Mon, March 27, 2023
Lecture #27
MSSC 6000

Announcements

- * Normal Office Hours today 1pm - 2pm in CU 307
- * OH Wed are moved to 4:30pm - 5:30pm (Teams)
- * HW 4 due Mon, Apr 3.
- * Fri, Apr. 7 - no class
  Mon, Apr. 10 - no lecture (home work day)
  no OH

Topic 11 - Hill Climbing (continued)

Inspired by Gradient Ascent:
MH #2: Steepest Ascent Hill Climbing
(for discrete only)

```
x = random element of S
while True:
    N = nbhd(x)
    s = element of N with the highest
                                score
    if score(s) > score(x):
        x = s
    else:
        # we're at the top of a hill
        quit
```

If continuous, N is probably an infinite set, so we can't compute the score of everything in N.

Pros	Cons
* Guaranteed to find a local optimum	* Unlikely to find global opt. unless lucky and/or search space is nice

* **very slow** because the neighborhoods can be big and we are forced to score every thing in the nbhd.

What's the slow part?

Only doing two things:

(1) generating the nbhd

(2) scoring the nbhd

TSP – scoring a tour of 300 cities is not too bad

300 distance calculations

$$d((x_1,y_1),(x_2,y_2)) = \sqrt{(x_1-x_2)^2+(y_1-y_2)^2}$$

two subtractions
two squarings
one addition
one square root

But the size of the nbhd is

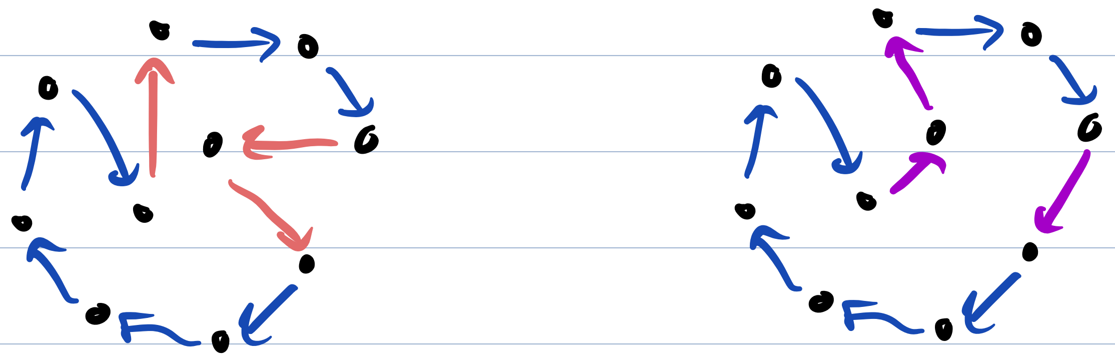
$$\binom{299}{2} = 44,551$$

Doing 300·44551 distance calculations <u>is</u> slow.

How can we speed this up?

(1) When you want the score of a tweaked tour, start with the score of the original tour, and alter it accordingly.

Original Tour → Nbhd



Score of new = [Score of old] − [3 orange distances] + [3 purple dist]

(2) Pre-calculate and store the distance between all $\binom{300}{2} = 150 \cdot 299$ pairs of cities.

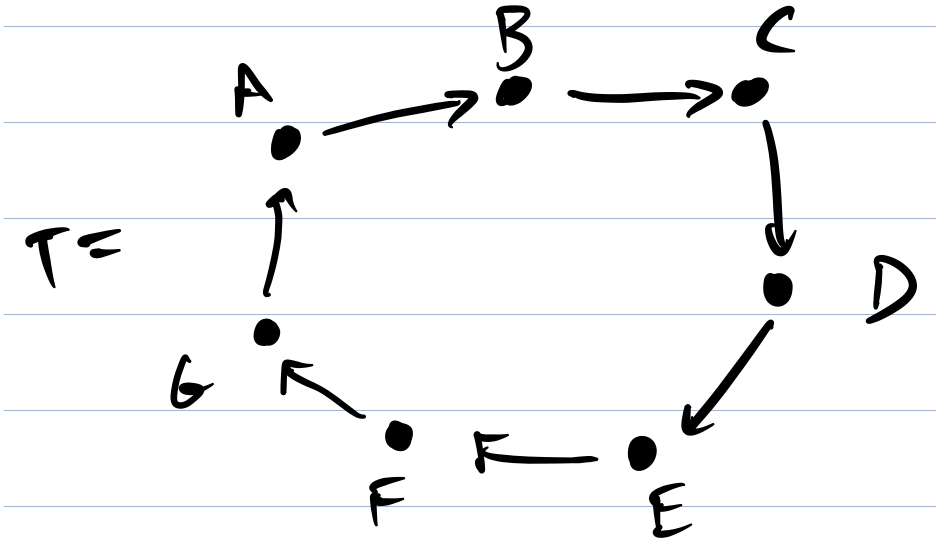
Now you don't need to do any more distance calculations.

You do still need to add all these distances for each tour.

For large problems, may not have enough memory.

Can be combined with (1).

Concrete example of (1):



Let d be the distance function.

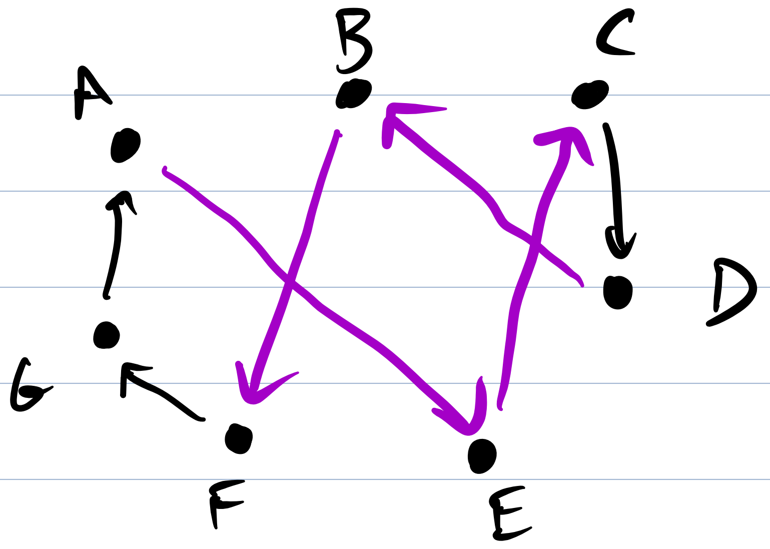
$$\text{score}(T) = d(A,B) + d(B,C) + d(C,D) + d(D,E) + d(E,F) + d(F,G) + d(G,A)$$

Swap B + E

A → B → C → D → E → F → G → A

A → E → C → D → B → F → G → A

$T' =$



$$\text{score}(T') = \text{score}(T) - [d(A,B) + d(B,C) + d(D,E) + d(E,F)] + [d(A,E) + d(E,C) + d(D,B) + d(B,F)]$$

If you have 300 cities:
new = old − 4 edges + 4 edges
8 distance calculations vs. 300

$\frac{300}{8} = 37.5x$ faster

Is this tweak (swapping two cities) a good tweak?

- * makes small changes
- * gets good results

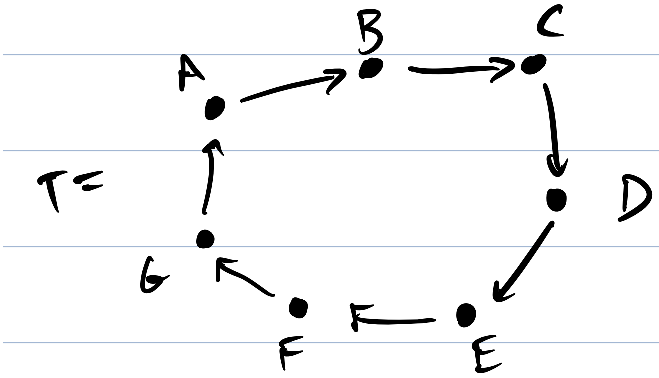
Demo, 50 cities, gives a bad result

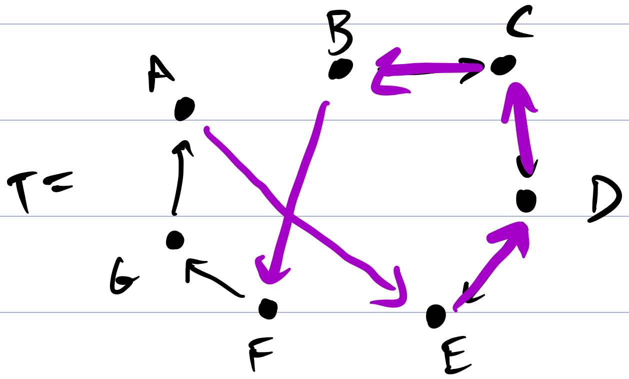
Can we think of another tweak?

Pick two cities and reverse the entire route in between them.

$$A \to \underbrace{B \to C \to D \to E} \to F \to G \to A$$

$$A \to E \to D \to C \to B \to F \to G \to A$$





changes 2 edges at a time, not 4

Many other possible tweaks.

Demo - much faster when we use the scoring tricks
- this new tweak (reverse a block) gives better results than the old one
- still slow because it looks at the whole nbhd

How can we adapt this for continuous spaces (when the nbhd is infinite?)

MH #3 n-trial steepest ascent

```
x = random element of S
while True:
    temp = x
    repeat n times:
        s = tweak(x)
        if score(s) > score(temp):
            temp = s
    x = temp
```

(temp is the best of the n tweaks)

if none of the tweaks beat x, then it stays the same

tweak = a random thing in the nbhd, and there are many different ways to do that

Next time: the n=1 version, which is just called "hill climbing".