Travelling Salesman Problems

Alessandro Hill

rev. 1.0(AH) – 2024

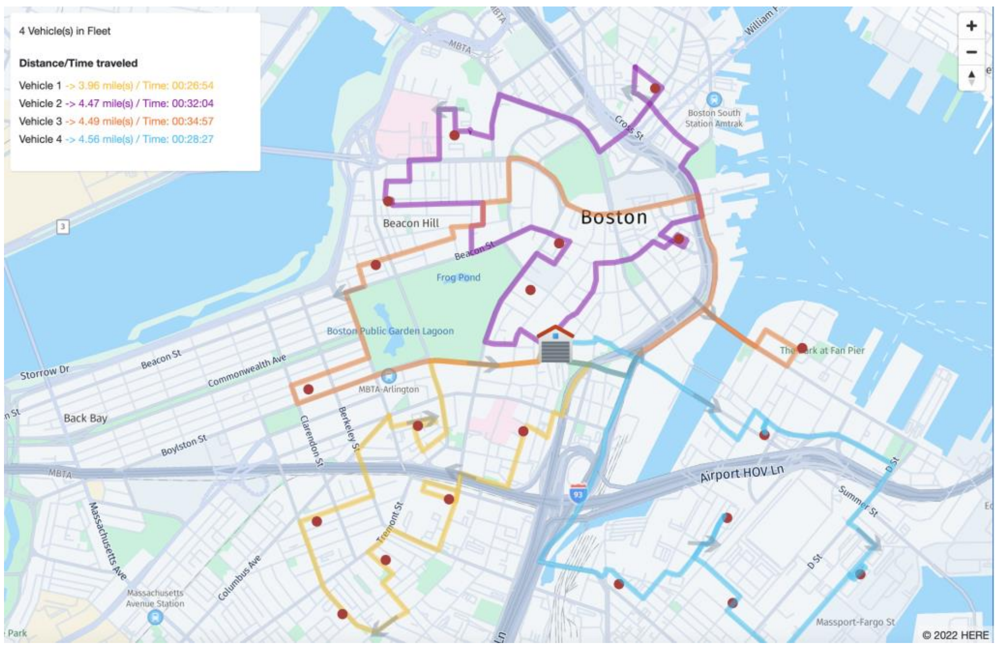
The Traveling Salesman Problem

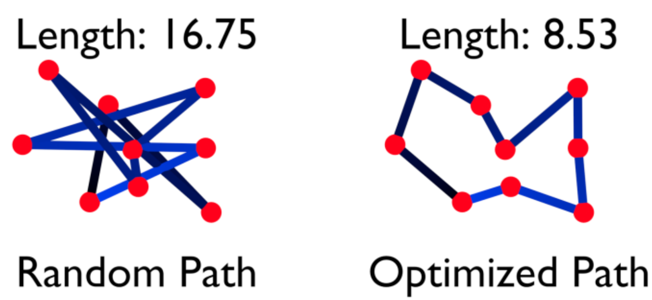
Problema del commesso viaggiatore

We are given a directed base graph with node set N (locations) and arc set A (travel segments). Each arc (i,j) has a non-negative weight $c_{i,j}$ (distance/duration).

> The Travelling Salesman Problem (TSP) asks for a tour (directed cycle, or circuit) of minimum total length that includes every node exactly once.*

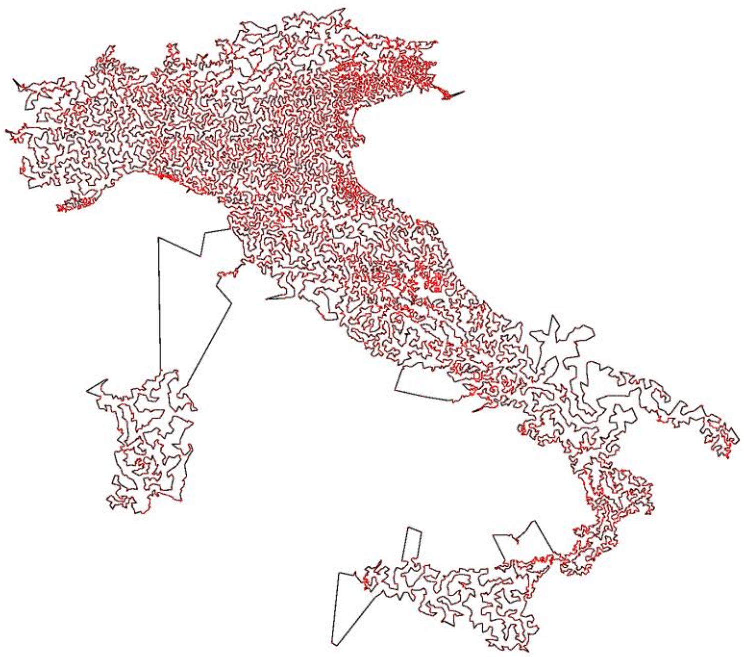
*NP-hard (extremely difficult to optimize)





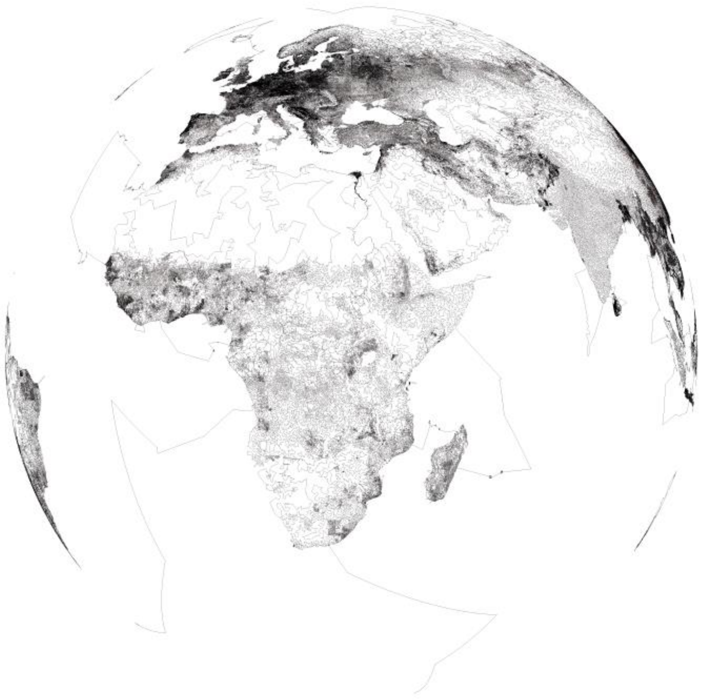
The Traveling Salesman Problem

16,862 Cities in Italy



557,315 km

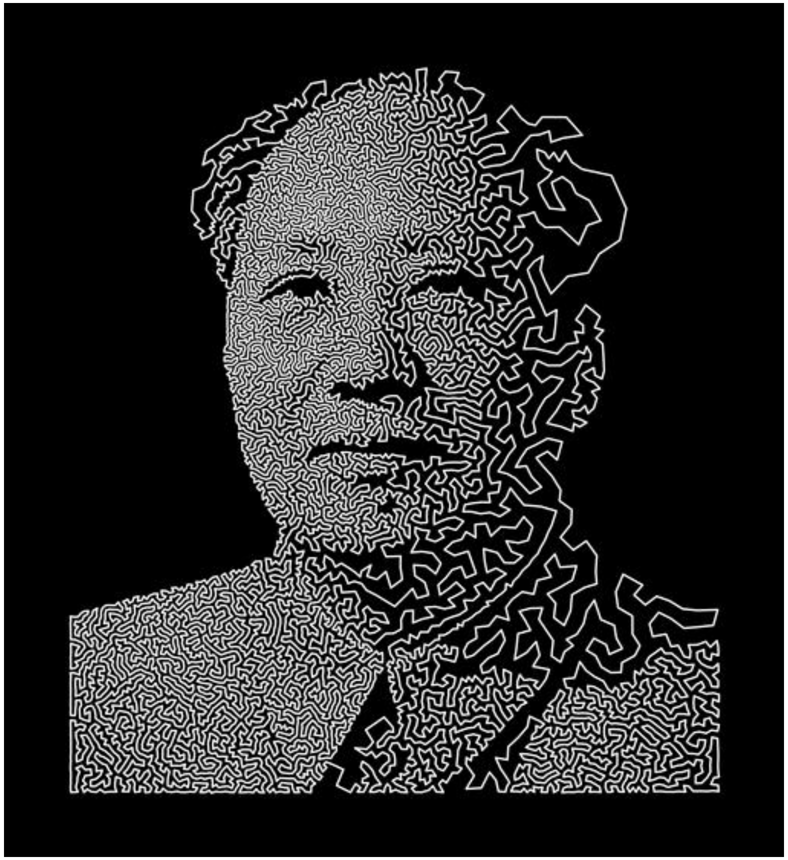
1,904,711 Cities Worldwide

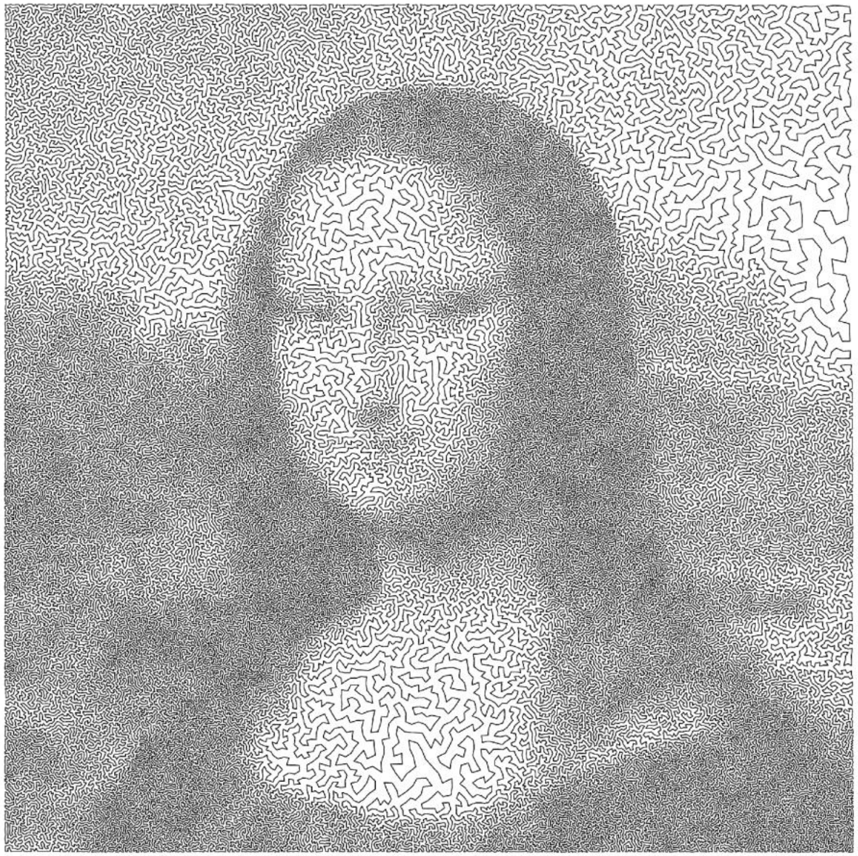


7,516,353,779 km

Also see http://www.math.uwaterloo.ca/tsp/world/index.html

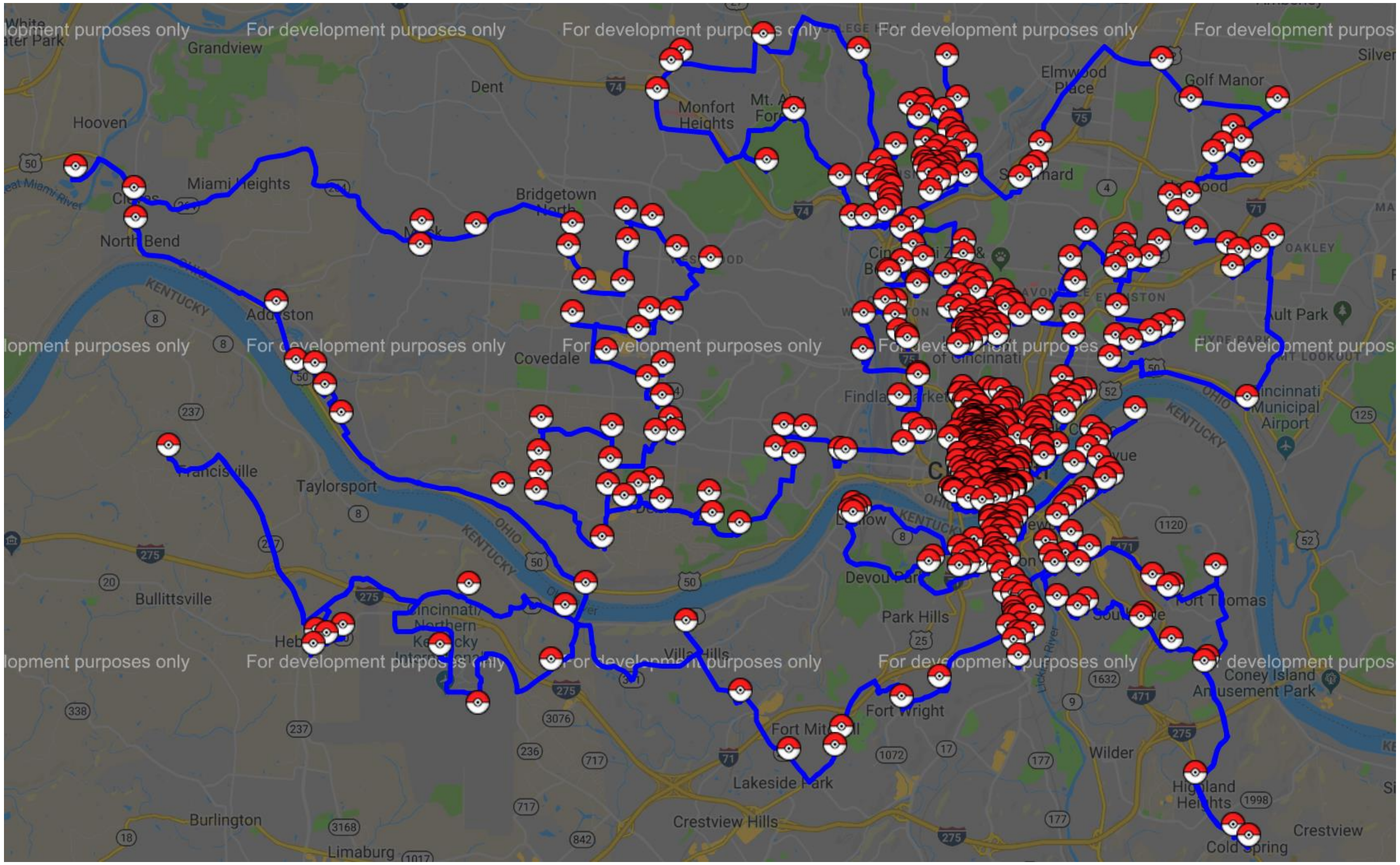
The Traveling Salesman Problem





Also see http://www2.oberlin.edu/math/faculty/bosch/tspart-page.html and https://cs.uwaterloo.ca/~csk/other/tsp/

The Traveling Salesman Problem



Also see http://www.math.uwaterloo.ca/tsp/poke/index.html

The Traveling Salesman Problem

IP Model*

Binary arc variables for each arc that could be part of a tour:

$$x_{i,j} = \begin{cases} 1 & \text{if arc } (i,j) \text{ will be used on the tour,} \\ 0 & \text{otherwise.} \end{cases}$$

Minimize $\sum_{(i,j)\in A} c_{i,j}\, x_{i,j}$

Idea: Each node *v* has exactly one incoming arc and one outgoing arc in a tour.

Force node out-arc:

$x_{v,j_1} + \ldots + x_{v,j_k} = 1$ for out-neighbors $j_1, \ldots, j_k$ of v.

Force node in-arc:

$x_{i_1,v} + \ldots + x_{i_k,v} = 1$ for in-neighbors $i_1, \ldots, i_k$ of v.

*May contain sub-tours…

The Traveling Salesman Problem

IP Model**

Subtour Elimination Constraints (SEC):

$\sum_{(i,j)\in A[S]} x_{i,j} \leq |S| - 1$ for each node subset $S \subset V$

Miller-Tucker-Zemlin Constraints (MTZ):

$\mathrm{u_i - u_j + 1 \leq |N| \cdot (1 - x_{i,j})}$ for $\{\mathrm{i,j}\} \subset \mathrm{N}$ (link variables)

$u_1 = 1$

$u_i > 0$ for i in $\mathrm{N} = \{1,2,\ldots\}$ (new node sequence variables)

****No sub-tours…

The Traveling Salesman Problem

Nearest-Neighbor Heuristic (NN) (can fail!)

1. Initialize empty tour T.

2. Pick arbitrary start node, add to tour T, and use as current node i.

3. Pick the node j that is
   - the closest to the current node i, and
   - not in T.

   Connect i to j, add j to T, and make j the current node.
   (If no node j was found, the heuristic fails.)

4. Go to 3 unless T contains all nodes.

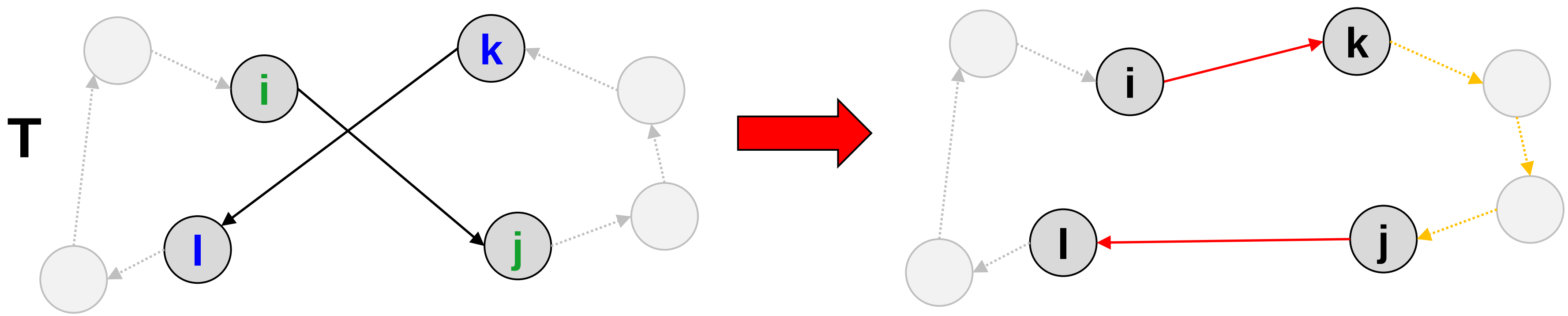
5. Return tour T.

The Traveling Salesman Problem

Local Search: "2-Opt"

For a tour T. For two arcs (i,j) and (k,l), let the first 2-opt move be defined as:
- Remove arcs (i,j) and (k,l) from T.
- Insert arcs (i,k) and (j,l) into T.
- Reverse path from j to k in T.

The second 2-opt move inserts (k,i) and (l,j), and reverses path from l to i.)



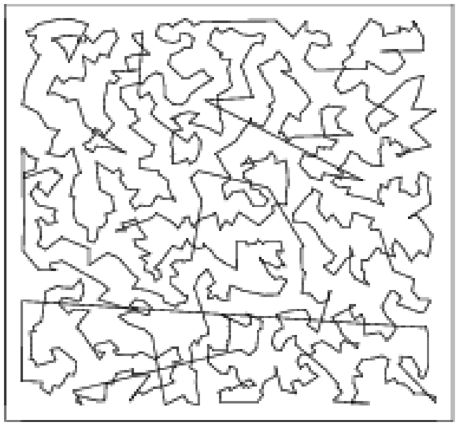
Local Search Heuristic:

INPUT: Feasible tour T.

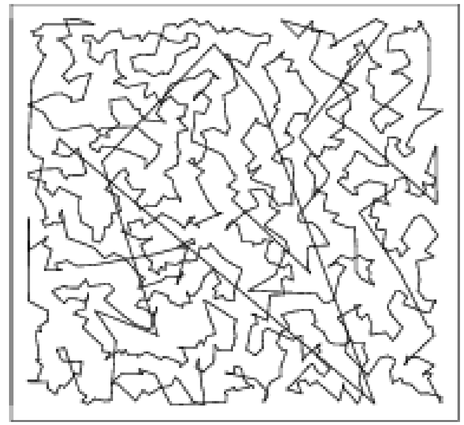
1. Find 2-opt move of highest improvement in T.
2. If improvement > 0 then implement move for T and go to 1.
3. Return tour T.

The Traveling Salesman Problem

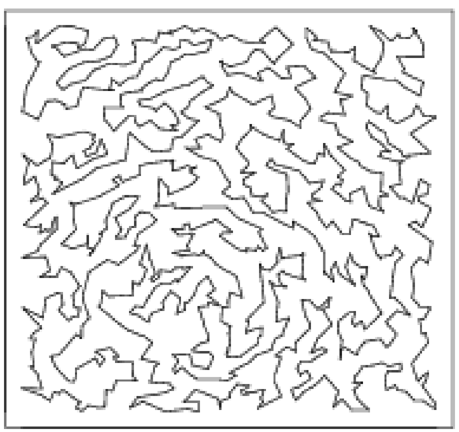
Examples



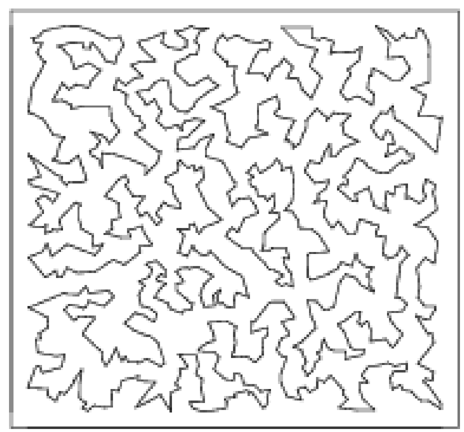
Greedy Tour



Nearest Neighbor Tour



Savings Tour



Optimal Tour

The Traveling Salesman Problem

What is the shortest tour that visits all cities A, …, E?

Travel Distances

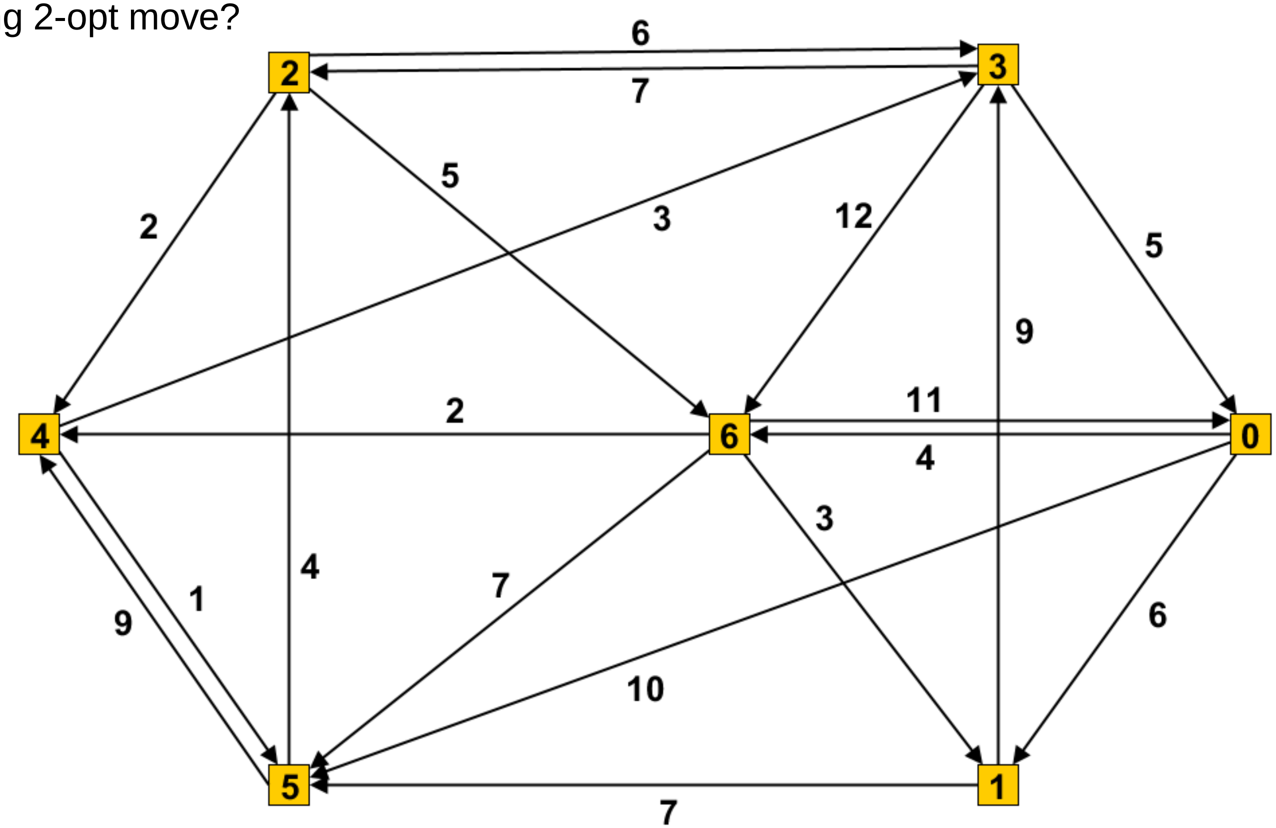
Cities	A	B	C	D	E
A	0	3	4	2	7
B	3	0	4	6	3
C	4	4	0	5	8
D	2	6	5	0	6
E	7	3	8	6	0

The Traveling Salesman Problem

1) Find a TSP tour using the nearest-neighbor heuristic.

Start from a) node 1; b) node 4; c) node 5.

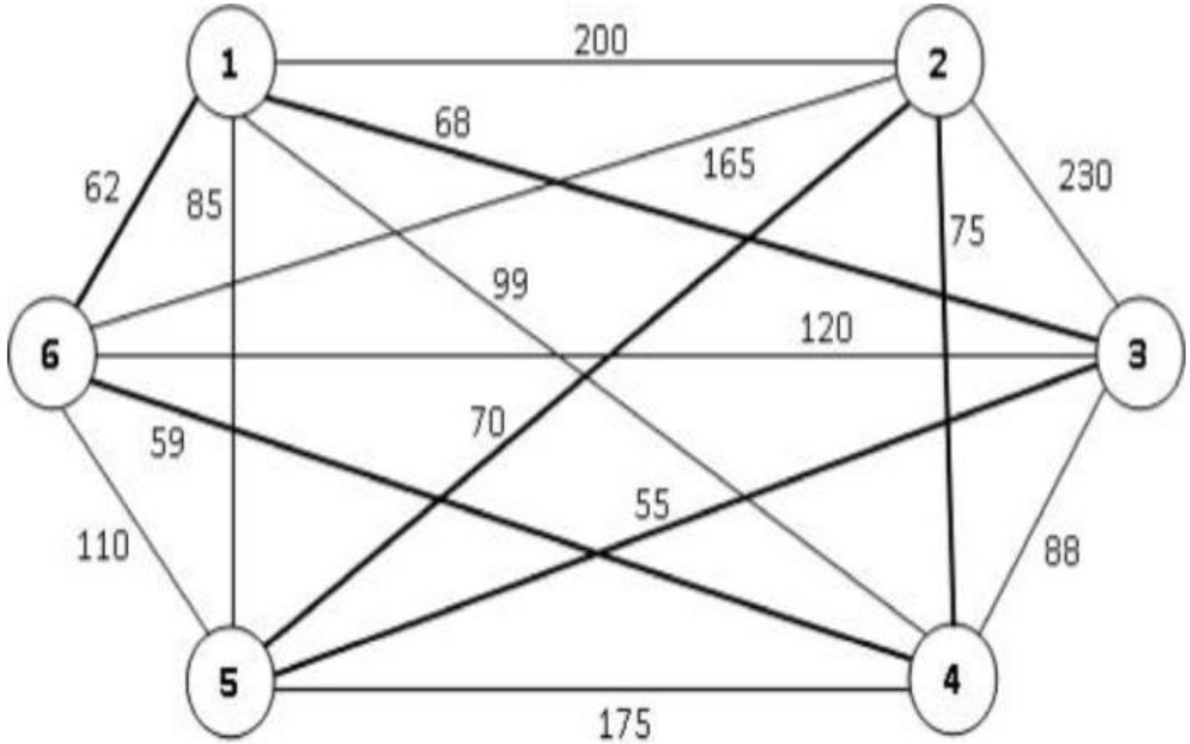
Can we find an improving 2-opt move?



2) Solve the TSP above using IP.

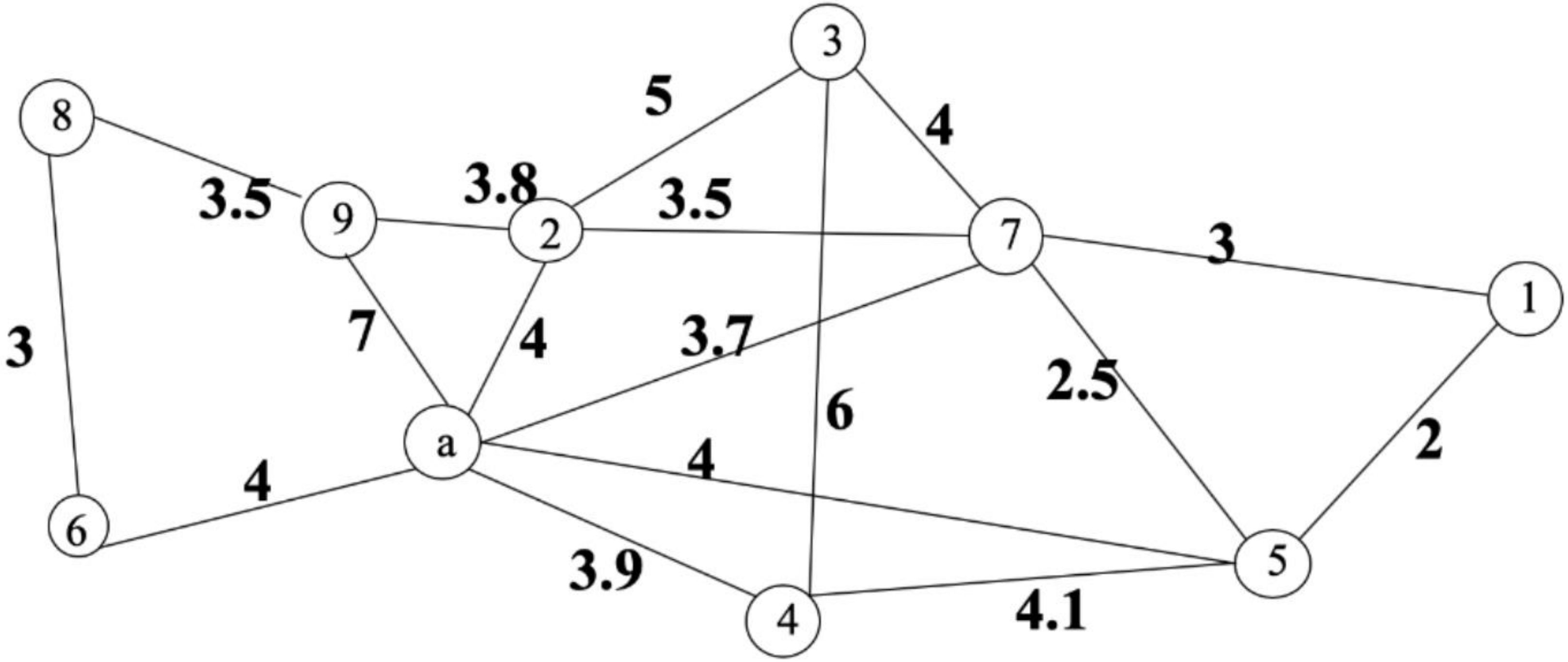
The Traveling Salesman Problem

3) Find a TSP tour using the nearest-neighbor heuristic and compare your results to an IP solution.



The Traveling Salesman Problem

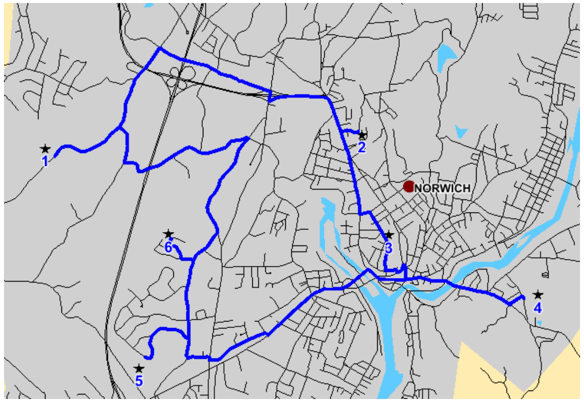
4) Find a TSP tour using the nearest-neighbor heuristic and compare your results to an IP solution.



The Traveling Salesman Problem

5) Group Exercise:

1. Find 5-10 points of interest that we would like to tour in Cesena.
For example, museums, shops, restaurants, bars, cinemas, parks.
They could also all be located within a building.
Other cities are of course also fine.



2. Estimate all the travel distances/durations.
Use an online map system such as Google Maps.
Is there an advanced tool that returns a distance matrix?
You can use walking, cycling, driving to do the tour.
Enter your data in a spreadsheet.

3. Find a short tour using the NN-Heuristic.
Use different starting points if necessary.

4. Build and solve an IP model.
Include either SEC or MTZ inequalities.

5. Visualize your best tour using yEd.
Can you add a map snapshot as background?