Holly Do - COMP 614

Homework 6

The URL for your final CodeSkulptor file:

https://py3.codeskulptor.org/#user310_HKmbd7oxYR_0.py

Question 3.A:

Question 3.A.i: What is/are the base case(s) for DFS recursive function? For each base case, you should clearly identify the following:

The when: The condition(s) under which we fall into this base case. You can & should refer to any parameters used by name.
The what: A detailed explanation of what needs to happen when we're in this base case. You can & should refer to any parameters used by name

Since we are searching in depth—meaning we follow a path to the end of a branch—the base case occurs when the current node has no neighbors. In the case where the current node has already been visited, meaning it is already recorded in parent.keys(), we pass and move on to inspect the next iteration for nbr in nbrs set.

In the base case, since we only care about the parent dictionary and not the distance dictionary, the function should pass, and the original parent dictionary won't have any modifications to it.

Question 3.A.ii: What is/are the recursive case(s) for this function? For each recursive case, you should clearly identify the following:

The when: The condition(s) under which we fall into this recursive case. You can & should refer to any parameters used by name.
The what: A detailed explanation of what needs to happen when we're in this recursive case. You can & should refer to any parameters used by name. You are also welcome to assign names to local variables that you intend to define. For any recursive calls that you intend to make, you should clearly specify what the inputs will be in terms of these parameters and local variables.

Recursive cases occur when there is more depth to explore—meaning the current node has one or more neighbors that have not yet been visited.

In the recursive case, the parameters should be updated the start_node to be the neighbor node. This means that the subsequent recursive call will treat the neighbor as the new starting point. If there is still depth to explore, the function will continue calling itself recursively until the base case is met, which is when a node has no more unvisited neighbors. At that point, no further recursive calls are made.

Question 3.B:

Question 3.B.i: Assume that you are not using any list comprehensions in your implementation. What is the minimum "depth" that your loop nest will need to be in order to implement this Page Rank algorithm? Why? You should clearly describe the purpose of each loop.

Page rank should have a loop with minimum depth of 3. First while loop is to ensure the delta is less than threshold 10^-8. Second loop is to loop through the nodes in the page_rank1 dict to update the ranks. Third loop is to loop through all the inbounds of current node to calculate their ranks' sum. This piece of data is then used to calculate the rank of the current node.

Question 3.B.ii: How will you ensure that you are updating all of the page ranks "simultaneously"? In other words, how will you ensure that you are using the old $\mathrm{PageRank}k-1$ values rather than some mix of old and new values in order to compute the new $\mathrm{PageRank}k$ values?

I made a copy of $\mathrm{PageRank}-1$ values named page_rank_image, which records the values at k-1 iteration. Then I made updates to the main page_rank1 dictionary at iteration k using values from page_rank_image.

Question 3.B.iii: If we think of the page ranks in terms of a random walk on the graph, what does the damping factor, d, represent? If we were to set d to 1, in what way(s) could that adversely impact the page ranks computed by the algorithm, and why? If we were to set d to 0, in what way(s) could that adversely impact the page ranks computed by the algorithm, and why?

If we think of PageRank in terms of a random walk on a graph, the damping factor represents the probability that the current iteration will follow the distribution from the previous iteration. If the damping factor is set to d, the walker follows the previous iteration's distribution with probability d, and with probability 1 - d, it randomly jumps to any node in the graph.

In the case where there are no outbound edges for a particular node (a "sink node"), and the damping factor d is set to 1, the walker has a 0% chance (1 - d = 0) of escaping the sink node. As a result, these nodes absorb all the ranking from the graph, leading to a misrepresented distribution. If we were to set d to 0, meaning the walker has 0% chance of following the previous distribution, thus the accuracy of the rank will be greatly compromised.

Question 3.B.iv: What does delta represent? Why do we want to stop when $\mathrm{deltak} < 10-8$?

We want to stop at a small threshold (such as 10^-8) to limit the number of iterations. Doing so saves memory and time while ensuring that the result has converged closely enough to the equilibrium state to represent an accurate enough distribution.

Discussion Questions:

Run PageRank on a graph built from wikipedia_articles_streamlined.txt with a damping factor of 0.85. Which ten articles have the greatest page ranks? Provide the top ten articles and their page ranks in the form of a list of tuples of (title, page rank), sorted in descending order. Do the results align with your expectations? Why or why not? Why do you think that these pages had the highest ranks?

Top 10 ranked pages are:

[('christianity', 0.011312675117802505), ('soviet union', 0.010042061855732 76), ('european union', 0.00955194826127956), ('roman empire', 0.00783236891615 8966), ('europe', 0.007825005084123865), ('india', 0.007600195827660242), ('china', 0.007534685124097696), ('middle ages', 0.007236654265055156), ('nazi germany', 0.007026348927497254), ('hinduism', 0.007009885604633514)]

These results make sense in terms of the number of people associated with these subjects, as well as their global significance. The popularity of major religions is understandable, as each has millions, if not billions, of followers. Other topics, such as historical events and China, are also global issues that have involved large numbers of people, thus tend to be the most popular.

Homework Reflection:

1. What were the two most important concepts and/or skills that were reinforced by this assignment? Why do you believe that these concepts are important? (Your response must be 3-6 sentences.)

Two most important concepts I've learned from this assignment is understanding how to implement page_rank algorithm using graph data structure and optimizing solution using recursion. I also paid attention to the difference between breath first search, which will return one of the shortest paths, versus depth first search, which will return a path by exploring the depth of a graph.

2. How do the skills and concepts that you applied in this assignment transcend the specific application/problem that you were tackling? How could you envision applying these skills and concepts again in the future? (Your response must be 2-4 sentences.)

I will use recursive functions in the future to optimize functional code to reduce run time. Page_rank algorithm can be applied in different subjects such as ranks of social media accounts, importance of articles/papers, to find the most referred to nodes among the graphs. Breath first search is useful to find shortest paths from a starting node, while depth first search will find a path from starting node by exploring the depth of the graph.

3. What do you believe you did well on this assignment? If you could do this assignment over, would you do anything differently? Why or why not? (Your response must be 2-4 sentences.)

I think I got the results correctly. If I could work on this assignment again, I would ensure that I fully understand the difference between depth first search and breath first search and the cases where each of these search algorithms applies.

4. Do you think you're comfortable enough with the concepts covered in this assignment that you would be capable of teaching them to a peer? Why or why not? (Your response must be 2-4 sentences.)

   I am comfortable enough to teach a peer the lessons I have learned. This assignment is important for helping us become familiar with the graph representation of entities and the analysis of the neighbors of these entities using graph's structure and overarching mathematic methods.