Section 10: Final Review

1. Reduction: Shortest Paths

Design an efficient algorithm for the following problem: Given a weighted, directed graph G where the weights of every edge in G are all integers between 1 and 10, and a starting vertex s in G, find the distance from s to every other vertex in the graph (where the distance between two vertices is defined as the weight of the shortest path connecting them, or infinity if no such path exists).

Your algorithm **must run faster (asymptotically) than Dijkstra's**.

Show the runtime of your algorithm in terms of V and E (the number of vertices and edges in G).

2. Design Decisions Out the Wazoo

(a) Given each of the following scenarios, choose the appropriate sorting algorithm and justify your choice with a short sentence.

insertion sort, selection sort, merge sort, quick sort, heap sort

(i) You are writing a program that sorts applicants for a prestigious fellowship. Each application is given a numerical score to provide an ordered ranking of applications. If two applications have the same score, the application that was received first will be selected, so the order in which the applications were submitted should be maintained.

(ii) You are a triage nurse and you are in charge of the queue that determines who sees the doctor first. You keep a single sorted list in a spreadsheet, and as new patients arrive you insert them into the queue based on the severity of their injury. If two patients have the same severity the patient that arrived first should get to see the doctor first.

(iii) You are writing a program that sorts test scores for a class' final. The finals will be received in the order they are finished which generally means that higher scores will be received before lower scores, but not perfectly. The sorting will be used to determine the exam statistics, so the ordering of equivalent scores does not matter.

(iv) You are a teacher and you release your students for lunch. The students know they are supposed to let younger students go first, but in their excitement, they forget and line up all out of order. Students don't like letting younger ones cut but are willing to swap line positions with a younger student if you ask them. There is not space in the cafeteria for an extra line, so you have to direct the students within the same line.

(v) At class pictures the photographers usually sort the children by height before assigning places. The photographer doesn't care who was originally where in line but he/she does need to get them sorted as fast as possible and there's a lot of extra space in the room to arrange them.

(vi) A company has lists of numbers that each need to be sorted. Because the lists can be long, short, reversed, in-order or a mix of all these situations, they need a sort that will always have the same tight big-O runtime regardless of the condition of the list.

(vii) When you were younger, you most likely had a box of Crayola crayons. When you first buy them however, all the colors are not sorted in order. Which sort can you use to sort the crayons by color in the box such

that you only take one or two of the crayons out of the box at any given time?

(viii) A customer requires that you use a sort that has recursion. Which one can you choose? Explain why this request is not a good idea.

(b) Given each of the following scenarios choose the appropriate ADT, justify your choice with a short sentence.

List, Stack, Queue, Dictionary, Heap, Disjoint Sets

(i) You are writing a program to manage a todo list with a very specific approach to tasks. This program will order tasks for someone to tackle so that the most recent task is addressed first. How would you store the transactions in appropriate order?

(ii) You are writing a study support program that creates flash cards where each flash card has two pieces of crucial information: the question to be asked and the correct answer. How would you store these flash card objects?

(iii) You are writing a program to store the history of transactions for a small business. You need to maintain the order in which the transactions occurred and be able to access entries based on the order in which they were received.

(iv) You are writing a program to surface candidates for a job. As candidates are met and evaluated, they are added to a collection from which hiring managers can request the next best applicant whenever they have an open job position. How would you store the candidates and their evaluations for hiring managers to be able to quickly get the next best applicant?

(v) You are writing a program to schedule jobs sent to a laser printer. The laser printer should process these jobs in the order in which the requests were received. How would you store the jobs in appropriate order?

(vi) You are developing a video game where you play as the Roman Empire. Every time you conquer a city-state, that city-state and everything they own is added to your empire.

(c) Given each of the following scenarios, choose the appropriate dictionary implementation and justify your choice with a short sentence.

Array Dictionary, LinkedList Dictionary, AVL Dictionary, Hash Dictionary

(i) You are writing a program to store incidents in a software service you maintain. The keys will be time stamps, so you know they will be unique. You will be adding incidents as they occur and removing them as they are resolved. You are more likely to need to access and remove incidents that have recently been added to the collection.

(ii) You are writing a program that implements a company directory. The directory will store employee ids mapped to contact information. You will also implement auto complete, so that as someone types in a user id you will suggest for them possible entries. Your directory will need to grow and shrink as people join and leave the company.

(iii) You are writing a program to store an extremely large dictionary of English vocabulary. You will know how many entries you will store at the onset of the program and will not add any more after the initial processing. You want to optimize for quick look ups of definitions.

(iv) You are writing a program to store a rather small dictionary that maps exam questions to the average score for that question across all students. The questions are numbered sequentially starting at 0. Often

you will want to read the entire set of scores in the order of the test.

3. More Sorting

During lecture, we focused on five different sorting algorithms: insertion sort, merge sort, quick sort, selection sort, and heap sort.

(a) Suppose we want to sort an array containing 50 strings. Which of the above four algorithms is the best choice?

(b) Suppose we have an array containing a few hundred elements that is almost entirely sorted, apart from a one or two elements that were swapped with the previous item in the list. Which of the algorithms is the best way of sorting this data?

(c) Suppose we want to sort an array of ints, but also want to minimize the amount of extra memory we want to use as much as possible. Which of the above algorithms is the best choice?

(d) Suppose we have a version of quick sort which selects pivots randomly and creates partitions in the manner described in lecture. Explain how you would build an input array that would cause this version of quick sort to always run in $\mathcal{O}\left(n^2\right)$ time.

Your answer should explain on a high level what your array would look like and what happens when you try running quick sort on it. You do not need to give a specific example of such a array, though you may if you think it will help make your explanation more clear.

(e) How can you modify both versions of quicksort so that they no longer display $\mathcal{O}\left(n^2\right)$ behavior given the same inputs?

4. Sorting Code Modeling

Following is a pseudocode for a sorting algorithm. Study the code and answer the following questions.

Note: You have not seen this sorting algorithm in class. Part of the exercise is to apply your analysis skills. Read the questions before analyzing the code.

```
public void fooSort(int arr[])  {
   int n = arr.length;
   for (int i = 0; i < n-1; i++) {
      for (int j = 0; j < n-i-1; j++) {
         if (arr[j] > arr[j+1]) {
            // swap temp and arr[i]
            int temp = arr[j];
            arr[j] = arr[j+1];
            arr[j+1] = temp;
         }
      }
   }
}
```

(a) What is the worst-case tight big-O runtime?

(b) Does the above algorithm do an in-place sort?

(c) Does the above algorithm do a stable sort?

5. Graphs and Design

Consider the following problems, which we can both model and solve as graph problems.

For each problem, describe (i) what your vertices and edges are and (ii) a short (2-3 sentence) description of how to solve the problem.

We will also include more detailed pseudocode to make solutions.

Your description does not need to explain how to implement any of the algorithms we discussed in lecture. However, if you *modify* any of the algorithms we discussed, you must discuss what that modification is.

(a) A popular claim is that if you go to any Wikipedia page and keep clicking on the first link, you will eventually end up at the page about "Philosophy". Suppose you are a given some Wikipedia page as a random starting point. How would you write an algorithm to verify this claim for the given starting point?

(b) Suppose you were walking in a field and unexpectedly ran into an alien. The alien, startled by your presence, dropped a book, ran into their UFO, and flew off.

This book ended up being a dictionary for the alien language – e.g. a book containing a bunch of alien words, with corresponding alien definitions.

You observe that the alien's language appears to be character based. Naturally, the first and burning question you have is what the alphabetical order of these alien characters are.

For example, in English, the character "a" comes before "b". In the alien language, does the character "$\daleth$" come come before or after character "ϱ"? The world must know.

Assuming the dictionary is sorted by the alien character ordering, design an algorithm that prints out all plausible alphabetical orderings of the alien characters.

6. More Graph Design Decisions

For each of the following graph-based computational tasks, specify the type of graph most appropriate for the data in question in terms of undirected or directed, and unweighted or weighted.

In addition, choose the graph algorithm from the following list best suited to computing a solution:

BFS, DFS, MST (e.g. Prim's or Kruskal's), Dijkstra's, Topological Sort

(a) You want to model a collection of Java files in a project. You know which files exist and which other Java files they depend on using (i.e. other classes you're importing). You want to determine the order that the files have to be compiled in before a given file f can be compiled and run.

(b) You want to model how a tweet gets re-tweeted by followers on Twitter. You have data on who the users of Twitter are and all the followers of each user. Given a source, a tweet can be re-tweeted by any follower of that source. If a tweet gets arbitrarily re-tweeted k times, you want to know which users could have seen the tweet.

7. Greenland Always Survives

Bob runs a pizza shop in a small town in Greenland. There are 100 houses in the town connected by roads. Bob delivers to every house in the town. Unfortunately, Bob can only carry one pizza at a time, so after delivering to one house, he has to come back to the shop to pick up another delivery.

Last week, there was a storm that left 4 feet of snow on all the roads in the town. Clearing snow on a road costs Bob some money, but once the snow is cleared the cost to travel on the cleared road (in either direction) is zero.

Bob decides that instead of clearing snow from all the roads on his delivery routes at once, he will clear the snow while delivering pizzas.

(a) Bob knows he will get an order for each of the 100 houses (they're all regular customers), so he'd like to plan in advance which roads he is going to clear. How should he choose which roads to use to minimize the total cost?

(b) When each new order comes in, how should Bob determine what roads to take to get to that house?

8. Frodo and Sam

Frodo and Sam are on their way to Mordor to destroy the ring, and along their path they have to pass through several human villages on their way, some of which are now deserted and likely Orc territory. They learn that there are some paths that are being heavily guarded by Orcs, and they want to avoid those paths at all costs. Friendly spies tell Frodo and Sam that they should avoid the road between two deserted villages, as it is more likely be monitored by Orcs.

(a) How would you represent a graph to capture this problem? Answer the following questions:

What do your vertices represent? If you store extra information in each vertex, what is it?

What do your edges represent? Your answer may be a real-world object or an abstract description of what edges will exist. If you store extra information in each edge, what is it?

Is your graph directed or undirected? Briefly explain why in 1-2 sentences.

Is your graph weighted or unweighted? Briefly explain why in 1-2 sentences.

Do you permit self-loops? Parallel edges? Briefly explain why in 1-2 sentences.

(b) How should Frodo and Sam find the shortest and safest path to Mordor? State the runtime of your solution.

9. Get Zucced

Suppose we model Facebook users with the following graph representation. An undirected, unweighted adjacency list where vertices are users and edges represent 'friend' relationships. Thus, if two users are friends, then we have an edge between those two users.

Given this graph representation, explain how to solve the following problems.

(a) How would you find a person with the most friends in the network? State the runtime of your solution.

(b) The company wants to introduce a new metric for each user called `networkSize`. The `networkSize` of a user u is the number of users v such that $u \to v$, i.e., there is a path from u to v. Describe how you would calculate `networkSize` for a user u. State the runtime of your solution.

(c) Suppose you won a competition at Facebook, and you are granted 3 "free" friend requests, i.e., any three users you choose will automatically become your friends. You want to choose three users who can maximize your `networkSize`. Describe an efficient solution for choosing three such users. State the runtime of your approach.

10. Memory, Locality, and Dictionaries

(a) In lecture, we discussed three different optimizations for disjoint sets: union-by-size, path compression, and the array representation.

If we implement disjoint sets using `Node` objects with a "data" and "parent" field and implement the first two optimizations, our `find` and `union` methods will have a nearly-constant average-case runtime.

In that case, why do we bother with the array representation?

(b) Suppose you implement a dictionary with a sorted array and another with an AVL tree. Consider the time needed to iterate over the key-value pairs of a `SortedArrayDictionary` vs an `AvlDictionary`. It turns out that iterating over the `SortedArrayDictionary` is nearly 10 times faster than iterating over the `AvlDictionary`. Think about why that might be.

Now, suppose we take those same dictionaries and try repeatedly calling the `get(...)` method a few hundred thousand times, picking a different random key each time.

Surprisingly, we no longer see such an extreme difference in performance. The `SortedArrayDictionary` is at most only about twice as fast as the `AvlDictionary`.

Why do you suppose that is? Be sure to discuss both (a) why the difference in performance is much less extreme and (b) why `SortedArrayDictionary` is still a little faster.

11. Code Modeling

Consider the following problems. Don't forget the **cheat sheet** at the end of this worksheet, which has some summations you can use.

(a) What is the simplified tight O bound for the runtime of each of the loops below? What is the simplified tight O bound for the runtime of method1?

```
public void method1(int n) {
    for (int i = 10; i < n; i++) {
        System.out.println("373");
    }

    for (int i = n; i >= 1; i /= 2) {
        System.out.println("is");
    }

    for (int i = 0; i < 9999999; i++) {
        System.out.println("cool");
    }

    for (int i = n; i >= n - 4; i-=1) {
        System.out.println("I guess");
    }
}
```

(b) This time, write out the summation of method2 as well as give the simplified tight oh bound for the runtime of this method in terms of n.

```
public void method2(int n) {
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < i; j++) {
            for (int k = 0; k < i; k++) {
                System.out.println("Hi, my name is " + n);
            }
        }
    }
}
```

(c) What is the simplified tight oh bound for the runtime of each of the loops below? Ignore the context of the if conditions for now. Finally, give the overall simplified runtime of method3.

```
public void method3(int n) {
    if (n < 10000) {
        for (int i = 0; i < n * n * n; i++) {
            System.out.println(":0");
        }
    } else if (n == 10001) {
        for (int i = 0; i < n * n * n * n; i++) {
            System.out.println(":/");
        }
    } else {
        for (int i = 0; i < 2 * n; i++) {
            System.out.println(":D");
        }
    }
}
```

(d) What is the simplified tight oh bound for the runtime of each of the loops above? Ignore the context of the if conditions for now. Finally, give the overall simplified runtime of method4.

```
public void method4(int n) {
    for (int i = 0; i < n; i++) {
        if (i == 0) {
            for (int j = 0; j < n; j++) {
                System.out.println("lol");
            }
        } else if (n < 1000) {
            for (int j = 0; j < n * n; j++) {
                System.out.println("rofl");
            }
        } else {
            for (int j = 0; j < 10000; j++) {
                System.out.println("haha");
            }
        }
    }
}
```

12. Debugging

Suppose we are in the process of implementing a hash map that uses open addressing and quadratic probing and want to implement the `delete` method.

(a) Consider the following implementation of `delete`. List every bug you can find.

Note: You can assume that the given code compiles. Focus on finding run-time bugs, not compile-time bugs.

```
1  public class QuadraticProbingHashTable<K, V> {
2      private Pair<K, V>[] array;
3      private int size;
4
5      private static class Pair<K, V> {
6          public K key;
7          public V value;
8      }
```

```
9
10              // Other methods are omitted, but functional.
11
12              /**
13               * Deletes the key-value pair associated with the key, and
14               * returns the old value.
15               *
16               * @throws NoSuchKeyException if the key-value pair does not exist in the method.
17               */
18              public V delete(K key) {
19                  int index = key.hashCode() % this.array.length;
20
21                  int i = 0;
22                  while (this.array[index] != null && !this.array[index].key.equals(key)) {
23                      i += 1;
24                      index = (index + i * i) % this.array.length;
25                  }
26
27                  if (this.array[index] == null) {
28                      throw new NoSuchKeyException("Key-value pair not in dictionary");
29                  }
30
31                  this.array[index] = null;
32
33                  return this.array[index].value;
34              }
35          }
```

(b) Let's suppose the `Pair` array has the following elements (pretend the array fit on one line):

["lily", V_2]	["castle", V_6]	["resource", V_1]	["hard", V_9]	["bathtub", V_0]
["wage", V_4]	["refund", V_7]	["satisfied", V_6]	["spring", V_8]	["spill", V_3]

And, that the following keys have the following hash codes:

Key	Hash Code
"bathtub"	9744
"resource"	4452
"lily"	7410
"spill"	2269
"wage"	8714
"castle"	2900
"satisfied"	9251
"refund"	8105
"spring"	6494
"hard"	9821

What happens when we call `delete` with the following inputs? Be sure write out the resultant array, and to do these method calls *in order*. (**Note**: If a call results in an infinite loop or an error, explain what happened, but don't change the array contents for the next question.)

(i) `delete("lily")`

(ii) `delete("spring")`

(iii) `delete("castle")`

(iv) `delete("bananas")`

(v) `delete(null)`

(c) List four different test cases you would write to test this method. For each test case, be sure to either describe or draw out what the table's internal fields look like, as well as the expected outcome (assuming the `delete` method was implemented correctly). **Hint**: You may use the inputs previously given to help you identify tests, but it's up to you to describe what kind of input they are testing generally.

Master Theorem

For recurrences in this form, where a, b, c, e are constants:

$$T(n) = \begin{cases} d & \text{if } n \leq \text{some constant} \\ aT(n/b) + e \cdot n^c & \text{otherwise} \end{cases}$$

$$T(n) \text{ is } \begin{cases} \Theta(n^c) & \text{if } \log_b(a) < c \\ \Theta(n^c \log n) & \text{if } \log_b(a) = c \\ \Theta\left(n^{\log_b(a)}\right) & \text{if } \log_b(a) > c \end{cases}$$

Useful summation identities

Splitting a sum

$$\sum_{i=a}^{b}(x+y) = \sum_{i=a}^{b} x + \sum_{i=a}^{b} y$$

Adjusting summation bounds

$$\sum_{i=a}^{b} f(x) = \sum_{i=0}^{b} f(x) - \sum_{i=0}^{a-1} f(x)$$

Factoring out a constant

$$\sum_{i=a}^{b} cf(i) = c\sum_{i=a}^{b} f(i)$$

Summation of a constant

$$\sum_{i=0}^{n-1} c = \underbrace{c + c + \ldots + c}_{n \text{ times}} = cn$$

Note: this rule is a special case of the rule on the left

Gauss's identity

$$\sum_{i=0}^{n-1} i = 0 + 1 + 2 + \ldots + n - 1 = \frac{n(n-1)}{2}$$

Sum of squares

$$\sum_{i=0}^{n-1} i^2 = \frac{n(n-1)(2n-1)}{6}$$

Finite geometric series

$$\sum_{i=0}^{n-1} x^i = 1 + x + x^2 + \ldots + x^{n-1} = \frac{x^n - 1}{x - 1}$$

Infinite geometric series

$$\sum_{i=0}^{\infty} x^i = 1 + x + x^2 + \ldots = \frac{1}{1-x}$$

Note: applicable only when $-1 < x < 1$