CS41 Lab 3

September 21, 2020

In typical labs this semester, you'll be working on a number of problems in groups of 3-4 students. You will not be handing in solutions; the primary purpose of these labs is to have a low-pressure space to discuss algorithm design. However, it will be common to have some overlap between lab exercises and homework sets.

The goal of this lab session is to gain more experience with asymptotic analysis. Do not expect to complete all parts of all problems by the end of the lab. Consider it a successful lab session if you can complete the first problem and make reasonable progress on either the second or third problem.

For these problems, your example functions should have domain and range the positive integers $\mathbb{N}$.

1. **Better Runtime for Gale-Shapley Algorithm.** In class, we saw that the Gale-Shapley Algorithm runs in $O(n^4)$ time. However, with a bit more care on selecting data structures, this runtime can be improved to $O(n^2)$.

   ```
   Gale-Shapley()
   Initialize all hospitals, doctors := free.
   while there is a free hospital h and a doctor she hasn't proposed to:
       d := highest ranked doctor in h's preference list that h hasn't yet proposed to
       if d is freee
           (h, d) become engaged.
       else if d prefers h to h'
               (h, d) become engaged.
               h' becomes free again.
   return the set S of engaged pairs.
   ```

   (a) First, focus on each iteration of the While loop. We've given you two below.

   - Identify a free hospital h who hasn't proposed to everyone
   - Identify the highest ranked doctor d in h's preference list that h hasn't yet proposed to.

   What other tasks must be performed?

   (b) For each task, identify a data structure that can be used to manage this data/task, and analyze how much time it will take to initialize the data and how much time the task will take during the while loop.

   As an example, we've given you a solution for the first task below.

   - For the first task (identify a free hospital h who hasn't proposed to everyone), we start with all hospitals as free. We want to delete a hospital from this list when it gets engaged, and add a hospital when an engagement is broken. However, of all the free hospitals that haven't yet proposed to everyone, we're free to choose which is next.
     Solution: Use a LinkedList FreeHospitals. Initialize this list to contain all hospitals. Call First() to get the next free hospital. Add hospitals to the end of the list.
     Initialization time: $O(n)$. Loop Iteration time: $O(1)$.

(c) Now, pull everything together to analyze the overall runtime. How much time is spent before the while loop? How much time is spent during each iteration of the while loop?

We've seen in class that there are $O(n^2)$ loop iterations. In order to get an overall $O(n^2)$ runtime, your preprocessing (i.e., work before the while loop) step must take $O(n^2)$ time and your work inside each loop iteration must run in $O(1)$ time.

2. **Rates of Growth.** Arrange the following functions in ascending order of growth rate. That is, if g follows f in your list, then it should be the case that $f = O(g)$.

- $f_1(n) = n^{2.5}$
- $f_2(n) = \sqrt{2n}$
- $f_3(n) = n + 10$
- $f_4(n) = 10^n$
- $f_5(n) = 100^n$
- $f_6(n) = \log_{1.1}(n)\sqrt[3]{n}$
- $f_7(n) = n^n$
- $f_8(n) = n^2 \log_2(n)$
- $f_9(n) = n^{\log_2(n)}$

No proofs are necessary; just arrange the functions in ascending order of growth.

3. **Asymptotic analysis.** Assume you have functions f and g such that $f(n)$ is $O(g(n))$. For each of the following statements, decide whether you think it is true or false and give a proof or counterexample.

(a) $\log_2(f(n))$ is $O(\log_2(g(n)))$.

(b) $2^{f(n)}$ is $O(2^{g(n)})$.

(c) $(f(n))^2$ is $O((g(n))^2)$.

4. **Asymptotic Proofs.** Let $f(n) = 2(\log(n))^3 + 6$ and $g(n) = 5n^{1/4} + 10$. Prove that $f(n) = O(g(n))$. You may use techniques and facts from class and the textbook. Your proof should be complete and formal.