Project 4: MSTs and TSP

Eric A. Autry

Info

This set of slides includes the written description of the Traveling Salesman Project and the MST Approximation approach from the project.

Towards the end, there are a number of slides with images to help demonstrate how the approximation works, to help visualize the idea.

Please read these slides over before starting the project!

Traveling Salesman Problem

Suppose you are a salesperson who will need to meet with a number of clients living in various cities. To give you as much time with the clients as possible, you would like to minimize the time you spend traveling between the different cities.

Specifically: we can create a *complete* graph where the cities are the vertices and the edge weights are given as the distance (on the sphere) between all the pairs of cities. We want to find the cheapest **tour** of the graph that visits every vertex only once outside of returning to a given starting city.

This is a variation of the classic Traveling Salesman Problem (TSP), where you would like to minimize the *cost* of the tour. The variation we are solving in this project defines the 'cost' between two cities to be the great circle distance between them.

Traveling Salesman Problem

The Traveling Salesman Problem is an example of an NP-Complete problem. These problems are very hard to solve. In fact, there are no known polynomial time algorithms that can solve any of them. All known algorithms perform in exponential time, or worse. As an example, the naive approach (called **brute force**) would be to simply check every possible tour looking for a minimum tour. If there are n cities, this will take $(n-1)!$ time (number of permutations of the non-start cities).

While we do not know of an efficient algorithm to optimally solve this problem, there is an approach that promises an approximation within a factor of 2 of optimal, i.e., if the optimal solution has cost $|\sigma|$, then this approximation has cost $|A| < 2|\sigma|$.

MST and TSP (Project 4)

We are looking for a cheap TSP tour that visits all the cities.

- Note that a Minimum Spanning Tree (MST) *reaches* all the cities at a minimum cost M.
- Important observation: the weight of the MST is less than the weight of all tours.
  - Consider a tour that visits all cities only once.
  - When you remove an edge from this tour, you have a spanning tree.
  - But the *minimum* spanning tree has less weight than all spanning trees.
- But, the TSP solution is a tour.
  - Say the optimal TSP tour has weight σ.
  - Then we know that $M < \sigma$.

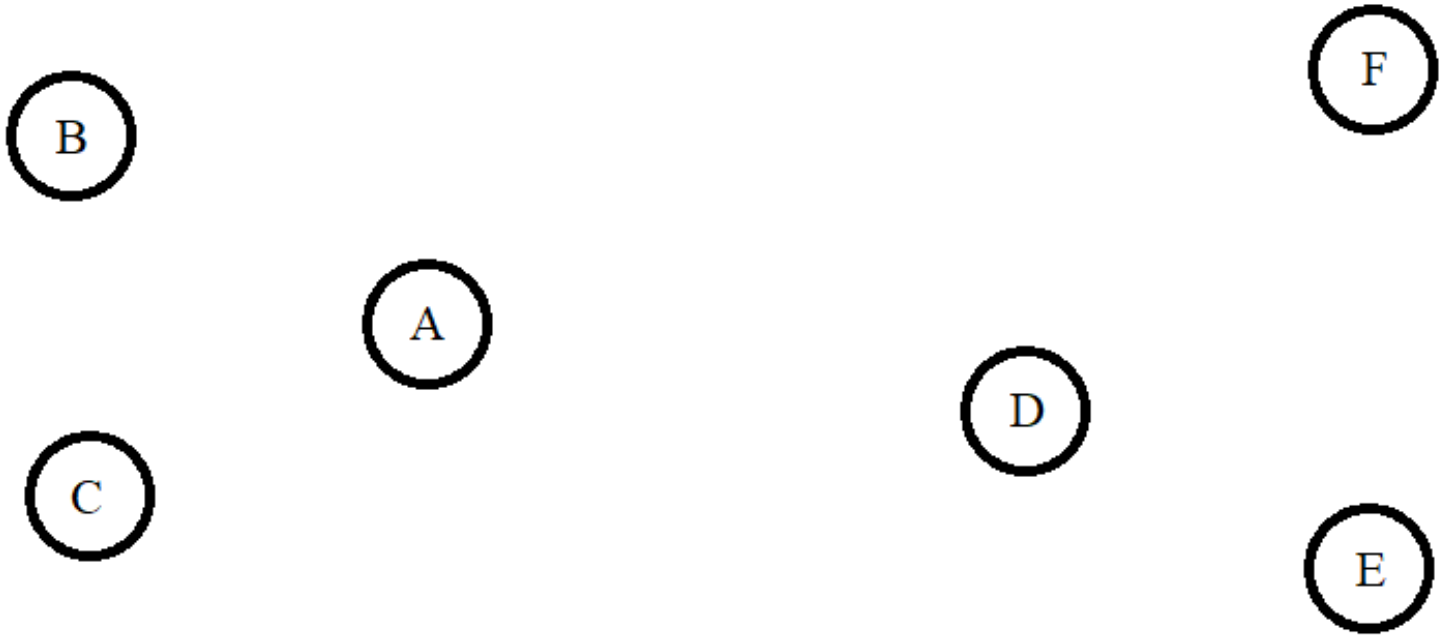
MST and TSP (Project 5)

What if we use Depth-First Search (DFS) to explore the MST?

- For this approach, we will force DFS to revisit cities as it passes them. (i.e., we force our salesman to stay on the edges of the MST.)
  - We will change this later.
- We would have a tour of the cities, except that we might visit some cities twice.
- The cost for this 'tour' would be $2M$.
- But remember that $M < \sigma$, so the cost for this 'tour' would be $2M < 2\sigma$.
- We will have visited the cities for a cost that is within a factor of 2 of optimal!

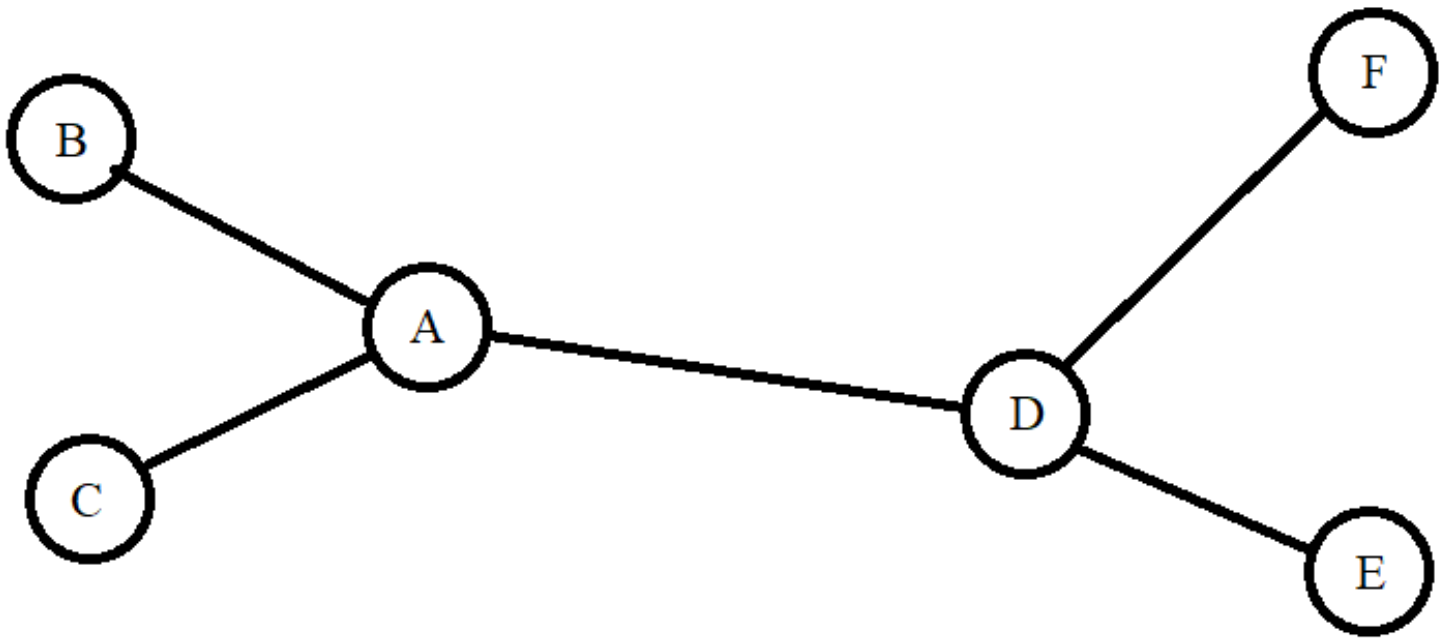
MST and TSP (Project 5)

Consider the following cities. Remember that the TSP problem gives us a complete graph, so while they are not drawn in, there are edges between every pair of cities.



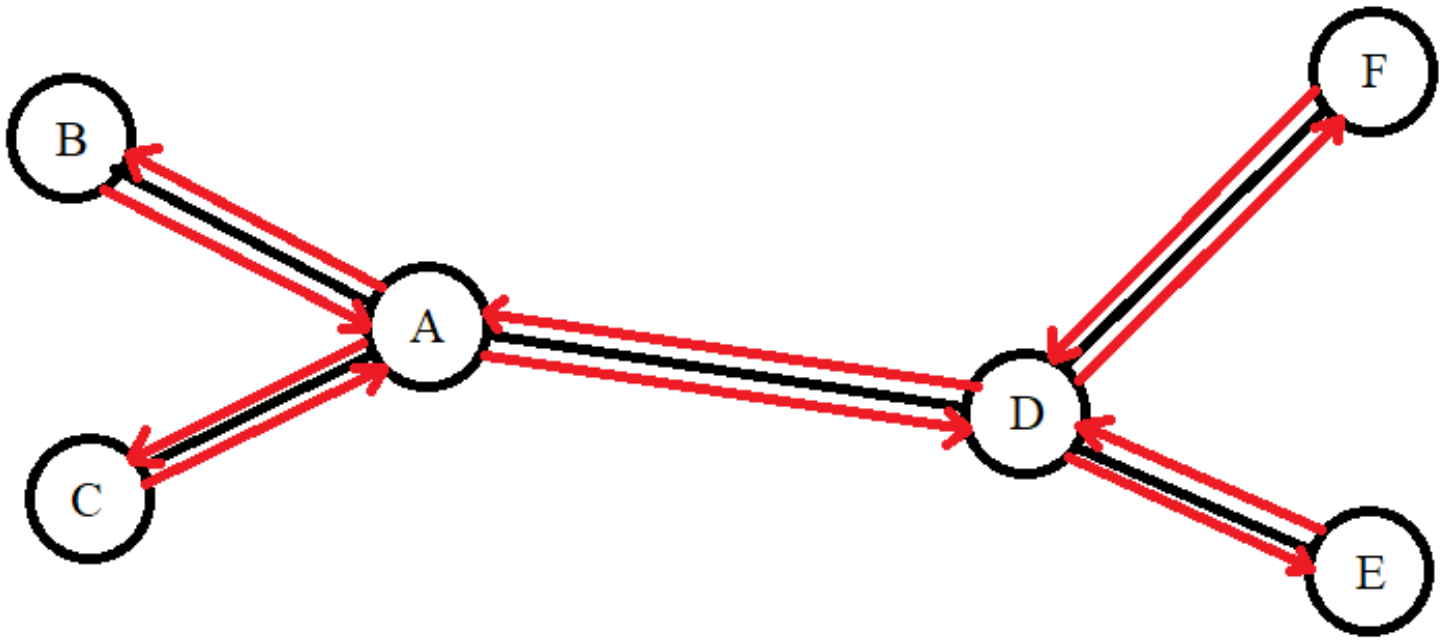
MST and TSP (Project 5)

We now draw the MST in black, which has a weight M.



MST and TSP (Project 5)

Starting at city A, we visit every city using DFS, remaining only on the MST edges. Note that this 'tour' uses each edge twice, for a cost of $2M$.



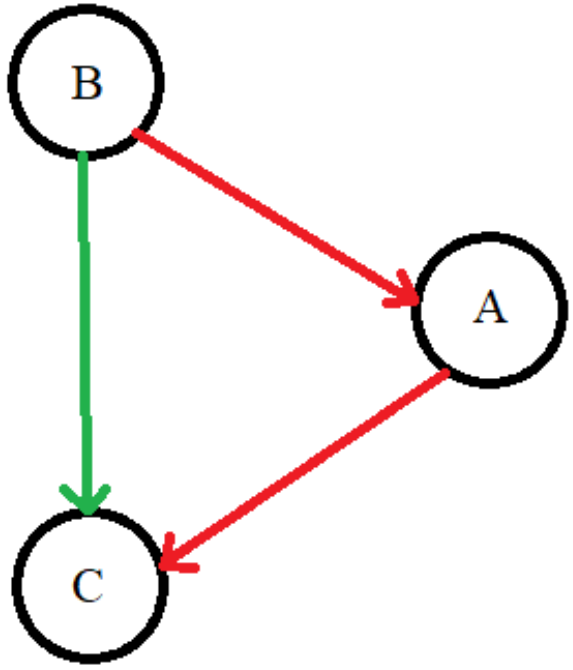
MST and TSP (Project 5)

Problem: our DFS-MST approach is not a proper tour.

- We will now need an important restriction on the TSP problem.
- The distances must obey the triangle inequality:
  - The sum of the lengths of two sides of a triangle is always greater than or equal to the length of the third side.
- Real life distances between cities obey this property!

MST and TSP (Project 5)

The triangle inequality will help us because we can now make a significant observation: it is always better to simply go directly from B to C instead of revisiting city A in between.



MST and TSP (Project 5)

We can now modify our DFS-MST approach:

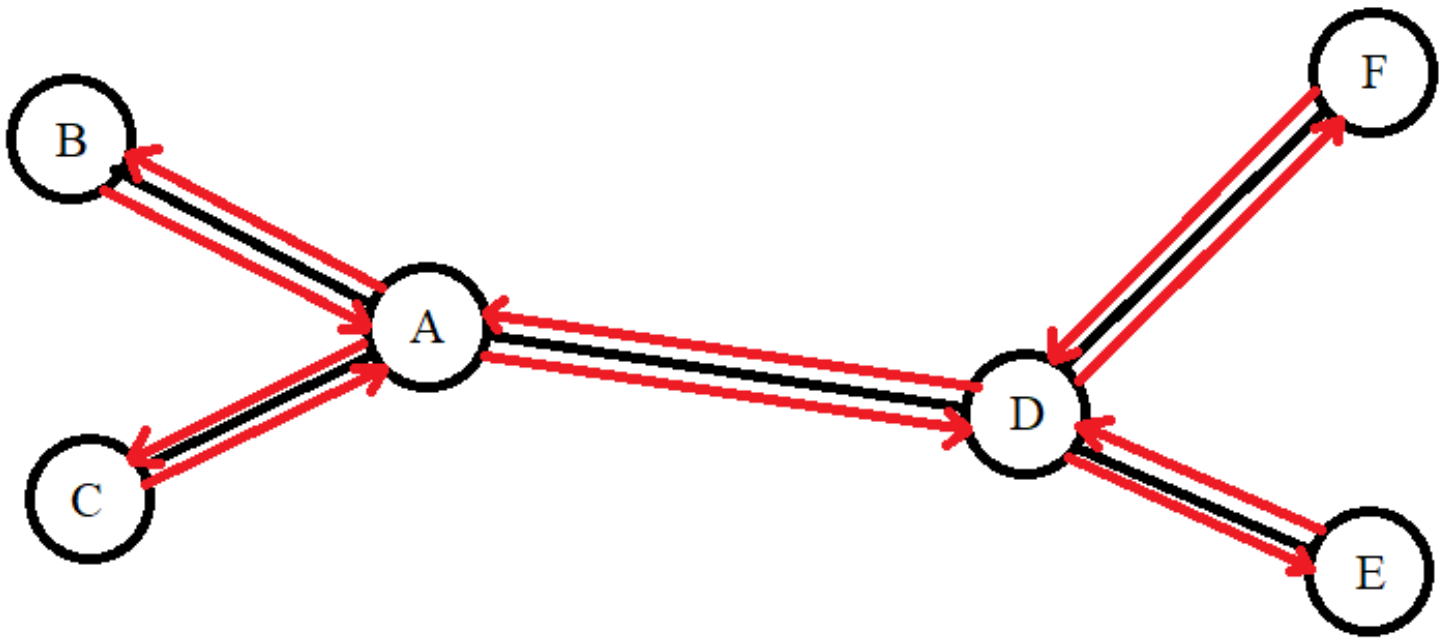
- We still use DFS to determine the order in which we will visit the cities.
- But now we will not allow it to return to cities as it explores.
- As an example: say the DFS goes through cities $A \to B \to A \to C$.
  - i.e., city B was a leaf of the MST and we had to return to A before moving on to C.
  - Recall: the TSP graph is a complete graph, so we can go directly from B to C.
  - So we will say that the tour visits $A \to B \to C$ directly.
  - Due to the triangle inequality, the distance of the path $B \to A \to C$ (two sides of the triangle) is greater than or equal to the distance of the path $B \to C$ (the third side).

So our approximation is less than $2M < 2\sigma$, and we will have an answer within a factor of 2 of the optimal tour!

MST and TSP (Project 5)

Originally, our MST 'tour' gave us (where the capital letters are used to denote when a new city is visited)

$$A \to B \to a \to C \to a \to D \to E \to d \to F \to d \to A$$



MST and TSP (Project 5)

Removing the cities that we revisited (keeping only the capital letters from before, but in the order they were encountered using DFS), we obtain the approximation tour:

$$A \to B \to C \to D \to E \to F \to A$$