CS 223 Lec 12

Dynamic Programming

- solving optimization problems

↓

find a solution that maximizes an objective fn

OR

minimizes a cost fn →

Steps to finding a dynamic programming algorithm

1. Characterize the structure of an optimal soln.

2. recursively define the value of an optimal soln

3. compute the value of an optimal soln using recurrence eqn above.

4. construct an optimal soln from computed info in step 3.

Example.

Suppose you want to build a tower of height h.

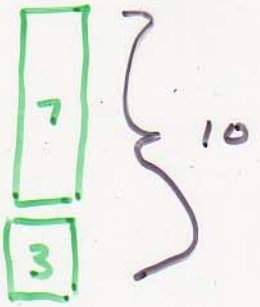
Given a set of block lengths $\{ l_1, l_2, \ldots l_k \}$ that can be produced.

Objective function:

Minimize the # of blocks needed to build a tower of specified height.

ex1 $h = 10$

$l_1 = 2$, $l_2 = 7$, $l_3 = 3$

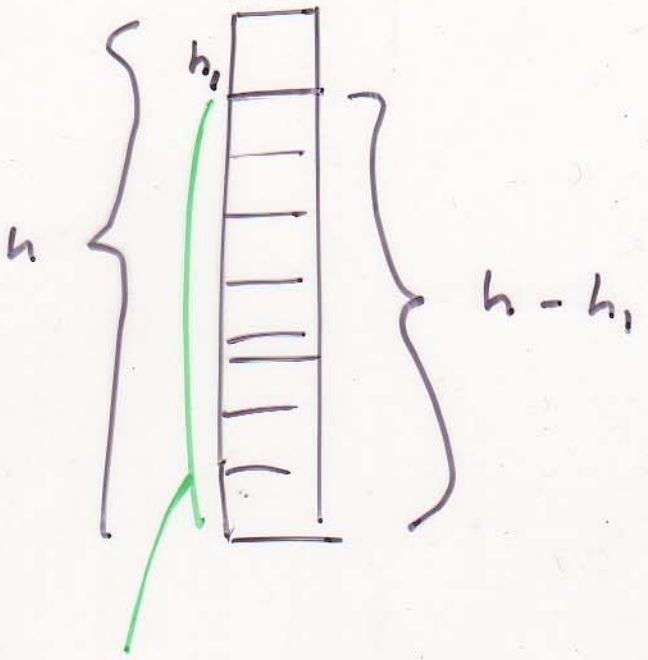


ex2 $h = 30$

$l_1 = 1$, $l_2 = 10$, $l_3 = 25$

Note greedy strategy doesn't work.

optimal soln



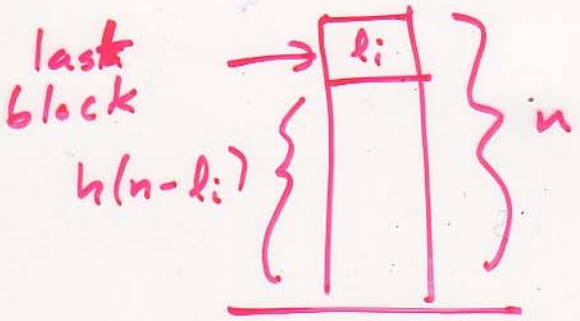
Claim: must also be optimal

Optimal soln for height h then removing the last block gives an optimal soln of height $h - h_1$

Define $h(n)$ = min # of blocks needed to form a tower of height n.

... next find a recursive relationship for $h(n)$

$$h(n) = \min \begin{cases} h(n-\ell_1) \\ h(n-\ell_2) \\ \vdots \\ h(n-\ell_k) \end{cases} + 1$$



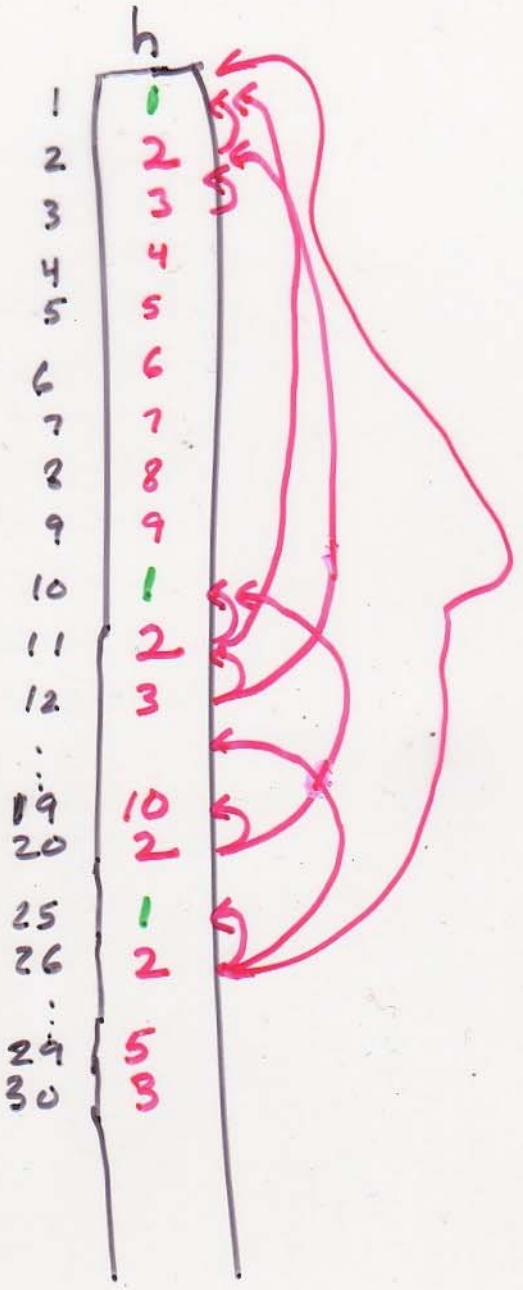
Next step: compute $h(n)$

Base cases: towers of one block only

heights = $\{\ell_1, \ell_2, \ldots \ell_k\}$.

$h(\ell_1) = h(\ell_2) = \ldots = h(\ell_k) = 1$

$l_1 = 1,\ l_2 = 10,\ l_3 = 25$



Construct an optimal soln from the optimal values.