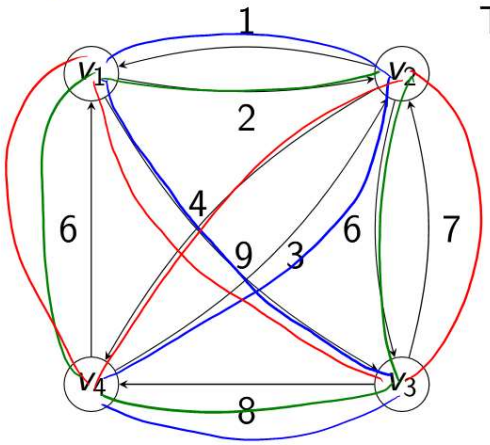
Computer Science 161
Spring 2021 Lecture 18:
Dynamic Programming:
Traveling Salesperson

Traveling Salesperson

WLOG all begin at v_1

How many tours exist? $O(n!)$



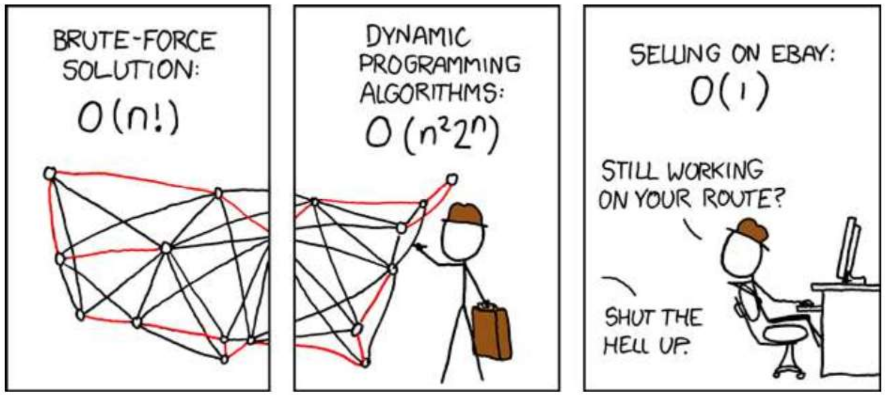
Tours:

- v_1, v_2, v_3, v_4, v_1 cost 22
- v_1, v_3, v_2, v_4, v_1 $9+7+4+6=26$
- v_1, v_3, v_4, v_2, v_1 $9+8+3+1=21$

best we found

$w_{34} = 8$ $w_{32} = 7$ $w_{23} = 6$

3 # XKCD # 399



What's the complexity class of the best linear program cutting-plane techniques? I couldn't find it anywhere. The Garfield guy doesn't have these problems....

4 # Traveling Salesperson

W_{ij} = cost of edge $i \to j$ V_1 ← set A ← V_i

Define OPT(i, A) to be the length of the shortest path from v_i to v_1 passing through each vertex in A exactly once.

A: vertices not yet visited

Suppose $A \neq 0$. Where do we go first?

// $j \leftarrow$ some vertex in A (ask Shindler)

$$OPT(i, A) = \min_{\substack{j \in A \\ \text{w/ edge } i \to j}} \left\{ W_{ij} + OPT(j, A - \{v_j\}) \right\}$$

unless $A = \emptyset$: return W_{i1} (or ∞ if no such edge)

Overall: (1st after start)

$$\min_{\substack{i:\text{ vertices} \\ \text{not } v_1}} \left\{ W_{1i} + OPT(i, V - \{v_1, v_i\}) \right\}$$

5 Traveling Salesperson Time Complexity

$//\text{OPT}(i, A) = \min_j\{w_{ij} + v_j, A - \{v_j\}\}$

1. How many sub-problems are there?

$\mathcal{O}(n \cdot 2^n)$

2. How long does each take to compute?
(Assume that recursive calls are $\mathcal{O}(1)$ lookups)

$O(n)$

$\mathcal{O}(n^2 \cdot 2^n)$

6 Traveling Salesperson Iterative

$//\text{OPT}(i, A) = \min_j\{w_{ij} + \text{OPT}(v_j, A - \{v_j\})\}$

Considerations:

1. How to store incremental solutions?

Table: n vertices x 2^{n-1} subsets

2. What order to fill the table?

for each subset in increasing size

3. How to find the optimal cycle?

for each vertex (not in subset) fill in

bitset interp. as #

std::vector<unsigned, unsigned> table

↑ vertex