Dynamic Programming

- 2xn domino tiling
- log cutting
- matrix chaining
- longest common subsequence
- seam carving
- roller coaster (HW5)
- gerrymandering

- Requires Optimal Substructure
  - optimal solution to the larger problem contains optimal solutions to the smaller ~~subproblems~~ versions of this problem
- 3 steps:
  1. identify recursive structure
     "what is the last thing we did?"
  2. Save solutions to any subproblems in memory (memoization)
  3. Is there a good order to solve our smaller subproblems?
     - top-down
     - bottom-up

Domino Tiling (2xn)

Tile(n) = number of way to tile a board of size 2xn

$$\text{Tile}(n) = \text{Tile}(n-1) + \text{Tile}(n-2)$$

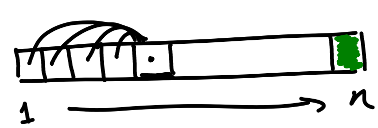
anytime we compute Tile(i) we will store in memory

Log Cutting

Cut(i) = ~~how~~ the most much money I can make by cutting a log of size i.

$$\text{cut}(i) = \max \begin{cases} \text{cut}(i-1) + P[1] \\ \text{cut}(i-2) + P[2] \\ \text{cut}(i-3) + P[3] \\ \vdots \\ \text{cut}(0) + P[i] \end{cases}$$

anytime we compute cut(i) we will store in memory

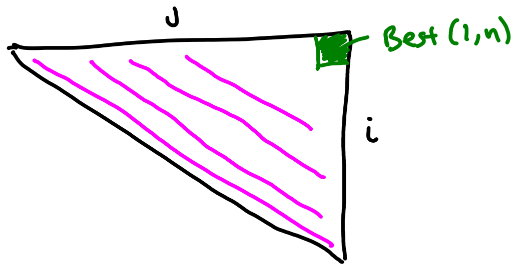


Matrix Chaining

Best(i,j) = fewest (best) number of operations to multiply matrices $i \cdots j$

$$= \min_{k=i}^{j-1} \left(\text{Best}(i,k) + \text{Best}(k+1,j) + r_i r_{k+1} c_j \right)$$

Any time we compute Best(i,j), store in memory.



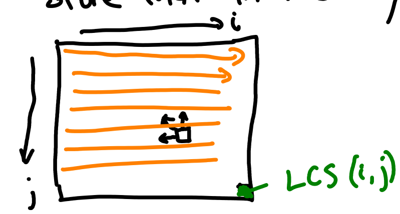
Longest Common Subsequence

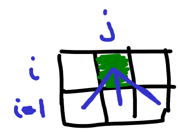
given strings X, Y

LCS(i,j) = length of the longest common subsequence among first i chars of X and j chars of Y.

$$\text{LCS}(i,j) = \begin{cases} 0 & \text{if } i=j=0 \\ \text{LCS}(i-1,j-1)+1 & \text{if } X[i]=Y[j] \\ \max\begin{cases}\text{LCS}(i,j-1) \\ \text{LCS}(i-1,j)\end{cases} & \text{if } X[i] \neq Y[j] \end{cases}$$

Anytime we compute LCS(i,j), Store that in memory



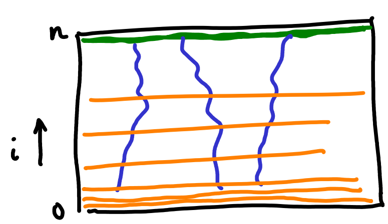


Seam Carving

Seam(i,j) = weight of least weight seam starting at the bottom of the image and ended at pixel $[i,j]$

$$\text{Seam}(i,j) = \text{energy}(\text{pixel}[i,j]) + \min \begin{cases} \text{Seam}(i-1,j-1) \\ \text{Seam}(i-1,j) \\ \text{Seam}(i-1,j+1) \end{cases}$$

Anytime we compute Seam(i,j), we should store in memory.



best seam overall = $\min_k (\text{Seam}(n,k))$

Roller coaster

longest decreasing path

- length of longest decreasing path