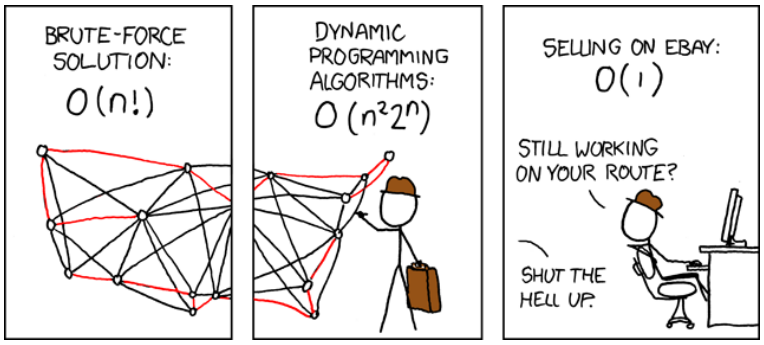
Traveling Salesman Problem

Let $G = (V, E)$ be a complete graph on n vertices. Let $c : E \to \mathbb{R}^+$. The Traveling Salesman Problem (TSP) is to find a closed cycle/circuit C through all vertices of minimum cost.

$$\min \left\{ \sum_{e \in C} c(e) : C \text{ is a circuit through all vertices of } G \right\}$$

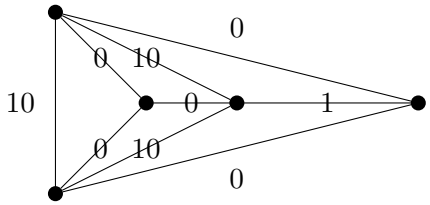
The problem is NP-complete. There is no k-approximation algorithm in general for any k. Otherwise, it would solve Hamiltonicity of a graph G by considering K_n with weights 1 on $E(G)$ and *huge* on the rest.

We will try TSP *only* for points in the plane (triangle inequality satisfied and cost = distance is positive).



xkcd.com/399

1: Solve Travelling Salesman Problem for the following graph.



Solution: Cost is 10.

Heuristics for getting approximate solutions to TSP

- *Nearest neighbor:* Build a path, always including the nearest neighbor. On test data, this gives 1.26 times the optimum.

  2: Show that the nearest neighbor algorithm can do arbitrarily bad without the triangle-inequality.

  Solution: Four vertices are enough. One could lead the shortest path to a trap.

  If the triangle inequality is satisfied, the solution can be $\frac{1}{3}(\log_2(n+1)+\frac{4}{9})$ times the optimum. Guaranteed to be at most $\frac{1}{2}\log_2(n)$-approximation.

- Insertion algorithms: Start with an edge and keep adding vertices one by one. There are different rules possible.

  - *Cheapest insertion:* Add vertex that is cheapest to insert. At most 2 times optimum.
  - *Nearest insertion:* Add vertex that is closest to any of vertices already in the tour. At most 2 times optimum.
  - *Furthest insertion:* Add vertex that is furthest away from any of vertices already in the tour. At most $\log_2 n + 1$ times optimum. Better in experiments than previous.

  Note: No instance is known where any insertion method would do worse than 4 times the optimum.

- *Christofides Heuristics:* Start with a Minimum Spanning Tree T and add a Minimum Cost Perfect Matching M on the set of vertices of odd degree to make a graph H with all degrees even. Follow an Eulerian cycle in H, but skip repeated vertices. At most $\frac{3}{2}$ of optimum.

  3: Show that the upper bound of the algorithm is at most $\frac{3}{2}$ of optimum.
  Hint: First compare $c(T)$ to $c(C)$ by deleting an edge from a circuit C. Next, suppose C_O solves TSP only on the odd vertices from T. Compare $c(M)$ and $c(C_O)$ by splitting C_O into two matchings.

  Solution:

  Since T is a minimum spanning tree, $c(T) \leq c(C-e) \leq c(C)$ for any $e \in E(C)$. Now we show $c(M) \leq \frac{1}{2}c(C)$. First, note that $c(C_O) \leq c(C)$ since contracting along even vertices does not increase the cost. Now we split C_O into two perfect matchings N_1 and N_2 on the set of odd vertices. Then $c(M) \leq \min\{c(N_1), c(N_2)\}$, so $c(M) \leq \frac{1}{2}c(C)$ as desired. Matching can also have cost less than half of the TSP if we walk along the TSP. Splitoff does not increase the cost.

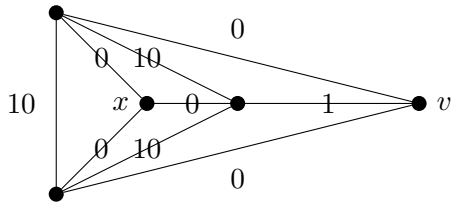
Tour improvements

- *2-optimal switch:* Replace 2 edges by different 2 edges. More generally, k-switch. May take exponential time. Usually 3-switch seems to work great.
- *Lin-Kernighan:* Better way of doing 2-switches.

Lower Bounds

- *Held-Karp:* We start with a 1-*tree bound* and then improve it. *1-tree bound*: Find a vertex v and minimum spanning tree T in $G-v$, then add v to T by using two edges of smallest cost incident to v.

  4: Try the above procedure on the graph below. What is the resulting lower bound and what is an optimal solution to TSP?



  Solution: The lower bound is 0, obtained from edges incident to x as the minimum spanning tree of $G-v$ and the two edges incident to v of cost 0.

  5: What is the trouble with the minimum spanning tree? What would be an 'ideal shape' of a minimum spanning tree to get a nice lower bound?

  Solution: If we have a cycle and remove one vertex, the result is a path. So it would be nice, if the resulting spanning tree is a path. But it is a star.

  6: What happens the cost of *all* three edges incident to the middle vertex x is increased from 0 to 10? What happens to the cost of the minimum spanning tree in $G-v$ and what happens to the cost of TSP?

  Solution: The cost of TSP increases by 2×10 because it contains two edges incident with x and cost of each is increased by 10. That is the cost of TSP is now 30. But the cost of MSP is now also 30. Ha!

7 *Held-Karp Algorithm*: Design an algorithm that will try to obtain a best possible bound using the idea of modifying costs of edges around vertices. How do you modify the costs of edges around vertices based on their degrees? What are the desired degrees in the MST and what means that the degree is different from the desired?

Solution: If a vertex on MST has degree 1, it means edges around it are too heavy and should be lighter a bit. If a vertex has degree more than 2, it means edges around it are too light and should be heavier.

One can implement it by giving a wight y_v to a vertex v and every edge uv has weight as $c(uv) + y_u + y_v$, that is cost of edge + cost of incident vertices.

Question is by how much the cost should change - that is not clear and perhaps experiments would help. Some recommendation is to change y_v as

$$y_v = y_v + t(2 - d_T(v)),$$

where t is a parameter and $d_T(v)$ is a degree of v in the spanning tree. The parameter t is called *step size* and it perhaps could change with time, should be positive. There is some suggestion in the book for points in the plane. Not clear how quickly it converges.

- *Linear-programming:* Can be used to provide a relaxation of integer programing formulation of TSP. (can be modified to match Held-Karp)

  8: How to formulate a lower bound for TSP using linear programming?
  Hint: Make an integer program and relax it.

  Solution: Let $G = (V, E)$ be the input graph. Make a variable x_e for every edge e. Notice we included extra constraint that for every $S \subsetneq V$, there are at least two edges leaving. This is included to prevent a solution that contains disjoint cycles.

$$(P) \begin{cases} \text{minimize} & \sum_{e \in E} c(e) x_e \\ \text{subject to} & \sum_{v \in e} x_e = 2 \text{ for all } v \in V \\ & \sum_{e \in \delta(S)} x_e \geq 2 \text{ for all } \emptyset \neq S \subsetneq V \\ & 0 \leq x_e \leq 1 \text{ for all } e \in E \end{cases}$$