Longest Common Subsequence

Input: two strings, u and v
Output: a common substring
Objective: maximize length of the substring

Example: carrot
party

\[ S[j,k] = \text{max length of a common substring of } u[1...j] \text{ and } v[1...k] \]
Longest Common Subsequence

Input: two strings, u and v
Output: a common substring
Objective: maximize length of the substring

Example:
carrot
party

The heart of the solution:

\[
S[j, k] = \begin{cases} 
0 & \text{if } k = 0 \text{ or } j = 0 \\
\max \{ S[j-1, k], S[j, k-1] \} & \text{if } u_j \neq v_k \\
\max \{ S[j-1, k], S[j, k-1], S[j-1, k-1] + 1 \} & \text{if } u_j = v_k
\end{cases}
\]
Longest Common Subsequence

Input: two strings, u and v
Output: a common substring
Objective: maximize length of the substring

The heart of the solution:

\[ S[j][k] = \]
Longest Common Subsequence

Input: two strings, u and v
Output: a common substring
Objective: maximize length of the substring

The heart of the solution:

\[ S[j][k] = \text{the length of the longest common substring of strings } u[1...j] \text{ and } v[1...k] \]
Longest Common Subsequence

$S[j][k] =$ the length of the longest common substring of strings $u[1...j]$ and $v[1...k]$

**LONGEST-COMMON-SUBSEQUENCE** $(u,v)$

1. init $S[j][k]$ to 0 for every $j=0,...,|u|$ and every $k=0,...,|v|$.
2. for $j=1$ to $|u|$ do
3.   for $k=1$ to $|v|$ do
4.     $S[j][k] = \max\{ S[j-1][k], S[j][k-1] \}$
5.   if $(u[j] = v[k])$ then
6.     $S[j][k] = S[j-1][k-1]+1$
7. RETURN $S[|u|][|v|]$
Matrix Chain Multiplication

Input: a chain of matrices to be multiplied
Output: a parenthesizing of the chain
Objective: minimize number of steps needed for the multiplication

\[
\begin{align*}
A_1 & & A_2 & & A_3 & & A_4 & & A_5 & & A_6 \\
3\times2 & & 2\times7 & & 7\times1 & & 1\times8 & & 8\times3 & & 3\times6 \\
\text{( many possibilities )} & & \text{( exponentially )}
\end{align*}
\]
Matrix Chain Multiplication

Input: a chain of matrices to be multiplied
Output: a parenthesizing of the chain
Objective: minimize number of steps needed for the multiplication

Matrix multiplication:

A of size $m \times r$, B of size $r \times p$

How many steps to compute $A \cdot B$?

$$c_{i,j} = \sum_{k=1}^{r} a_{i,k} \cdot b_{k,j}$$

Cost of multiplying $A \cdot B =$ $m \cdot r \cdot p$ math operations
Matrix Chain Multiplication

Input: a chain of matrices to be multiplied
Output: a parenthesizing of the chain
Objective: minimize number of steps needed for the multiplication

Example:

\[
\begin{array}{ccccccc}
4 & 2 & 5 & 1 & 2 & 3 \\
\end{array}
\]

\[
\begin{array}{c}
(4 \times 2 \quad 2 \times 5) \\
(4 \times 2 \times 5 = 40)
\end{array}
\quad
\begin{array}{c}
(5 \times 1 \quad 1 \times 3) \\
(5 \times 1 \times 3 = 15)
\end{array}
\quad
\begin{array}{c}
40 + 15 + 60 = 121
\end{array}
\]

overall cost of this parenthesizing is 121
Matrix Chain Multiplication

Input: a chain of matrices to be multiplied
Output: a parenthesizing of the chain
Objective: minimize number of steps needed for the multiplication

Heart of the solution:

\[ S[L,R] = \begin{cases} 
\min_{L \leq k \leq R} \{ a_{L-1} \cdot a_k \cdot a_R + S[L,k] + S[k+1,R] \} \\
0 & \text{if } L = R \\
& \text{if } n^2R > L \geq 1 
\end{cases} \]

\[ A_k \text{ is of dimensions } a_{k-1} \times a_k \]
Matrix Chain Multiplication

Input: a chain of matrices to be multiplied
Output: a parenthesizing of the chain
Objective: minimize number of steps needed for the multiplication

Heart of the solution:

\[ S[L,R] = \text{the minimum number of steps required to multiply matrices from the L-th to the R-th} \]
Matrix Chain Multiplication

Running time is $O(n^3)$.

MATRIX-CHAIN-MULTIPLICATION ($a_0, ..., a_n$)

1. for $L=1$ to $n$ do $S[L,L] = 0$
2. for $d=1$ to $n$ do
3. for $L=1$ to $n-d$ do
4. \[ R = L+d \]
5. \[ S[L,R] = \infty \]
6. for $k=L$ to $R-1$ do
7. \[ \text{tmp} = S[L,k]+S[k+1,R]+a_{L-1} \cdot a_k \cdot a_R \]
8. if $S[L,R] > \text{tmp}$ then $S[L,R] = \text{tmp}$
9. return $S[1,n]$