Completely optional additional questions a.k.a. Homework 11

DS-210 @ Boston University

Spring 2022

1. In lecture 31, Professor Onak showed an incorrect implementation of Dijkstra's algorithm. Instead of removing the vertex with the lowest associated distance from the priority queue, he was removing the one with the highest. Surprisingly, for the sample graph, the incorrect implementation still computed correct distances.

   You can find both the incorrect implementation and its corrected version at:

   - https://onak.pl/teaching/download/2022-spring-ds210/dijkstra.tar.xz
   - https://onak.pl/teaching/download/2022-spring-ds210/dijkstra.zip

   (a) Your task is to either find an example for which the incorrect implementation computes incorrect distances or show that it still computes correct distances from the start vertex. For the former, demonstrate it by modifying `sample_graph.rs` in the provided implementation. The only limitations are that all edge weights should be integers between 0 and 1000 and the number of vertices should be at most 1000.

   (b) What is the running time of the incorrect implementation?

2. (a) Write a function that takes a reference to a vector of integers and returns the maximum sum of any subset of consecutive entries of the vector, including the empty subset.

   (b) What is the time complexity of your function? Do you think there could be a better solution?

   (c) Rewrite your function to take a slice and return the subslice that maximizes the sum. What are advantages and disadvantages of using slices in this case?

3. Create a *viral* meme that demonstrates an important concept that you learned in DS-210. Post it on one or more social networks and watch how it spreads like wildfire. Is the process of its spreading

   - similar to DFS?
   - similar to BFS?
   - very different from both DFS and BFS? (If you give this answer, can you come up with a good model for the process of its spreading?)

4. Suppose that your goal is to write a function that will be used multiple times throughout your execution of your application. In particular, it may be called multiple times for different parts of your input vector of integers. This function, given access to a vector of integers, has to compute the number of even entries.

(a) Implement the function.

(b) What is the best argument for this function to take? Should the vector be moved into the function? Should the function take a mutable reference, immutable reference, mutable slice, or immutable slice? Explain why.

(c) What is the running time of your function?

(d) If you wrote your function using iterators, can you write it using regular loops? If you wrote it using regular loops, can you write it using iterators?

5. Create a single data type that can be used for representing the following functions f:

   - $f(x) = ax^2 + bx + c$ for arbitrary a, b, and c of type `f64`
   - $f(x) = \sin(\alpha x)$ for arbitrary α of type `f64`

   Provide a method that takes x and computes $f(x)$.

6. Write a function that receives two vectors of integers and computes the number of *different* values $x + y$, where x belongs to the first vector and y to the second. For `[2,7,4]` and `[1,4]`, the correct output is 5.

   Bonus question: Can you rewrite the function in such a way that it can be used for different integer types?

   In both cases, assume that all sums can be represented via an integer of the same type.

Preparing for the final

When trying to solve the above problems, put your computer away. Instead try to write a solution on paper. Only once you are done, copy it to your computer to see if it compiles. Run it on different inputs to discover any potential problems.

Try to come up with variations of your problems. Try to solve them.