CSCI 3104 Fall 2021
Instructors: Profs. Grochow and Waggoner

Final- Standard 3

Due Date . . . . Dec / 15th / 2021
Name . . . . **Michael Ghattas**
Student ID **109200649**

Contents

1 Instructions

- The solutions **should be typed**, using proper mathematical notation. We cannot accept hand-written solutions. Here's a short intro to LaTeX.
- You should submit your work through the **class Canvas page** only. Please submit one PDF file, compiled using this LaTeX template.
- You may not need a full page for your solutions; pagebreaks are there to help Gradescope automatically find where each problem is. Even if you do not attempt every problem, please submit this document with no fewer pages than the blank template (or Gradescope has issues with it).
- You **may not collaborate with other students**. **Copying from any source is an Honor Code violation. Furthermore, all submissions must be in your own words and reflect your understanding of the material.** If there is any confusion about this policy, it is your responsibility to clarify before the due date.
- Posting to **any** service including, but not limited to Chegg, Discord, Reddit, StackExchange, etc., for help on an assignment is a violation of the Honor Code.
- You **must** virtually sign the Honor Code (see Section 2). Failure to do so will result in your assignment not being graded.

2 Honor Code (Make Sure to Virtually Sign)

Problem 1.

- My submission is in my own words and reflects my understanding of the material.
- I have not collaborated with any other person.
- I have not posted to external services including, but not limited to Chegg, Discord, Reddit, StackExchange, etc.
- I have neither copied nor provided others solutions they can copy.

I agree to the above, Michael Ghattas. □

3 Standard 3- Dijkstra

3.1 Problem 2

Problem 2. Recall that in lecture, we used a binary heap to implement the priority queue. Suppose instead we used a data structure with the following time complexities:

- Insertion: $O(\sqrt{n})$.
- Removing the first element from the priority queue: $O(\sqrt{n})$.
- Updating an element's position: $O(1)$.

Carefully analyze the time complexity of Dijkstra's algorithm, using this new data structure to implement the priority queue. [**Note:** It may be helpful to have the course lecture notes open during this quiz.]

Answer:

- We note that Dijkstra's Algorithm examines each vertex once, taking $O(\sqrt{n})$ steps at each iteration. Here, the $O(\sqrt{n})$ complexity comes from pushing each vertex into the priority queue. So the complexity of the initial part is $O(|V| \cdot \sqrt{|V|})$, where $|V|$ is the number of vertices in the graph.
- Now Dijkstra's Algorithm examines each edge of G exactly once. Then, it polls a single vertex from the priority queue. As G is connected, we poll each vertex from the queue exactly once. As polling takes time $O(\sqrt{n})$, this adds complexity $O(|V| \cdot \sqrt{|V|})$.
- Now when we evaluate each edge, we at most update the position of a vertex in the priority queue. This accounts for time complexity $O(|E| \cdot 1)$, where $|E|$ is the number of edges in the graph.

Thus, the time complexity of Dijkstra's algorithm, given the new data structure and when using a binary heap as our priority queue, is $O(|V|\sqrt{|V|}) + |E|)$. □