Analysis of TSP algoritmhs

Maciej Woczyk

December 7, 2016

Abstract

TSP algorithms by Maciej Woczyk.
In this document we'll see how Brute Force and Nearest Neighor solutions behave with certain set of cities

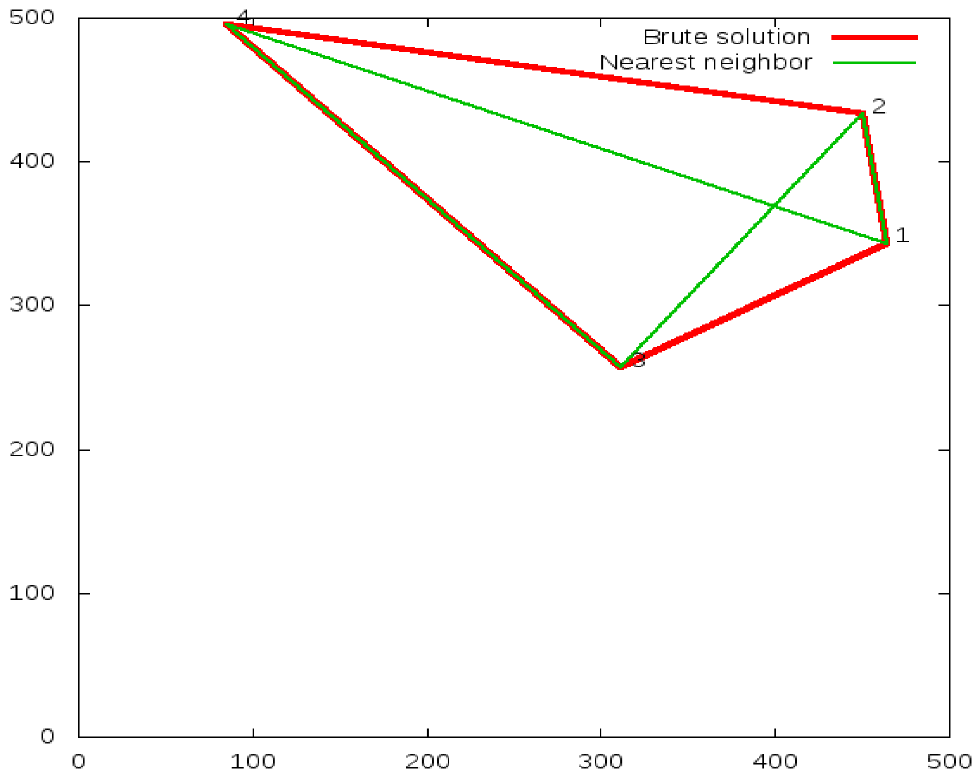

Figure 1: As we can see in this example brute force is much more efficent. Nearest Neighbor starts with town #1, goes to the closest #2, then #3 and #4, making unnecesarily long comeback to #1. Brute Solution often goes on the "edge" of our town map, since going through the center would mean coming a long way back.

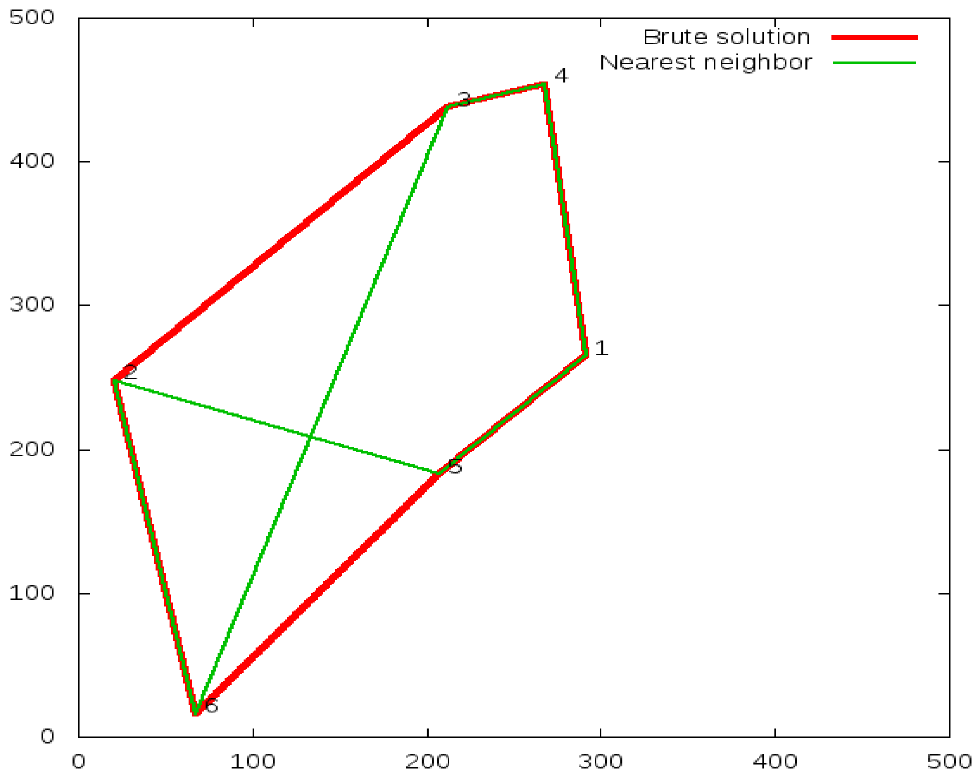

Figure 2: The more the points, the worse Nearest Neighbor behaves. As we can see, routes of both algorithsm are the same at the beginning #3 ⇒ #4 ⇒ #1 ⇒ #5 but then Nearest Neighbor takes the worse path to #2 instead od #6. That means it has to go unnecesarily long way to get back to #2.

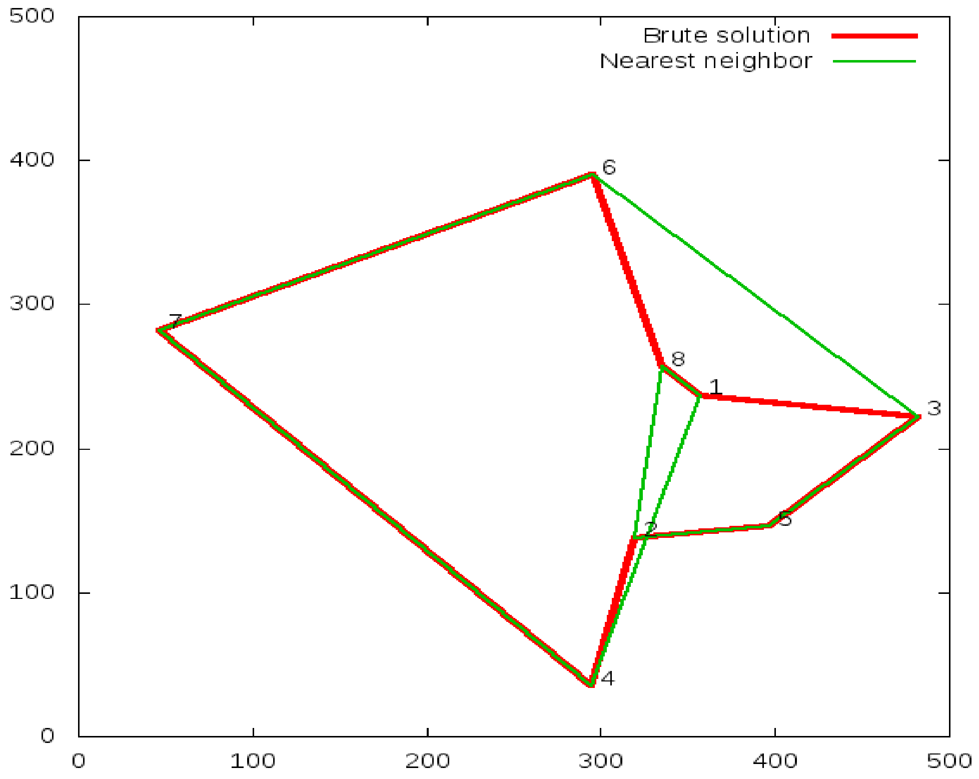

Figure 3: With 8 points routes become more complicated. Brute solution keeps going on the edge of our map, but nearest neighbor takes some wrong turns (f.e. #6 ⇒ #3)

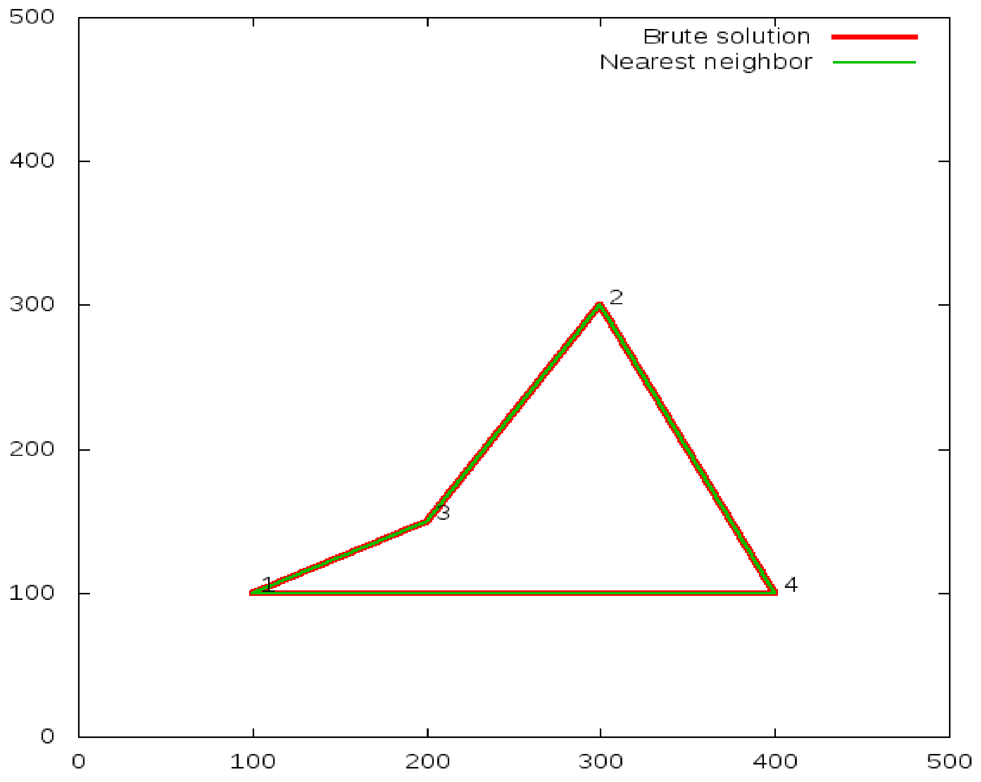

Figure 4: Simple example, showing that with simple town placement Nearest Neighbor manages to keep up with the best solution produced by brute forcing.

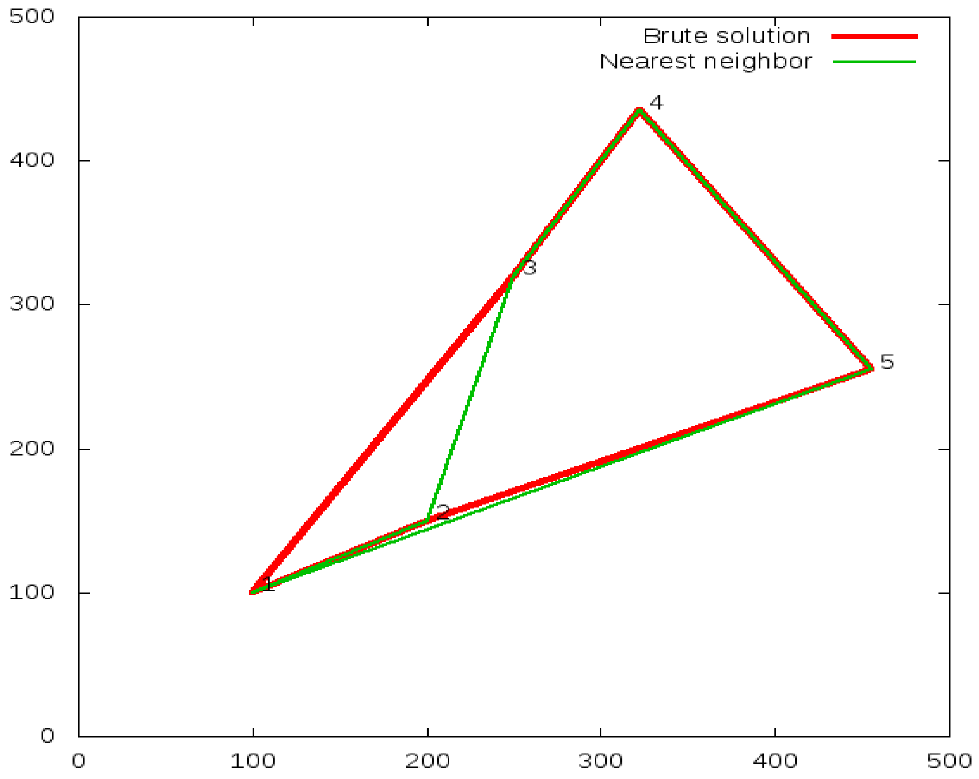

Figure 5: But if the distances between towns are a little bit more tricky, Nearest Neighbor makes some mistakes. The main advantage of this algorithm is that it is relatively quick - with $O(n^2)$ complexity. In comparison, brute force solution is $O(n!)$ complex. Since TSP is NP-hard, there is probably no algorithm which gives precise, exact solution faster than $O(2^n)$).