Yet Another 2-opt

Simple And Fast Approximation For
The Traveling Salesperson Problem

Jacob Thomsen
jakethom02@gmail.com

John Tappen
jtappen@gmail.com

Jonah Simmons
jonahksimmons@gmail.com

Professor Tony Martinez
Brigham Young University
April 18, 2023

1 Abstract

The Traveling Salesperson Problem (TSP) is arguably the most well know NP-Hard problem in the world. From its formalizing in the 19th century, it seems all possible solutions have been exhausted: from optimal exponential solutions to fast heuristic approximations. We will be focusing on an approximation that uses a greedy approach to find a fast solution. Then, we will improve upon it using an optimization known as *k-opt*.

2 Greedy

2.1 Disadvantages

Greedy algorithms solves problems with the "best now" mentality. It looks at all current options, assigns some quantifier to each, and chooses the best. For example, a greedy chess solver would protect its pieces or capture rather than sacrificing a piece for a checkmate in three moves. As it does not take into account future decisions, it usually does not return the optimal for more complex problems, TSP being one of those.

2.2 Advantages

Although it usually does not return an optimal, greedy approaches return good enough solutions for most problems. From the tests we ran, the greedy solutions were hard to beat by much.

In addition to returning a good value, it does it very fast. The main issue with guaranteed optimal approaches is that it must take into account all other possibilities to ensure the best solution; this takes an unpractical amount of time.

2.3 Time Complexity

Algorithm 1 Greedy algorithm

```
1: procedure GREEDY → PATH
2:     let current = starting city
3:     while not visited all cities do
4:         let best = closest neighbor to current that has not been visited
5:         path.add(best)
6:         current = best
7:     end while
8: end procedure
```

Given the pseudo-code, we can see the time complexity is $O(V \times E)$, where V is the number of cities and E is the number of out edges. The outer loop goes through all nodes, $O(V)$, and looks at all neighbors, $O(E)$.

2.4 Purpose

The purpose of including the greedy algorithm is to offer a benchmark and comparison to the other algorithms. It produces fine results, but can be improved with a little bit of work.