3/17/2020

The Traveling Politician Problem

Advisor: Verbus M. Counts
Interns: Trent Rogers, Kranthi Raj Vellanki, Ember Carpenter

A politician hopes to become the president of the United States. Their campaign starts with the presidential primaries in the capital of Iowa. The politician then wants to visit the capital of every U.S. state to campaign before ending in the White House in the nation's capital of Washington, D.C. The politician does not want to visit any capital more than once. They would like to campaign in every capital one and only once. To be efficient and save on time and money, they would like to do this in as short a path as possible. The Traveling Politician Problem aims to find this shortest path. The map can be thought of as a graph with 51 points (the capitals of all 50 U.S. states, plus Washington, D.C.) and a set of distances between each of them. The starting point and ending point are already set (the capital of Iowa and Washington, D.C., respectively). This leaves 49 points to be visited in between the starting and ending points, this does not include the start and end points.

This problem is much harder than it may sound. The main solution to the problem is factorial time—that is, the time it takes to solve will be proportional to 49!. It is not 51! because the starting and ending cities are already set. After starting in Iowa, one of the 49 remaining capitals can be chosen as the first one to travel to. Now that one of the 49 has been chosen as the first, one of the remaining 48 capitals can be chosen as the second to travel to. Now there are 47 remaining capitals to choose as the third, and so on. The total number of paths to be compared will be 49*48*47* … *3*2*1, which is 49! (49 factorial). This evaluates to around 6*10^62 different total paths to be compared. That's around a trillion trillion trillion trillion trillion.

One particular mistake is very easy to make here: why not just find the shortest path from the capital of Iowa to any other one state capital, then take the shortest path from there to any other one state capital, and then keep going until you wind up in D.C.? This could possibly give you a better solution than trillions of trillions of other solutions, but it's unlikely to give you the very best overall path to D.C. For example, let's say we only visit Texas and California in the middle: the distance from Iowa to Texas is shorter than the distance from Iowa to California, so you go to Texas first and then to California and then to D.C. This is around 5,300 miles. It's longer to go to California first than to Texas, but if you visit California first, then Texas, then D.C., you get around 4,900 miles, which is the shorter path. As you can see, finding the shortest distance from one capital to another at any given point is not necessarily going to give you the shortest overall path to visit each capital only once.

This problem is based on the "Traveling Salesman Problem", which is a well-known graph theory problem that has been heavily studied by mathematicians. Many resources are available to study this problem under the title "Traveling Salesman Problem".

https://en.wikipedia.org/wiki/Travelling_salesman_problem

Programming Notes

What not to do:

1. We do not want to boil the ocean, that's what we do not want to do. This means do not start programming thinking you know what to do, YOU DO NOT.

What we WANT TO DO:

1. Start with MVP (minimum viable product). This means start the first version with just calculating the distance between the capital of Iowa and Washington D.C.
    a. What reference do we have?
        i. We have a table of all the zipcodes, with the longitude and latitude.
        ii. We have a program that can calculate the distance between two longitudes and latitudes using their zip codes. This is how we can compute the distance between these two places.
    b. Find the capitol of Iowa:
        i. How do we find this? This is where we use google.
    c. Then find the zipcode of the capitol of Iowa: Hint: Des Moines, IA **50319**.
        i. Get help from google once again
        ii. Use this to get the longitude and latitude
    d. Find the ending point Hint: Washington D.C **20515**
        i. Find the address of White House, and use that to find the zip code.
        ii. Use this to find the longitude and latitude.
    e. This is V-1.0 which allows us to find the distance between the starting and ending points. (Zero points in between the two cities).
2. **V1.1** (The .1 means that we have added a point between our start and end points.)
    a. Pick California. Hint: Sacramento, CA **95814**
        i. Find capitol/zip code
        ii. Find longitude and Latitude
    b. Output should include two things:
        i. Distance between Iowa and California (x)
        ii. Distance between California and Washington D.C (y)
        iii. Output should be the two above distances added together. (x + y = z)
        iv. Note: NO DECISIONS TO BE MADE
3. **V1.2** (two states between start and end point)
    a. Pick New York. Hint: Albany, NY **12242**
        i. Find capitol/zip code

ii. Find longitude and Latitude

b. Output should include two things:
   i. We have a decision to be made. Do we go Iowa to California OR Iowa to New York before going to Washington D.C.

c. First choice:
   i. Iowa to California to New York, than Washington D.C
      1. Distance between Iowa and California (w1)
      2. Distance between California and New York (x1)
      3. Distance between New York and Washington D.C (y1)
      4. Output should be the two above distances added together. (w1 + x1 + y1 = z1)

d. Second Choice:
   i. Iowa to New York to California, than Washington D.C
      1. Distance between California and New York (w2)
      2. Distance between Iowa and California (x2)
      3. Distance between New York and Washington D.C (y2)
      4. Output should be the two above distances added together. (w2 + x2 + y2 = z2)

e. Find the optimal route:
   i. Now we have to compare z1 and z2 and find the shortest distance.
   ii. Output the shortest route and distance.

4. **V1.3** (three states in between start and end point)
   a. Pick Washington State (NOT WASHINGTON D.C). Hint: Olympia, WA **98504**
      i. Find capitol/zip code
      ii. Find longitude and Latitude
   b. Output should include three things:
      i. We have a decision to be made. In which order should we visit the states?
      ii. How many possible routes are there?
         1. There are going to be 3! Routes, because we have 3 points in between the start and end points which we can choose from. For this one there would be 6 possible routes.
   c. Use previous method to solve for this
5. V1.N (n states between the start and end point)
   a. N = range (0, 49)
6. A way to keep track of our routes is to think of it as a Redis Key (like a Python Dictionary)
   a. I.E
      i. IA
      ii. CA

iii. NY
iv. WA
v. DC

b. The Redis Key would be:
   i. Key: IA:CA:NY:WA:DC Value: Zn(total route distance) This is a specific Route Distance
   ii. Pick the one with the smallest Z value out of the 6 (for this specific example).
   iii. Everything but the endpoints can change

c. Build String for each possible combination, then store them into a data structure.
   i. IA:CA:WA:NY:DC
   ii. IA:NY:CA:WA:DC
   iii. IA:NY:WA:CA:DC
   iv. IA:WA:CA:NY:DC
   v. IA:WA:NY:CA:DC

d. Prevent duplicates??
   i. Compare strings?

7. Parallel Processing (once we figure out the preventing duplicates) (take advantage of this as much as possible in any program you can)
   a. Use of the Fork function
8. Use Python 3.8, C++17, Elixir 1.10.2, Haskell 8.8.3 to code to versions 1.0 - 1.3
9. Feel free to contact Trent if you would like to collaborate/ask some questions
   a. trentr314@gmail.com
10. Everyone needs to do their own versions, in their own language, of 1.0 - 1.3, as well put the code on their own GitHub.
    a. Everyone also needs to create their own websites with a link to their GitHub (to use on your resumes)
       i. Lets employers know that you can create a website
       ii. Lets employers know that you can write code (from the code shared on your GitHub)
    b. The website and code on GitHub are critical to getting jobs.

Programming Notes II

One way: to build the strings (all possible routes)

Case of three: assign random number to each state abbreviation

Then sort codes by the random number assigned

This then builds Redis key

Store key in Redis database, set value to 0

Assign another set of random numbers to codes; sort them again; build another key; store in Redis database

This overrides it, continue until we have 6 unique keys in database,

Random number approach eventually will give us the 6 routes

Ex. iteration 1 (give them each a random number)

NY: 7

CA: 21

WA: 56

(sort them by number order), now we set the key in Redis database to zero, with the string being: IA: NY: CA: WA: DC (14 characters)

Iteration 2 (assign another set of random numbers)

CA:78

WA: 45

NY: 97

Sort: WA:CA:NY (if this key already exists in database, then it keeps reiterating until we have 6 unique keys)

Key: IA:WA:CA:NY:DC (set redis key to zero again)

Using our advantage that Redis can't hold duplicate keys!

Question for the group:

In what order do we select the next state abbreviation to be added to the Redis key?