i	1	2	3	4	5	6	7	8	9	10
p_i	1	3	5	8	8	10	10	15	18	18

i	1	2	3	4	5	6	7	8	9	10
r_i	1	3	5	8	9	11	13	16	18	19

cuts: 1, 9

also possible: 2, 4, 4

Finish Matrix chain order:

Recall input $\langle p_0, \ldots, p_n \rangle$ positive integers

For $1 \le i \le j \le n$:

$$m[i,j] = \min_{k=i}^{j-1} \left[m[i,k] + m[k+1,j] + p_{i-1} p_k p_j \right]$$

Split $A_{1..n} = \underline{A_{1..k} A_{k+1..n}}$ optimally,

$$m\boxed{[1,n]} = m[1,k] + m[k+1,n] + p_0 p_k p_n$$

$i=1$
$j=n$

for some $k \in \{1, \ldots, n-1\}$

record the value of k in another table (s-table) for each $i<j$

Only using $m[i,j]$ for $i \le j$.

Store optimal k for splitting $A_{i..j}$ in $m\boxed{[j,i]}$ (for all $i<j$) $i \le k < j$

Longest Common Subsequence

Input: two strings

$\vec{x} = x_1 x_2 \cdots x_m$

$\vec{y} = y_1 y_2 \cdots y_n$

of symbols over some alphabet (e.g., $\{0,1\}$, $\{A,G,C,T\}$, ..)

Find the largest length of any common subsequence of $\vec{x}$ & $\vec{y}$. [and an actual longest common subsequence (LCS).]

A subsequence of a sequence $x_1 x_2 \cdots x_m$ is one obtained by removing 0 or more symbols (and closing up the gaps)

$x_3 x_4 x_7 x_{11}$

Optimal substructure?

Yes.

$x = x_1 \cdots x_m$

$y = y_1 \cdots y_n$

Easy: $m=0$ or $n=0$

LCS length $=0$

Else: $m>0$ & $n>0$

$x = x_1 \cdots x_{m-1}\ x_m$

$y = y_1 \cdots y_{n-1}\ y_n$

Case 1: $x_m = y_n$

For any subsequence of x & y there is a subsequence at least as long that uses $x_m \leftrightarrow y_n$

$\therefore$ There is an LCS that match x_m with y_n, ~~and~~

preceded by an LCS of $x_1 \cdots x_{m-1}$ & $y_1 \cdots y_{n-1}$

Case 2: $x_m \ne y_n$

Then no common subsequence can match x_m with y_n as its last symbol.

Then the length of an LCS of x & y is either the length of an LCS of $x_1 \cdots x_{m-1}$ & y or an LCS of x and $y_1 \cdots y_{n-1}$, whichever is longer.

Table $L[0..m, 0..n]$

where $L[i,j]$ = length of an LCS of $x_1 \cdots x_i$ & $y_1 \cdots y_j$

Oct 18-12:50 PM

For $j = 0$ to n do
  $L[0, j] := 0$
end-for
For $i = 1$ to m do
  $L[i, 0] := 0$

  For $j := 1$ to n do
  // Fill $L[i, j]$
    if $x_i == y_j$ then
      $L[i,j] := \underbrace{L[i-1, j-1]}_{\text{LCS of truncated strings}} + 1$

      fin… mat…

    else // $x_i \neq y_j$
      $L[i,j] := \max(L[i-1, j], L[i, j-1]$…

  end-for j
end-for i
return $L[m, n]$

$x = 001110101$
$y = 11001$

L	x		$y=$ 0	1 (1)	1 (2)	0 (3)	0 (4)	1 (5)
		0	0	0	0	0	0	0
	0	1	0	0	0	1	1	1
	0	2	0	0	0	1	2	2
	1	3	0					
	1	4	0					
	1	5	0					
	0	6	0					
	1	7	0					