Description: In optimization, **2-opt** is a simple local search algorithm for solving the traveling salesman problem. The main idea behind it is to take a route that crosses over itself and reorder it so that it does not.

- A B - - A - B -

× ==>

- C D - - C - D -

Explaination:

Input: route(length = n), a, b

1. Take route[0] to route[a] to add them to new\_route
2. Take route[a+1] to route[b], reverse them and add to new\_route
3. Take the rest of route: route[b+1] to route[n-1] and add them to new\_route.

For example: route = [A,B,C,D,E,F,G], a = 2, b = 4

We got: new\_route = [A,B,C,E,D,F,G]

Then we compare the new\_route with route to check if there’s improvement (usually is faster, shorter path,…).

A complete 2-opt local search will compare every possible valid combination of the swapping mechanism.

We keep performing 2- opt swap until there’s no room for improvement, while checking the violation of time due.

Conclusion: A simple yet efficient solution for this problem