HW 2 due 5/28

CS3000: Algorithms & Data
Drew van der Poel

Midterm - 6/1

Lecture 12

- Midterm Review

May 27, 2021

Regrade Policy

- If you believe there was a mistake in grading your work:
  - 1. Make a **regrade request** on Gradescope -> the TA who graded your HW will review. If you remain unhappy with your grade, then
  - 2. You may issue a regrade challenge with Drew

- **Regrade challenge**:
  - Drew reserves the right to regrade the entire problem that you are challenging
  - If your new score is > your old score, you win the challenge. Otherwise, you lose the challenge.
  - If you lose two challenges over the course of the semester, you can no longer challenge
  - Not losing any challenges will be a small bonus to your final grade
  - Losing exactly 1 challenge will be a smaller bonus to your final grade
  - You have **one week** from the time grades are released to challenge

“Cover Page”

Quiz Instructions

You may have one sheet of one-sided handwritten notes, which you will be asked to attach. Otherwise this exam is closed book, closed phones, closed web, etc.

This exam has 6 problems, some with multiple parts (plus "problems" for the honor pledge and file uploads). Be sure to read/answer all parts of each question!

You have 120 minutes to earn 100 points. Note that there are 102 possible points, but your score is out of 100.

Do not spend too much time on any one problem. Read through all of them first, and then attack them in the order that allows you to make the most progress.

Show your work---partial credit will be given for meaningful progress towards a correct solution. You will be graded not only on the correctness of your answer, but also on the clarity with which you express it. Be neat.

It is fine if you prefer to type out notation. For example, if you want to denote $\binom{n}{k}$, you can write "n choose k". Just be clear.

Good luck!

Info

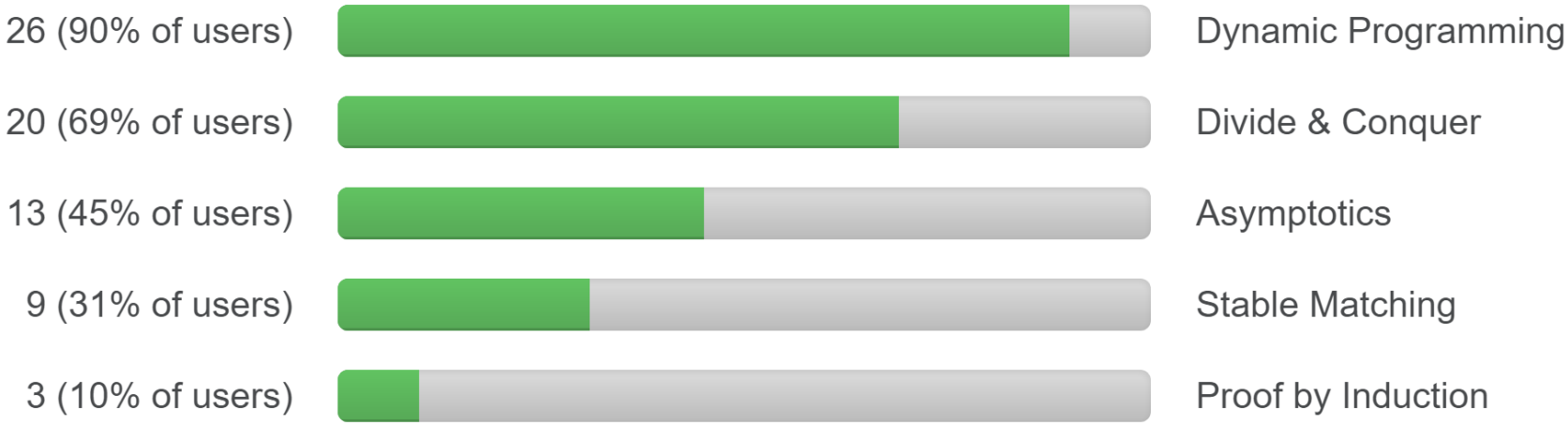
- Recitation is still on for Monday 5/31 (no class – Memorial Day)

- Exam will be on Canvas
  - I released a Sample Exam so you can practice with the interface tomorrow
  - Exam will be open from 12am Eastern on 6/1 to 11:59pm Eastern on 6/1 (so you should start by 10pm)
  - You do not need to use LaTeX
  - I will be "on demand" via MS Teams from 1-310p Tuesday to answer questions
  - You may not receive support from TAs
  - The exam may be curved up, but will not be curved down
  - You will have the option of uploading images/scrap work

More Info

- Anyone found cheating will receive significant penalty (including possibly a 0 for the course)
- Please don't post about or discuss the exam until the end of the week
- Please be sure to sign the honor pledge at the start of the exam
- I am cognizant that exams are stressful, perhaps particularly due to the current state of the world, and would like to support you to the best of my ability. Please reach out if you have questions, concerns, etc.

A total of **29** vote(s) in **23** hours



Practice Question: DP

Problem 2. *Dynamic Programming*

The dark lord Sauron loves to destroy the kingdoms of Middle Earth. But he just can't catch a break, and is always eventually defeated. After a defeat, he requires three epochs to rebuild his strength and once again rise to destroy the kingdoms of Middle Earth. In this problem, you will help Sauron decide in which epochs to rise and destroy the kingdoms of Middle Earth.

The input to the algorithm consists of the numbers $x_1, \ldots, x_n$ representing the number of kingdoms in each epoch. If Sauron rises in epoch i then he will destroy all x_i kingdoms, but will not be able to rise again during epochs $i+1, i+2$, or $i+3$. We call a set $S \subseteq \{1, \ldots, n\}$ of epochs *valid* if it satisfies this constraint that $|i-j| \geq 4$ for all $i, j \in S$, and its *value* is $\sum_{i \in S} x_i$. You will design an algorithm that outputs a valid set of epochs with the maximum possible value.

Example: Suppose there are $(1, 7, 8, 2, 6, 3)$ kingdoms of Middle Earth in epochs $1, \ldots, 6$. Then the optimal set of epochs for Sauron to rise up and destroy the kingdoms of Middle Earth is $S = \{2, 6\}$, during which he destroys 10 kingdoms, 7 in the 2nd epoch and 3 in the 6th epoch.

Using DP… $0 \leq j \leq n \rightarrow OPT(j) =$ # of kingdoms Sauron can destroy in the first j epochs

*** describe the set of subproblems you consider**

*** give a recurrence expressing the solution to each subproblem in terms of the solution to smaller subproblems**

***sketch pseudocode of your algorithm & give the runtime**

***describe how you would recover the solution (epochs) if asked**

DP Practice

Recurrence:

$$OPT(j) = \max\left(x_j + OPT(j-4),\ OPT(j-1) \right)$$

epoch j is soln.

epoch j is not in soln.

$$OPT(1) = x_1$$

$$OPT(-3) = OPT(-2) = OPT(-1) = OPT(0) = 0$$

$$OPT(2) = \max(x_1, x_2)$$

$$OPT(3) = \max(x_1, x_2, x_3)$$

DP Practice

R/T: $O(n)$

FindOPT($x_1, \ldots, x_n$, n)

Let M be an empty array of size $n+1$

$M[0] = 0$, $M[1] = x_1$, $M[2] = \max(x_1, x_2)$,

$M[3] = \max(x_1, x_2, x_3)$

for j from 4 to n:

  if $M[j-1] > x_j + M[j-4]$

    $M[j] = M[j-1]$

  else

    $M[j] = x_j + M[j-4]$

FindOPT(n)

  $M \leftarrow$ empty array of size $n+1$

  Base cases

  if $M[n]$ is not empty: return $M[n]$

  else:

    $M[n] = \max($FindOPT$(n-1), x_n +$ FindOPT$(n-4))$

    return $M[n]$

DP Practice

Use the filled-in DP table M
Let i = n

Compare M[i] vs. M[i-1]

if M[i] > M[i-1]
// ith epoch is in soln.
// look @ M[n-4]
Repeat w/ i := i-4

else (M[i-1] = M[i])
// ith epoch is not in soln.
// look @ M[i-1]
Repeat w/ i := i-1

Stop once $i \leq 0$

Practice Question: Divide-and-Conquer

You are babysitting your niece and before she will go to bed she insists on playing the following guessing game:

1. She picks a number x in $1, 2, \ldots, n$.
2. You make a guess y_1, and she simply says *correct* or *incorrect*.
3. You make a sequence of guesses $y_2, y_3, \ldots$. If your guess $y_i = x$ then your neice says *correct* and goes to bed. If your guess y_i is closer to x than the previous guess y_{i-1}, then she says *warmer* and if y_{i+1} is farther than the previous guess, then she says *colder*.

Your goal is to find x with as few guesses as possible so that your niece will go to bed. Design a divide-and-conquer algorithm that guesses your niece's number using $O(\log n)$ guesses.[1]

- Two versions:

  1. If your guess y_i is the same distance as guess y_{i-1}, your niece stays up forever and you lose

  2. If your guess y_i is the same distance as guess y_{i-1}, your niece says "same" and you win

Practice Question: Divide-and-Conquer

- Describe your algorithm in pseudocode

initialize $l = 1$, $r = n$

$y = l + \left\lfloor \frac{r-l}{2} \right\rfloor$,

Guess y

if correct → we win

Guess $y+1$

if correct → we win

if warmer // x is in right half

$l = l + \left\lfloor \frac{r-l}{2} \right\rfloor + 2$

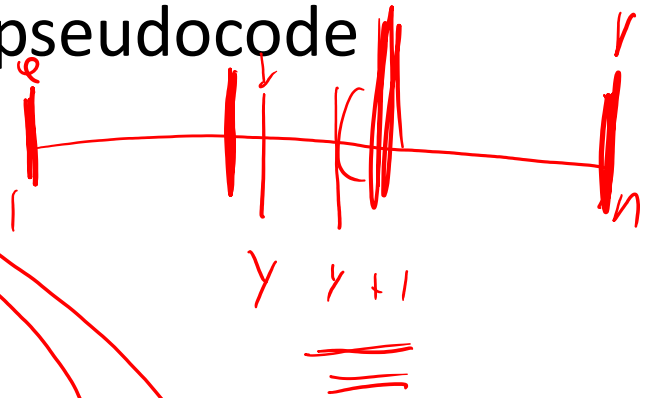
$r = r$

Guess again w/ new l and r

if colder // x is in left half:

$l = l$

$r = l + \left\lfloor \frac{r-l}{2} \right\rfloor - 1$ Guess again



Practice Question: Divide-and-Conquer

- Prove by induction that the algorithm is correct

Practice Question: Divide-and-Conquer

- Analyze your algorithm's running time by writing a recurrence

DP Practice

Longest Increasing Subsequence (LIS): Given a sequence of numbers, find the length of the longest subsequence such that the elements are in increasing order.

Ex. Input: 10, 22, 9, 33, 21, 50, 41, 60, 80, 47

Sol. LIS has length 6 (10, 22, 33, 50, 60, 80)

Let OPT(j) be the length of the LIS ending at the j-th number *(0 <= j <= n)*

Using DP…

* give a recurrence expressing the solution to each subproblem in terms of the solution to smaller subproblems

*sketch pseudocode of your algorithm & give the runtime

*describe how you would recover the LIS if asked

DP Practice

DP Practice

Practice Question: Asymptotics

- Suppose $f_1(n) = O(g(n))$ and $f_2 = O(g(n))$. Prove that $f_1(n) + 4f_2(n) = O(g(n))$.

Practice Question: Asymptotics

- Put these functions in order so that $f_i = O(f_{i+1})$
  - $n^{\log_2 7}$
  - $8^{\log_2 n}$
  - $2^{3.1 \log_2 n}$
  - $2^{(\log_2 n)^2}$
  - $n^2 \sum_{i=1}^{n} i$
  - $n^2 \log_2 n$

(See HW1 for more examples)