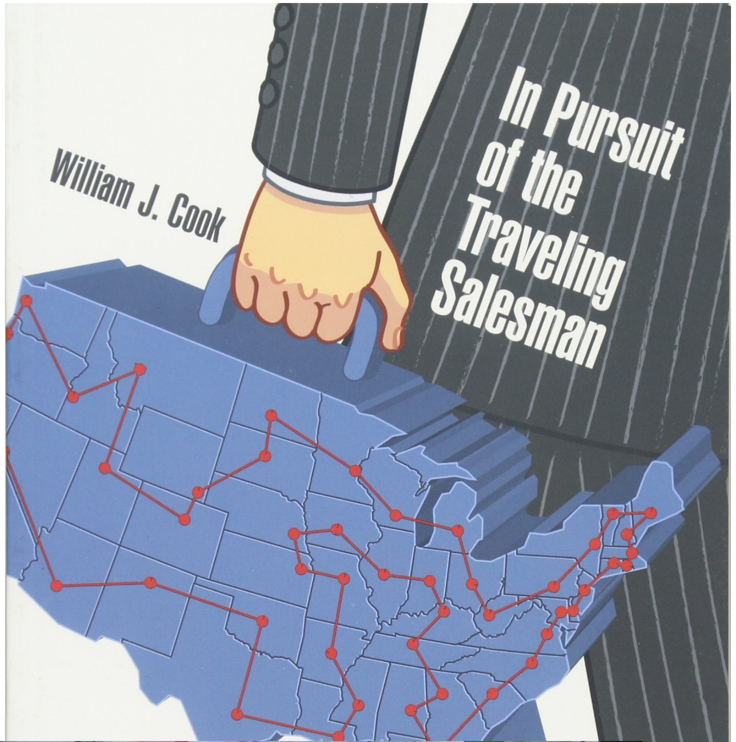
This lecture is optional! It will cover some cool and modern applications of the things we have learned so far but is not part of the ”core” material for CSE 373. Therefore, we won’t highlight learning objectives since this is focused more on showing off a cool topic!

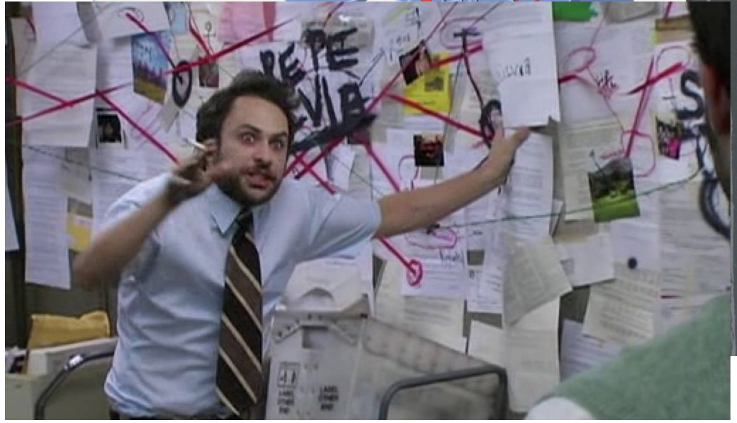
Lecture Outline

- Traveling Salesperson Problem
- Approximation Algorithms
- Recent* Developments

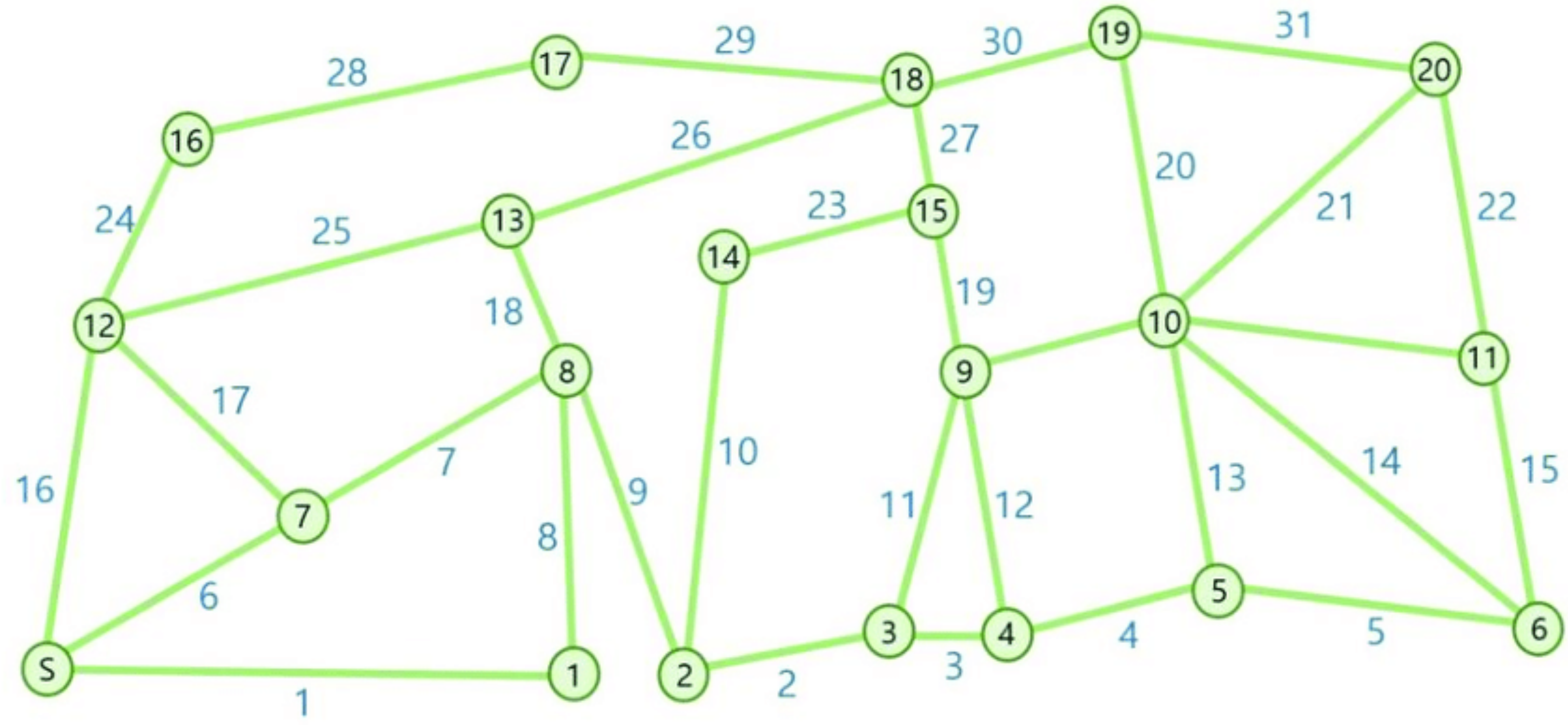
Traveling Salesperson Problem

- Congrats! You got hired as the new super delivery person at the USPS! Your first job is to deliver your letters across the country.
- **Goal**: In what order should you visit all of the cities to minimize the total distance of your trip?
  - You'll hear this also called the "Traveling Salesman Problem" or TSP for short.
- *Problem*: There are a LOT of possible paths





Try it Out


S
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
28
29
30
31
26
27
24
25
23
20
21
22
18
19
17
16
7
8
9
10
11
12
13
14
15
6
1
2
3
4
5

TSP More Formally

Input: A undirected, weighted graph G**.**

- Commonly assume G is complete (pair-wise connections)

Output: A path of minimum total cost from some start vertex that visits every node in the graph exactly once.

- Start vertex will be visited twice, once at beginning, and once at end
- Also called a “Hamiltonian cycle of minimum weight”

Aside Metric TSP

Many variations of this problem exist. Today, we will focus on one called "Metric TSP" which assumes the graph represents something like a bunch of cities and the edge weights are distances between them.

The "Metric" requirement means that the distances should behave normally:

- All the edge-weights are non-negative (0 if an edge from a node to itself).
- The edge weights follow the "triangle inequality"

$$d(A, C) \leq d(A, B) + d(B, C)$$

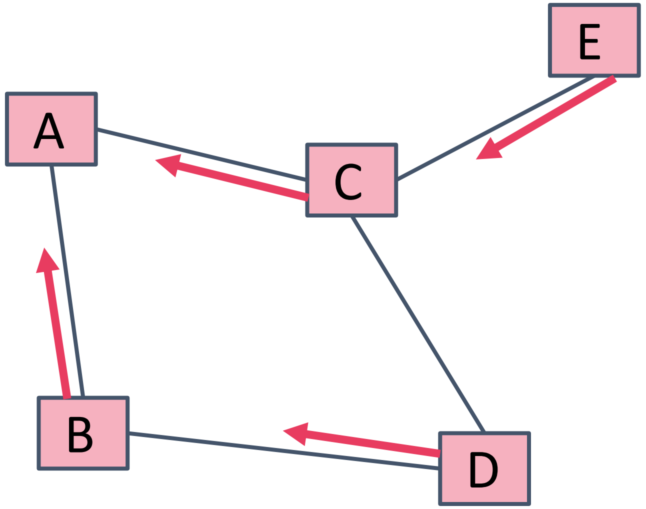
  - Intuition: "The hypotenuse is shorter than the sum of its sides"

Idea 1 Something Something Dijkstra's

Finding shortest paths in a graph? I know! Let's use Dijkstra's!

Algorithm:

- Pick a start vertex *v*
- Run Dijkstra's from *v* to get shortest path tree of graph
- ???
- ???



Not clear how to turn this shortest path tree into a solution! It gives an idea of where to start, but once you explore one path, you won't know how to finish the path that visits the rest of the graph.

An Exact Solution

The simplest algorithm is to try all possible paths

- Takes $\mathcal{O}(|V|!)$ time 😱

So far, the only algorithms we know of to solve TSP exactly involve some kind of “choose-explore-unchoose” pattern of recursive backtracking.

- Some of the best algorithms take something like $\mathcal{O}\left(2^{|V|}\right)$ 😱

Most people think that the best we can hope for is exponential.

- This means this problem isn’t tractable for even moderately sized inputs. With *only* 85k cities, would take something like 130 years to compute an exact solution on a fast computer (Applegate et al. (2006)).
- More on why computer scientists are pessimistic on finding a better solution on Monday! It has to do with this problem called “P vs. NP.”

Lecture Outline

- Traveling Salesperson Problem
- **Approximation Algorithms**
- Recent* Developments

Idea 2 Approximation Algorithms

If we don’t think we can solve the problem exactly, is there hope of finding an efficient algorithm that yields an answer that is “close enough”?

This is the idea behind **approximation algorithms.** We give up on exactness in the hope for efficiency gains.

- For example, if the true TSP solution was a 100-mile journey, is it that big of a deal if our approximation reports and answer that is 110 miles long?

Approximation Ratios

Define the **approximation ratio** of our proposed approximate solution *P* when compared to the optimal solution *P**. We call an approximation algorithm that has an approximation ratio of α an **α-approximation.**

$$cost(P) \leq \alpha \cdot cost(P^*)$$

This is a property that constrains the gap in quality between your approximation and the globally optimal solution. The lower α, the better your approximation will be!

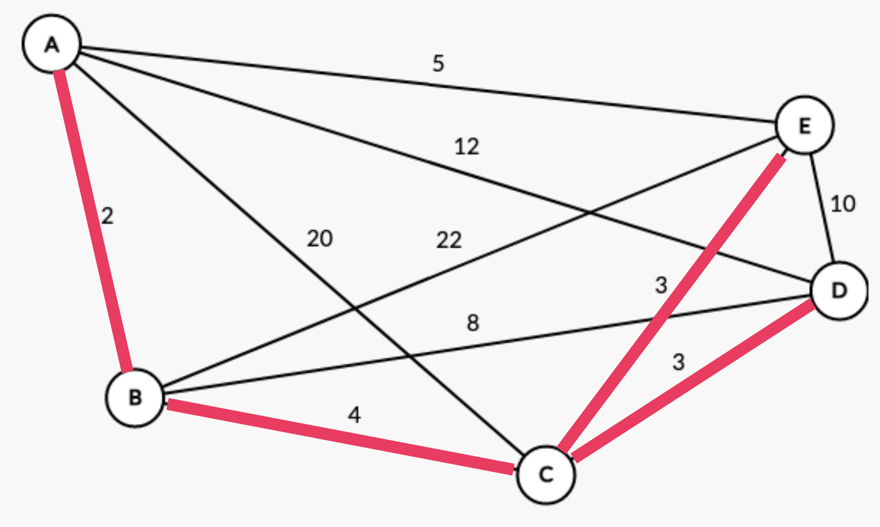
Examples:

- 2-approximation: If the true solution is 100 miles, we will guarantee an approximate solution with cost at most 200 miles.
- 1.1-approximation: If the true solution is 100 miles, we will guarantee an approximate solution with cost at most 110 miles.

Using MST to Approximate TSP

There is an approximation algorithm to TSP that uses a MST.

1. Start with Prim's to find an MST.
2. "Trace" a DFS on the MST to explore graph.
   - Build up path as you visit
   - Trace lists out nodes both times you visit it.
3. Skip vertices already visited. Use edges not included in MST when skipping.



Trace of DFS: A, B, C, D, C, E, C, B, A

Solution: A, B, C, E, A with total cost 24.

- In this case, the optimal solution is A,B,D,C,E,A with cost 21

Does this work?

Yes! This MST approach is a 2-approximation of the TSP problem!

- No matter what the optimal solution is, this algorithm will find one that's no worse than twice as bad.

Intuition: In most cases, there should be a lot of overlap with an MST and a solution to the TSP (using some of the smallest edges in the graph).

Natural question then is: Can we do better than a 2-approximation?

Lecture Outline

- Traveling Salesperson Problem
- Approximation Algorithms
- **Recent* Developments**

Christofides' Algorithm (1976)

Very similar in spirit to the MST approximation, but does a much better job at converting the MST to a path to approximately solve TSP.

- Has some added complication of nodes in the MST with even/odd degree.

Christofides' Algorithm is a 1.5-approximation to TSP.

- If the true solution is 100 miles, Christofides can find an approximation with cost at most 150 miles.

Natural question then is: Can we do better than a 1.5-approximation?

ALGORITHMS

Computer Scientists Break Traveling Salesperson Record

After 44 years, there's finally a better way to find approximate solutions to the notoriously difficult traveling salesperson problem.

**1.499999999999999999999999999999999
999-approximation**

Authors: Anna Karlin, Nathan Klein, Shayan Oveis Gharan

Why This Matters

Who cares about an epsilon improvement? Lots of theoretical computer scientists!

- TSP is one of the most well-studied problems in CS history. NO ONE was able to improve Christofides' algorithm for over 40 years!
  - People were losing hope that there was a better way.
  - This is a proof-of-concept that something better exists! A new hope!
- New theoretical machinery to prove result called "geometry of polynomials". Potentially a useful tool to solve other unsolved problems.
- Hopefully learn a bit more about computation.

How it Works

Fairly similar to Christofides' algorithm but doesn't use an MST. Uses a *randomized algorithm* to generate a random tree, and then turns that tree into a path.

A lot of the work was proving that their algorithm was, in fact, better!

- A common challenge for computer science work is not just coming up with an algorithm, but being able to prove that it works in the way you claim!

Resources to Learn More

- Wikipedia has a pretty good overview of the problem, its variations, and algorithms to solve it.
- More formal walkthrough of MST approximation and Christofides algorithm here.
- Quanta Magazine article on TSP breakthrough and its history