CS3000

May 18th - Thurs :)

Admin

- Short HW2 due 9pm
- Long HW2 out, due 5/23 9pm
- optional rec today (3:20 on zoom)
- 5/23 last day to submit rec 1-2

Agenda

1. LCS goal solution (length + what)
2. LCS implementation
3. Exam summary

Recap

- optimization problem ~ many valid solutions, pick the best
- LCS of X, Y — sequence of max. length subsequence of X, subsequence of Y
- LCS of . . .

CBA, LIBR — B (length 1)

CBA, LIBRARY — BA (length 2)

CBAWY, LIBRARY — BAY (length 3)

CBA, ABC — A, B (length 1)

1. LCS Goal Solution

- exhibits optimal substructure
  - ↳ solution to smaller problem is contained in solution to bigger problem
  - ↳ once we've solved a smaller version, we can use it in bigger problem

\/\

Solution can be bottom-up!

/\

Goal solution: LCS-Length

$X = \langle A, B, C, D \rangle$

$Y = \langle A, E, B, D, H \rangle$

Defined c table

Y is a sequence
↳ $\langle y_1, y_2, \ldots, y_n \rangle$

Y_i is a subsequence
↳ $\langle y_1, y_2 \ldots y_i \rangle$

y_i is an element

$C[i,j]$ = length of LCS of x_i, Y_j

	y_j	A	E	B	D	H
x_i	0	0	0	0	0	0
A	0	1	1	1	1	1
B	0	1	1	2	2	2
C	0	1	1	2	2	2
D	0	1	1	2	3	3

$C[0,0]$ = empty, empty

$C[1,2]$ = A, AE

$C[3,4]$ = ABC, AEBD

$C[0,5]$ = empty, AEBDH

Recursive Formula

$$C[i,j] = \begin{cases} 0 & \text{if } i=0 \text{ or } j=0 \quad \text{(base case)} \\ C[i-1,j-1]+1 & \text{if } x_i = y_j \\ \max(C[i-1,j], C[i,j-1]) & \text{if } x_i \neq y_j \end{cases}$$

Space of c table is $\Theta(m \cdot n)$

- so, if we can fill it up by visiting each cell once, we'll a run-time of $\Theta(m \cdot n)$

We want to create:

- c table: length of LCS
- b table: how to re-create an LCS
  - ↳ Filled with arrows

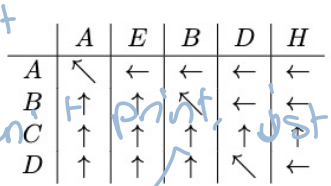
↖ go to $c[i-1, j-1]$ (x_i, y_i are same!)

← go to $[i, j-1]$ ($x_i \neq y_j$)

↑ go to $[i-1, j]$ ($x_i \neq y_j$)

Create the LCS once we have b table

- start in bottom-right
- see ↖, that element in LCS, so print and go up/left
- see ↑, ← don't print, just move



	A	E	B	D	H
A	↖	←	←	←	←
B	↑	↑	↖	←	←
C	↑	↑	↑	↑	↑
D	↑	↑	↑	↖	←

B table

start

- follow the arrow

#2

- D is in LCS
- follow the arrow

LCS: A B D

- #3 (↑)
  - follow the arrow
- #4 (↖)
  - B is in LCS
  - follow the arrow
- #5 (←)
  - follow the arrow
- #6 (↑)
  - A is in LCS

Space complexity:

- $\Theta(mn)$ for c table
- $\Theta(mn)$ for b table

} altogether $\Theta(mn)$

2. LCS Implementation

- c table: length of LCS
- b table: what is the LCS

- if we can fill up the c table by visiting each cell once, we have a good runtime

$$\begin{cases} 0 & \text{if } i=0 \text{ or } j=0 \quad \text{(base case)} \\ c[i-1, j-1] + 1 & \text{if } x_i = y_j \\ \max(c[i-1, j], c[i, j-1]) & \text{if } x_i \neq y_j \end{cases}$$

Recursive formula

↳ implementation

Bottom up

just focus on c table

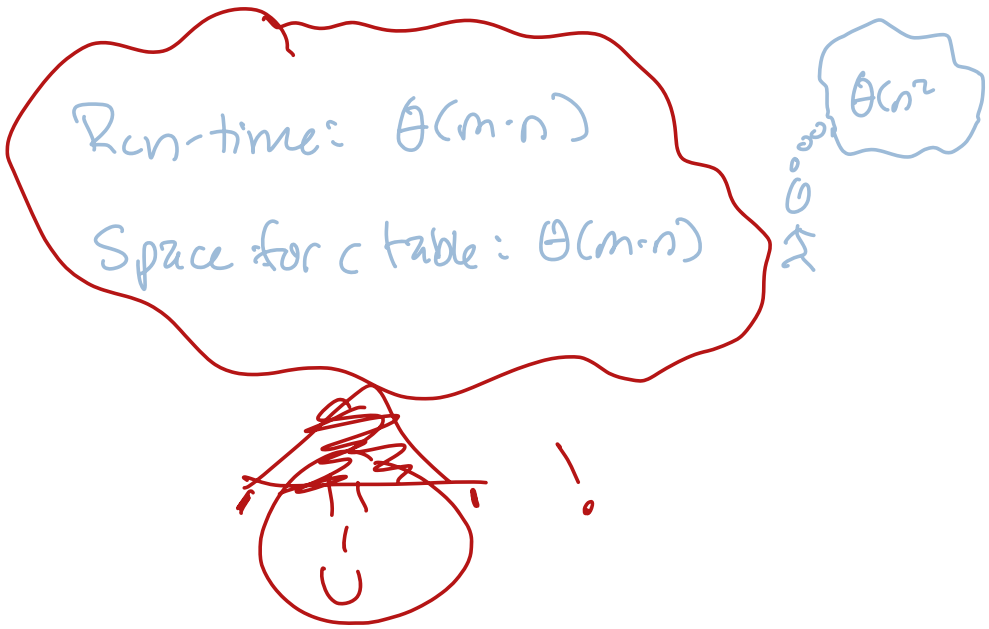
```
LCS-Length (X, Y, m, n)        // m: length of X, n: length of Y
  let c[0:m, 0:n] be a new table        ⇝ space allocation
  for i = 1 to m                  } base case
      c[i, 0] = 0
  for j = 0 to n                  } base case
      c[0, j] = 0
  for i = 1 to m
      for j = 1 to n
          if x_i == y_j
              c[i, j] = c[i-1, j-1] + 1
          else
              if c[i-1, j] > c[i, j-1]
                  c[i, j] = c[i-1, j]
```

10:47

else

$$c[i,j] = c[i,j-1]$$

return c



3. Exam Summary

- exam #1 5/25 (thurs)
- exam #2 6/15 (thurs)
- make up exam question 6/22

Day of exam

- in class
- on paper
- extra scratch paper
- DRC, schedule now
- ~90 minutes
- 8.5x11 inch cheat sheet (one side)

Whats on exam

- up through DP
- given an algo, formally state its run time
  - state a recurrence
  - solve the recurrence
- state bound on run time
- Given a recurrence formula, fill in the table

- Prove correctness: loop invariant
  by induction
- given a DP table, what is answer
- given recursive formula, write pseudocode
- come up with pseudocode for an algorithm
- compare run times
- run times of known algos

Preparation

- practice problems (Sat)
- next tues. recitation - exam prep
- next weds - class prep