TSP: Past and Present

Richard Xu

Harvard University

February 11, 2021

Prepared for Harvard CS 124.5, Week 2

Roadmap

1. Background and History about TSP
2. Exact Solutions
3. Heuristics
4. Approximation Algorithms

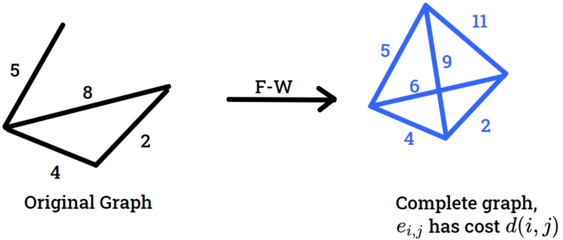
Note: In this talk, *efficient* refers to polynomial time.

What is the TSP Problem?

1. In the *Traveling Salesman Problem* (TSP), you are given a weighted graph $G = (V, E)$.
2. Goal: Find a cycle that visits each vertex at least once with the least cost.
3. Exercise 1: Think of some scenarios where this problem is useful.

What is the TSP Problem?

1. In the *Traveling Salesman Problem* (TSP), you are given a weighted graph $G = (V, E)$.
2. Goal: Find a cycle that visits each vertex at least once with the least cost.
3. Exercise 1: Think of some scenarios where this problem is useful.
4. WLOG, consider *complete graphs* where distances satisfy *triangle inequality*.
5. How?

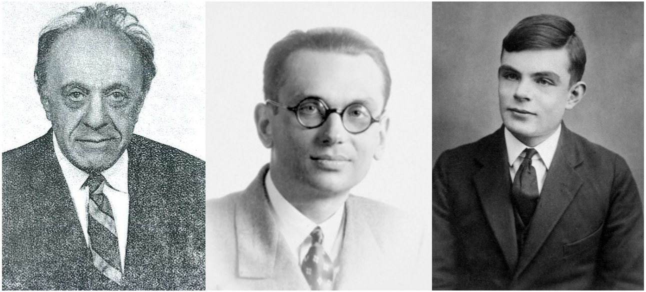


Pre-History: The 1930s

1. Exercise 2: give an $O(n \cdot n!)$ algorithm to the TSP problem.

Pre-History: The 1930s

1. Exercise 2: give an $O(n \cdot n!)$ algorithm to the TSP problem.
2. "Can we do better?" - Menger, 1930
3. Who's Menger?



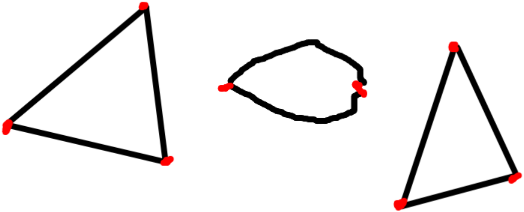
Above: Menger, Gödel, Turing

More Advances

1. Julia Robinson, RAND Corporation.
2. Coined the term "Traveling Salesman Problem".
3. Outside of TSP: MRDP Theorem.

More Advances

1. Julia Robinson, RAND Corporation.
2. Was also think about the Assignment Problem / Cycle Cover.
3. Instead of finding one cycle, find a *collection* of cycles that, together, visit each vertex at least once.
4. Polynomial Algorithm first by von Neumann and Munkres.



Example of a cycle cover

So, can we do better?

1. Question: is there an exact algorithm that runs in time $o(n!)$?
2. Poll: yes/no/we don't know.

[1] Idea: pick some source s. Let $f(S, v)$ be the shortest path from s to v, visiting vertices $S \subset T$. Use recursion and avoid waste (like we did w/ Fibonacci).

So, can we do better?

1. Question: is there an exact algorithm that runs in time $o(n!)$?
2. Poll: yes/no/we don't know.
3. Answer: yes! Bellman-Held-Karp 1962: $O(n^2 2^n)$ time.
4. Use Dynamic Programming, which you will learn in Lecture 10. [1]

[1] Idea: pick some source s. Let $f(S, v)$ be the shortest path from s to v, visiting vertices $S \subset T$. Use recursion and avoid waste (like we did w/ Fibonacci).

Can we do even better?

1. Question: is there an exact algorithm that runs in time $O(1.999^n)$?
2. Poll: yes/no/we don't know.

Can we do even better?

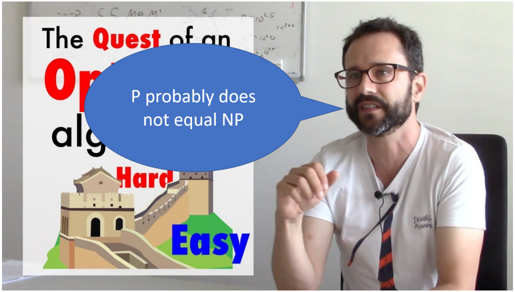
1. Question: is there an exact algorithm that runs in time $O(1.999^n)$?
2. Poll: yes/no/we don't know.
3. Answer: we don't know!
4. STOC 2020: assuming quadratic matrix multiplication (!), on bipartite TSP problems (!), we can do 1.9999^n.

10:30 Bipartite TSP in $O(1.9999^n)$ Time, Assuming Quadratic Time Matrix Multiplication (video)
Jesper Nederlof (Utrecht University)

Why so hard?

1. It's an NP hard problem.

Why so hard?

1. It's an NP hard problem.
2. 

3. So, what can we do?

Dealing with NP-Hardness

Question 3: what can we do when we can't find an efficient exact algorithm for a problem?

Dealing with NP-Hardness

Question 3: what can we do when we can't find an efficient exact algorithm for a problem?

0. Cry

Dealing with NP-Hardness

Question 3: what can we do when we can't find an efficient exact algorithm for a problem?

0. Cry
1. Use heuristics or practical algorithms: is this problem easy in practice?
2. Give an approximate answer.
3. Special cases?

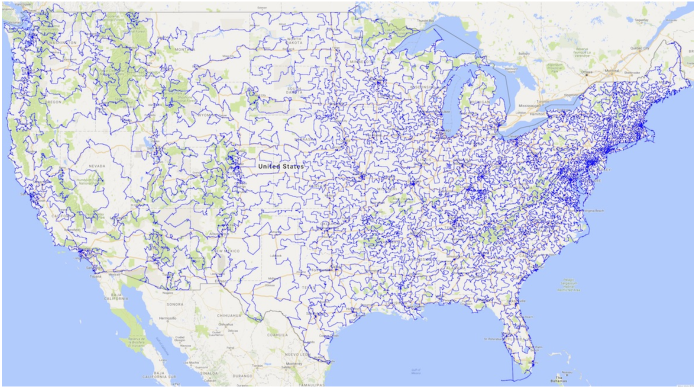
We will focus on ideas #1 and #2.

Heuristics

1. Many heuristics: Cutting plane, nearest neighbor, etc.
2. Many work well in practice.
3. Also heuristics for lower bounds.
4. Many success with real graphs. See UWaterloo's Exposition.

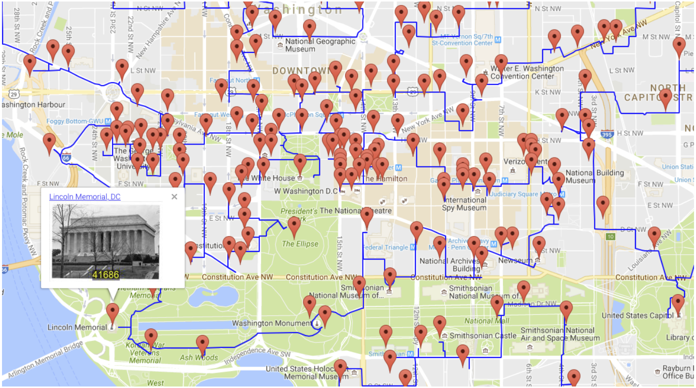
Illustrations

How would we visit every US Historical site?



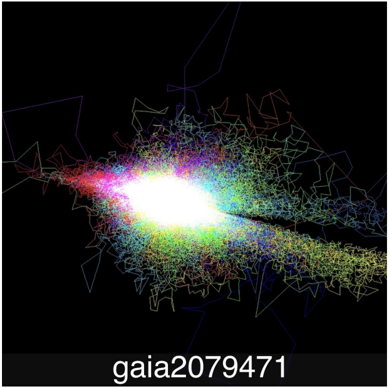
Illustrations

How would we visit every US Historical site?



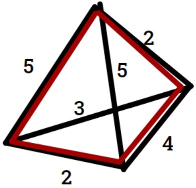
Illustrations

How would we visit every star in the milky way?
$2 \cdot 10^6$ stars here, correct within a factor of $7 \cdot 10^{-6}$.



Idea 2: Approximation

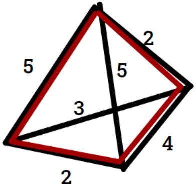
1. First, measure how good a solution P is. In this case: set $c(P)$ to be the length of the cycle.
2. α-approximation if $c(A(G)) \leq \alpha \cdot c(OPT(G))$ for *all* G.



How does the red cycle compare with OPT?

Idea 2: Approximation

1. First, measure how good a solution P is. In this case: set $c(P)$ to be the length of the cycle.
2. α-approximation if $c(A(G)) \leq \alpha \cdot c(OPT(G))$ for *all* G.

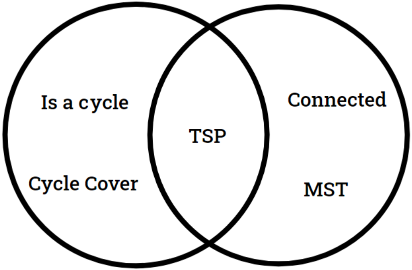


How does the red cycle compare with OPT?

3. Gold standard: constant α
4. Platinum standard: FPAS (see C224/6.854!)

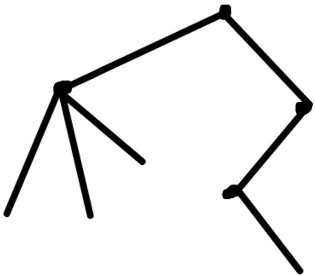
How to approximate TSP

1. We can think of TSP as having 2 condition: the answer needs to have only cycles, and that the answer is connected.
2. We know how to solve both of these problems, esp. MST!



2-Approximation

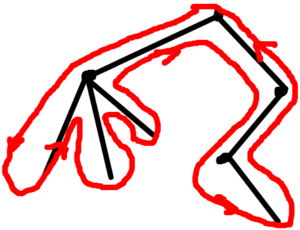
1. Let's start with a minimum spanning tree.
2. Let T be the MST, and let P be the optimal solution to TSP. Then, $c(T) \leq c(P)$.
3. Why?



MST of some graph G

2-Approximation

1. Let's start with a minimum spanning tree.
2. Let T be the MST, and let P be the optimal solution to TSP. Then, $c(T) \leq c(P)$.
3. Why?
4. Next, let's turn the tree into a solution.



MST of some graph G

Proving Approximations

1. Notice that $c(A(G)) \leq 2c(T)$.
2. Also, $c(T) \leq c(OPT(G))$.
3. So, $c(A(G)) \leq 2 \cdot c(OPT(G))$.

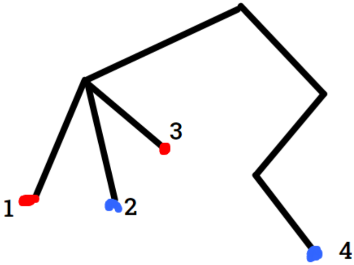
1.5-approximation

Can we do better? Yes!

Theorem (Euler)

If every vertex in a graph has even degree, then there is a path that visits every edge once.

Goal: add as few edges as possible to our MST to make vertices have even degree.



MST of some graph G

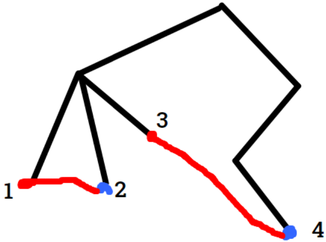
1.5-approximation

Can we do better? Yes!

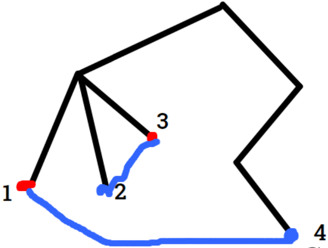
Theorem (Euler)

If every vertex in a graph has even degree, then there is a path that visits every edge once.

Goal: add as few edges as possible to our MST to make vertices have even degree.



MST of some graph G



MST of some graph G

Can we do better?

1. Is there a polynomial-time approximation with ratio $< 3/2$?
2. Poll: yes/no/we don't know

Can we do better?

1. Is there a polynomial-time approximation with ratio $< 3/2$?
2. Poll: yes/no/we don't know
3. Answer: *Yes!*

A (Slightly) Improved Approximation Algorithm for Metric TSP

Anna R. Karlin, Nathan Klein, Shayan Oveis Gharan

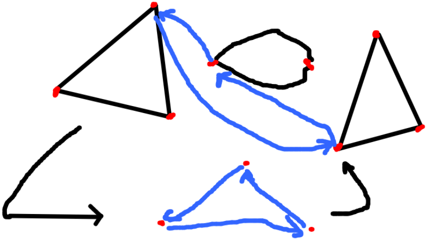
For some $\epsilon > 10^{-36}$ we give a $3/2 - \epsilon$ approximation algorithm for metric TSP.

Directed Graph

Question: why does our 2-approximation algorithm not work?

Approximations with Cycle Cover

1. Start with a cycle cover, and contract each cycle into 1 point.
2. Notice that the number of vertices have halved.
3. Build a cycle cover on the new graph, and repeat the process.



Cycle Cover, cont.

1. Analysis: how many iterations of cycle cover do we have to run?

Cycle Cover, cont.

1. Analysis: how many iterations of cycle cover do we have to run?
2. The cost of each cycle cover is at most $c(OPT(G))$, so the total cost is at most $\log n \cdot c(OPT(G))$.

Can we do better?

1. Is there a polynomial-time approximation with ratio better than $O(\log n)$?
2. Poll: yes/no/we don't know

Can we do better?

1. Is there a polynomial-time approximation with ratio better than $O(\log n)$?
2. Poll: yes/no/we don't know
3. Answer: *Yes!*

Recent progress in Directed TSP

1. Blaser et. al 2002: $0.99 \log n$,
2. Asadpour et. al 2017: $\log n / \log \log n$,

Recent progress in Directed TSP

1. Blaser et. al 2002: $0.99 \log n$,
2. Asadpour et. al 2017: $\log n / \log\log n$,
3. Svensson et. al 2018: 504,
4. Traub Vygen 2020: $22 + \epsilon$.
   Idea: Use Linear Programming Duality (Lecture 20) to think of costs as toll gates, combine this new perspective with cycle cover and network flows (Lecture 19).

Summary

1. TSP's past: important problem with many prongs of attacks, by many great minds.
2. TSP's present: breakthroughs in approximation algorithms.

Takeaways:

1. The driving question of algorithms research: "Can we do better?"
2. Many general principles: MST, DP, Linear Programming
3. Corollary: you are not far from the cutting edge :)