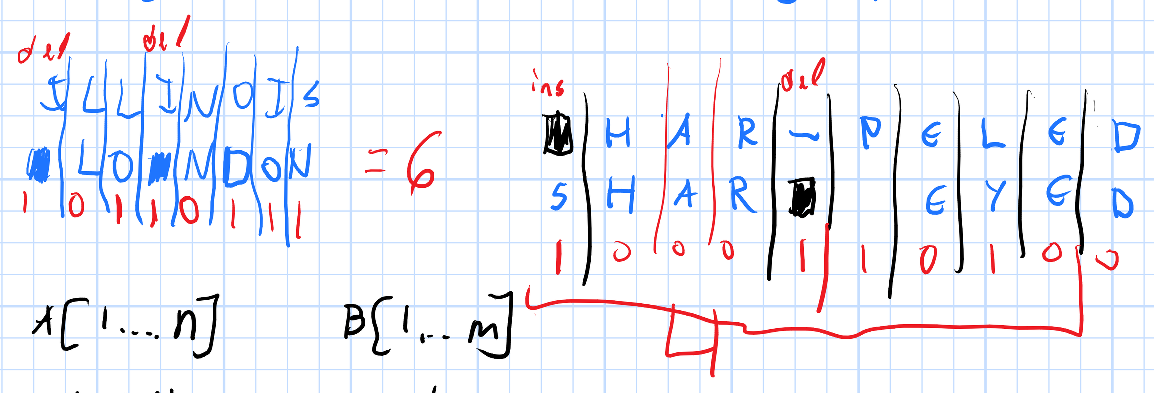
Lecture 6 Dynamic Programming 2 9/7/23

edit distance

ILLINOIS
LONDON

HAR-PELED
SHARP EYED



$A[1 \ldots n]$ $B[1 \ldots m]$

$ed(i,j)$: edit distance $A[1 \ldots i]$ $B[1 \ldots j]$

$$ed(i,j) = \begin{cases} j & i=0 \\ i & j=0 \\ \min \begin{cases} ed(i-1,j-1) + [A[i] \neq B[j]] \\ ed(i,j-1)+1 & \text{deletion} \\ ed(i-1,j)+1 & \text{insertion} \end{cases} \end{cases}$$

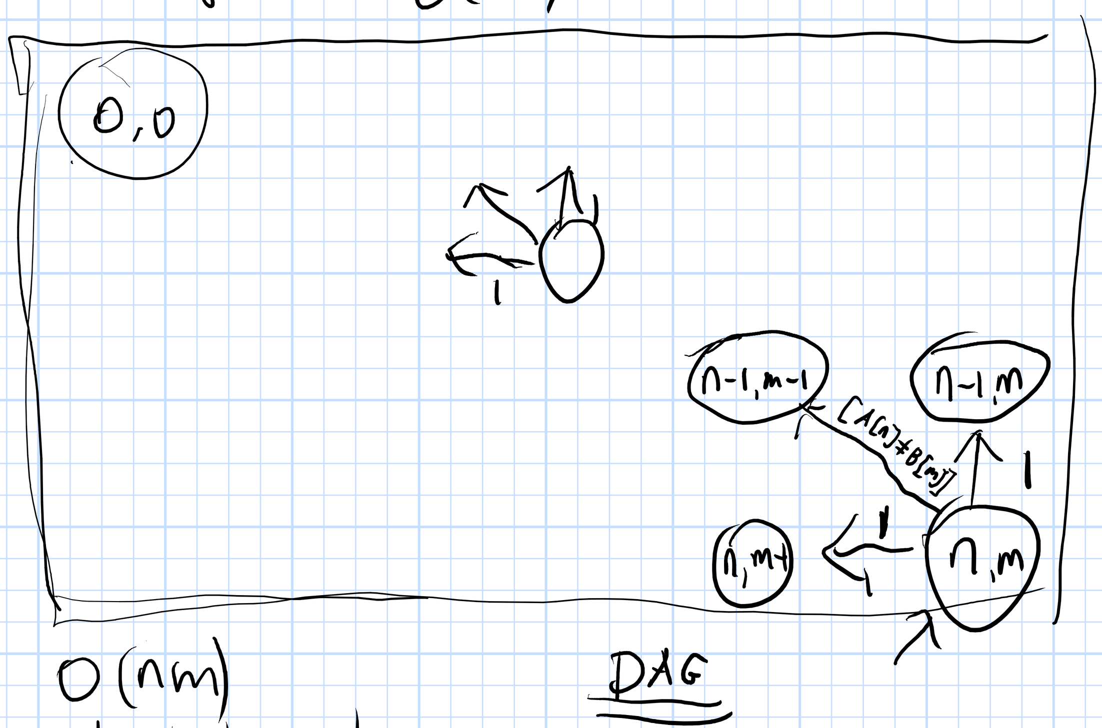
$A[1 \ldots i-1]$ | $A[i]$
$B[1 \ldots j-1]$ | $B[j]$

$A[1 \ldots i-1]$ | $A[i]$
$B[1 \ldots j]$ |

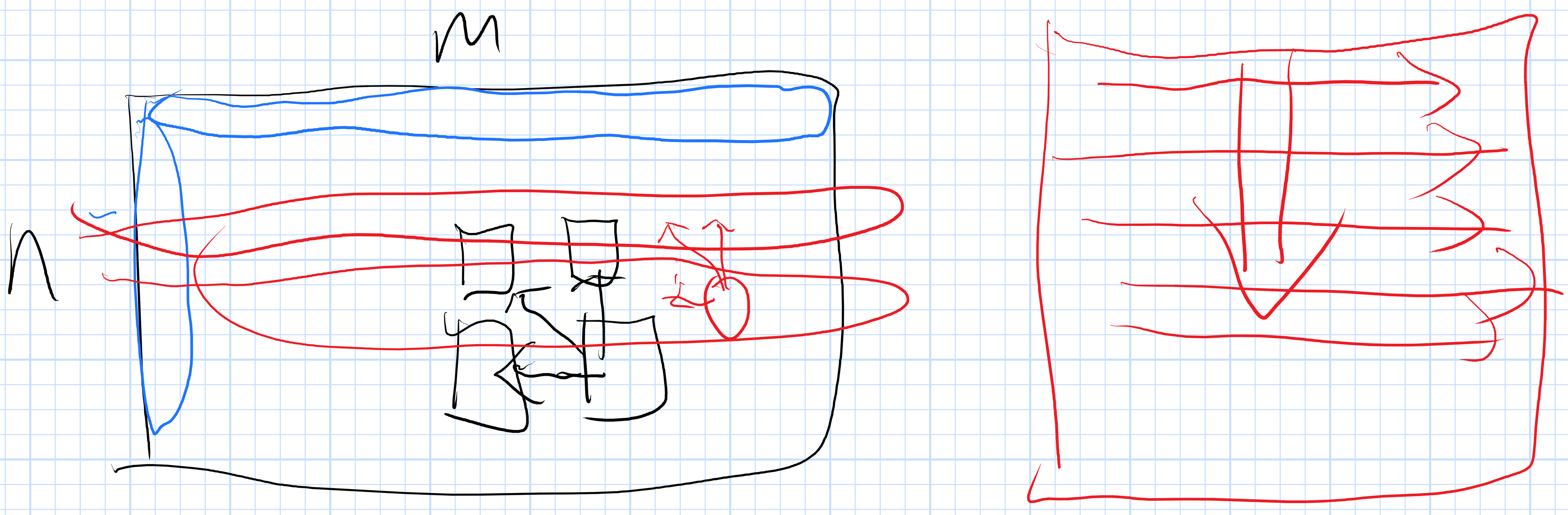
$A[1 \ldots i]$ |
$B[1 \ldots j-1]$ | $B[j]$

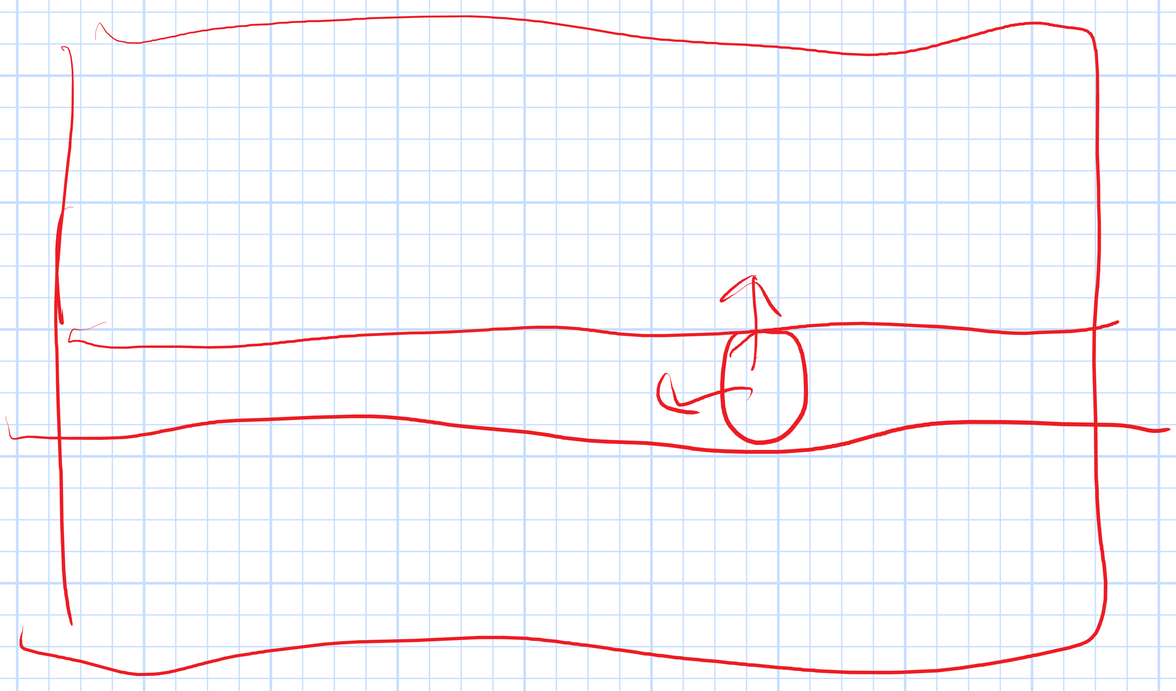
$i \in [0 \ldots n]$
$j \in [0 \ldots m]$ $(n+1)(m+1) = O(nm)$ # distinct recursive calls

ED + Memoization $= O(nm)$ time.



$O(nm)$ shortest path DAG





$O(\min(n,m))$

Exercise

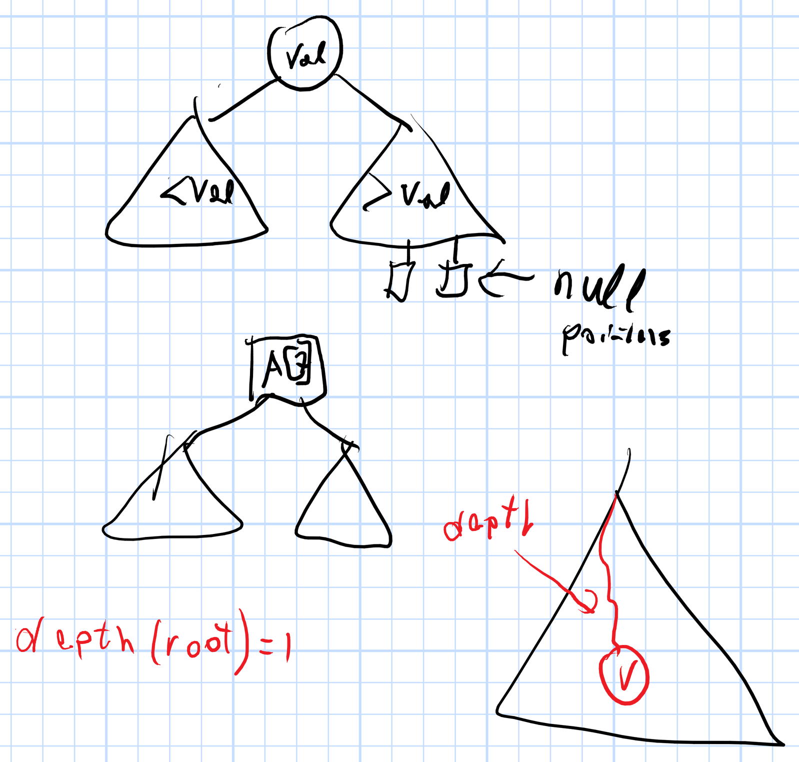
Show how to compute the alignment realizing the min edit distance using $O(n+m)$ space.

$O(nm \log(nm))$ Running time

optimal search trees

$A[1 \ldots n]$ values sorted

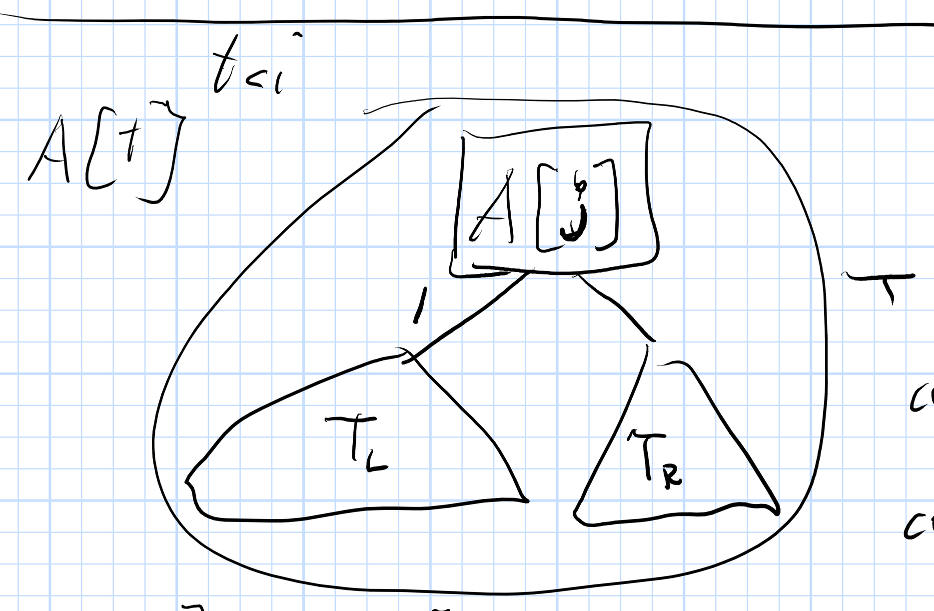
$F[1 \ldots n]$: positive integer numbers.



cost of a tree T

$$cost(F[1 \ldots n]) = \sum_{i=1}^{n} depth_T(A[i]) \cdot F[i]$$

$t < i$



$cost(T) =$

$$cost(T, 1 \ldots n) = \sum_{i=1}^{n} f[i] + cost(T_L, A[1 \ldots j-1]) + cost(T_R, A[j+1 \ldots n])$$

$A[i \ldots j]$ $F[i \ldots j]$

$$c(i,j) = \begin{cases} 0 & i > j \\ F[i] & i = j \\ \min_{k=i}^{j} \Big(\underbrace{\sum_{\ell=i}^{j} f[\ell]}_{O(1)} + c(i,k-1) + c(k+1,j) \Big) \end{cases}$$

$c(1,n)$ # distinct calls $O(n^2)$

$O(n)$ time per entry

$O(n^2 \cdot n) = O(n^3)$