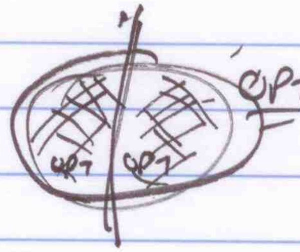
Dynamic Programming

1) Principle of D.P.



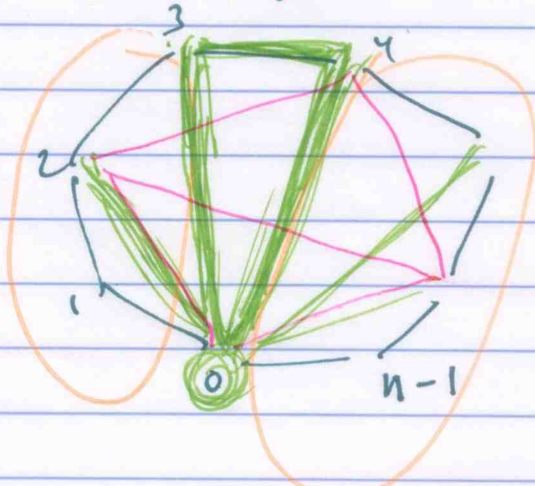
2) Memoization

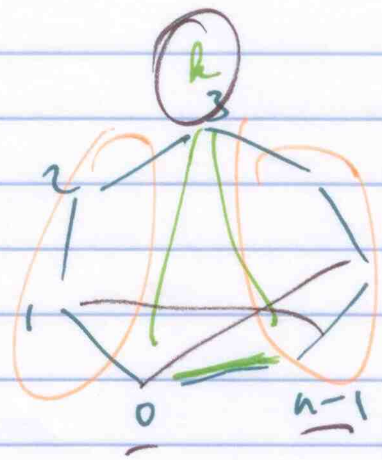
3) Recursive ⟶ Iterative

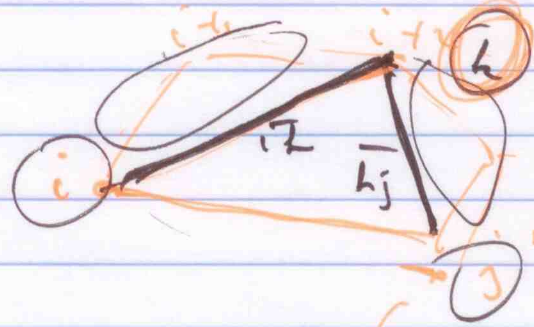
4) Recovery of opt solution

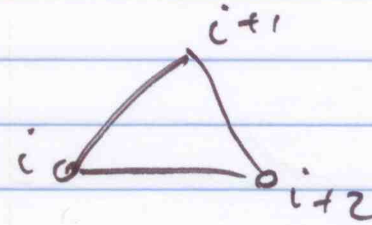
1) Matrix chain product ←

2) Triangulating convex polygons (comp. geometry)







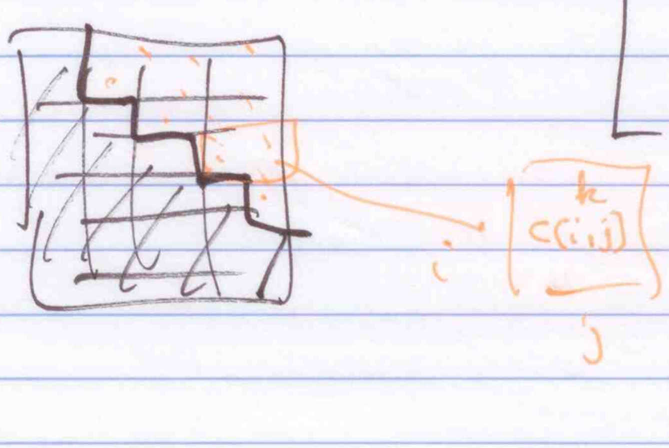


$$\begin{cases} c(i,j) = \min\limits_{i<k<j} \left(c(i,k) + c(k,j) + \overline{ik} + \overline{kj} \right) \\ c(i, i+2 \bmod n) = 0 \end{cases}$$

remember k

$c[i,j]$

if $c[i,j)$ exists return it



Largest Common ~~Substring~~ Sequence

$x_1\ x_2\ x_3 \cdots x_m$

$y_1\ y_2 \cdots y_n$

LCS

$2^m \times 2^n$

BRUTEFORCE

$(n+m)2^{n+m}$

$\cancel{x_1}\ x_2\ x_3$

$\cancel{x_1}\ x_3\ x_5$

L.C.S

$x_1 \cdots x_i$

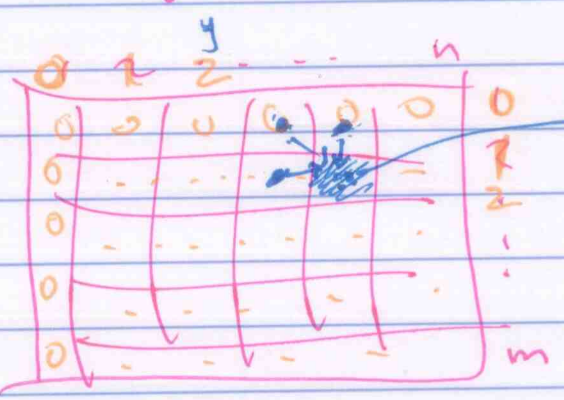
$y_1 \cdots y_j$

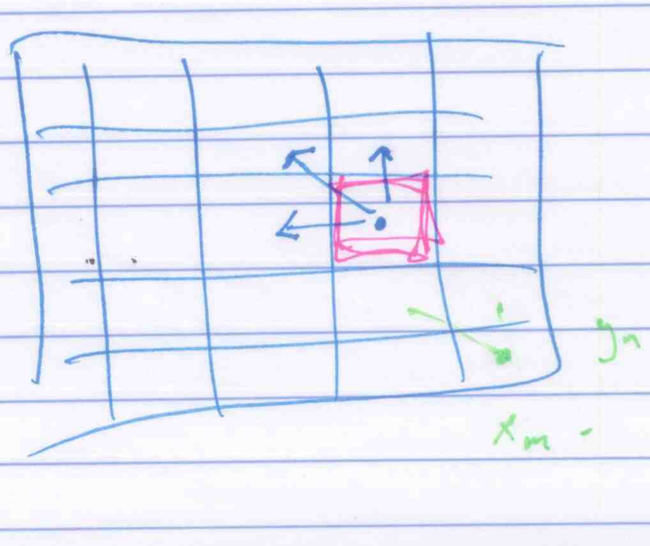
$$\boxed{L_{ij}} = \begin{cases} \boxed{0} & i=0 \text{ or } j=0 \\ L_{i-1,j-1}+1 & x_i = y_j \\ \max\left(L_{i,j-1},\ L_{i-1,j}\right) & \text{o.w.} \end{cases}$$

$\boxed{L[i,j]}$ $\quad 0 \le i \le m$, $\quad 0 \le j \le n$

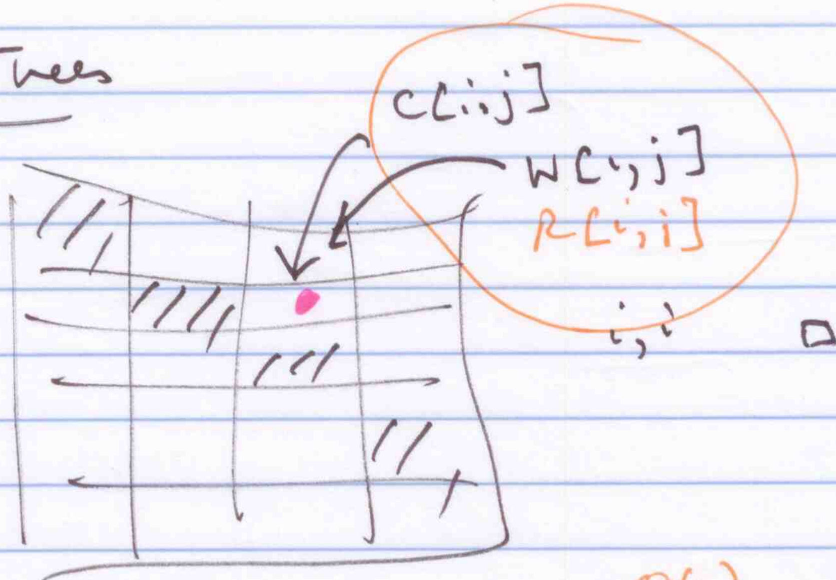
$O(1)$ $\quad \Theta(nm)$

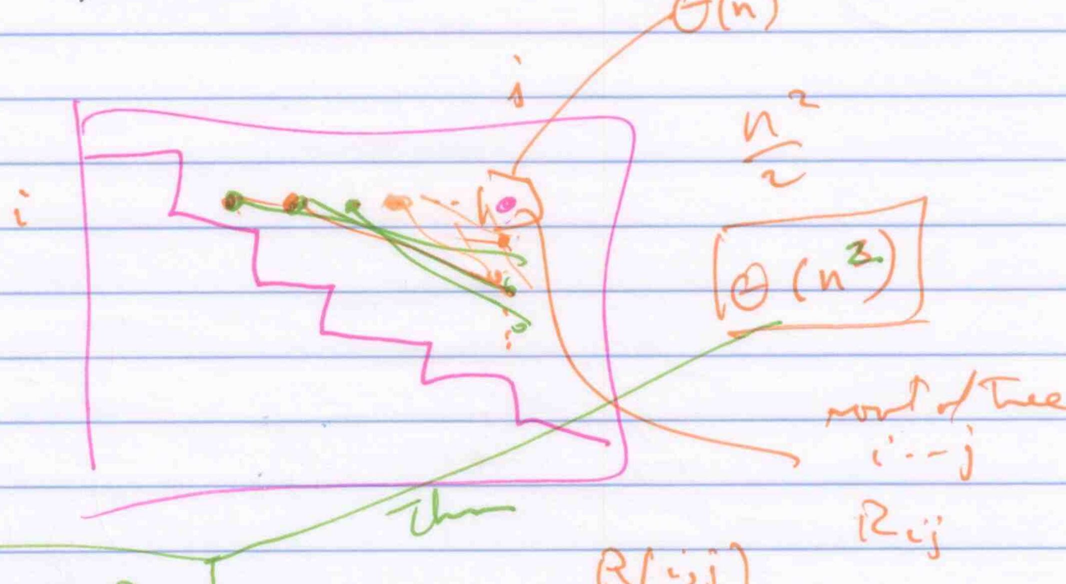
nm





Opt Binary Search Trees





P.S.

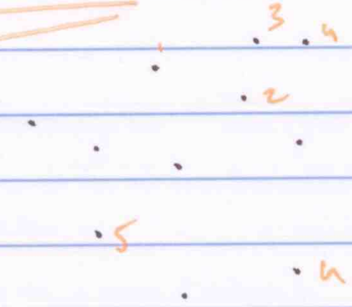
Thm

$$\boxed{R_{i,j-1} \le R_{ij} \le R_{i+1,j}}$$

D.P. in CLRS $\rightarrow$ ~~polynomial~~ time

exponential

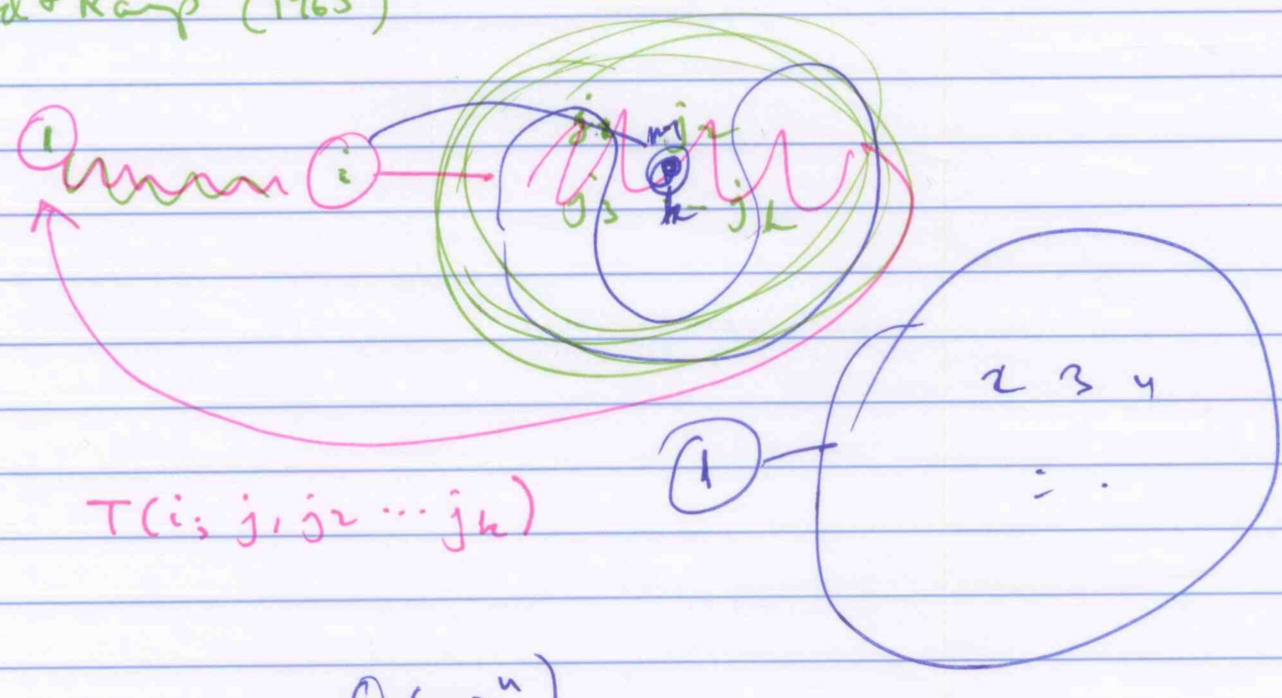
TSP — min route



$\boxed{(n-1)!}$

$$c_1 - \overbrace{c_2 - \cdots c_n}^{n-1}$$

Held + Karp (1965)



$T(i;\ j_1, j_2 \cdots j_k)$

$\Theta(n2^n)$