Dynamic Programming

- LIS
- TEXT SEGMENTATION
- SUBSET SUM
- EDIT DISTANCE.

Text Segmentation — I. A MAN

IAMAN

I AM AN

IAMASTUDENT

I AM A STUDENT

ISWORD( ) Yes/No

IAM

AM

A[1 . . . n]

Splittable(i) = True if A[i . . . n]
can be split into

i ... n   meaningful words

$$Splittable(i) = \begin{cases} \text{TRUE} & \text{if } i > n \\ \displaystyle\bigvee_{j=i}^{n} \big(IsWord(i,j) \wedge Splittable(j+1)\big) & \text{otherwise} \end{cases}$$

OR

A[i ... n]

IsWord(i, j) = Yes if A[i..j] is a word;

Using mathematical induction, one can prove it (Exc.)

Splittable(1) = Ans

$(IsWord(1,i) \wedge Splittable(i+1))$

$\vee$ (Isword(i, i+1) $\wedge$ splitable(i+1))

$\vee$ ...

Exc: Prove the above recursive formula is correct. (mathematical induction).

A[i ... n].

Split(i)

return true if i > n

For j = i to n.

If Isword(i, j) and Split(j+1) return true return false;

Worst case running ~~time~~ (the # of calls to IsWord)

$$T(n) = T(n-1) + T(n-2) + \dots + T(1) + 1$$

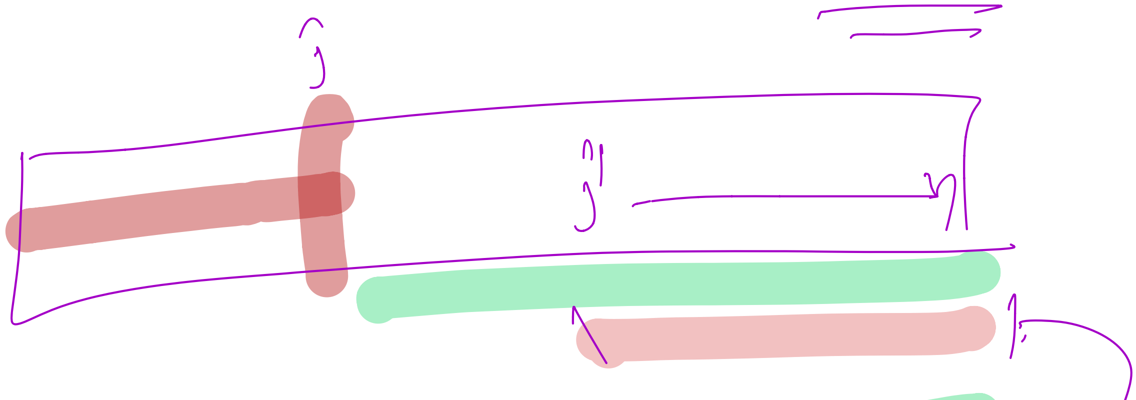
the basic operation is the IsWord $\leq n$

$$T(n) = T(n-1) \dots$$

$$- T(n-1)$$

$$T(n) = 2T(n-1)$$

$$= O(2^n)$$



$j+1$

$$Splittable(i) = \begin{cases} \textsc{True} & \text{if } i > n \\ \displaystyle\bigvee_{j=i}^{n} \big(IsWord(i,j) \wedge Splittable(j+1)\big) & \text{otherwise} \end{cases}$$

① computation order

② Splittable DS to store value.

Splittable(1)

Splittable(1) Splittable(2) … Splittable(n)

Datastructure:
An array of Boolean

n:

A[1 . . n] → S[ ]

```
FastSplittable(A[1..n]):
    SplitTable[n+1] ← True
    for i ← n down to 1
        SplitTable[i] ← False
        for j ← i to n
            if IsWord(i, j) and SplitTable[j+1]
                SplitTable[i] ← True
    return SplitTable[1]
```

True / False

Runtime

$T(n) = O(n^2)$

Space complexi:

$S(n) = O(n).$

SUBSETSUM.

Input: A[1 ... n], an array of n non negative integers and $t \geq 0$.

Output: Yes if there is a subset of the $\{A[1], \ldots, A[n]\}$ that add up to t.

A

5	7	3	2	0	1

$t = 12$

Answer is yes.

$t = 6$, then yes.

$t = 19$ — No.

$A[1 \ldots n]$.

$SS(i, t^*) = $ True if

there is subset

in $\{A[1 \ldots i]\}$

that adds to t^*

$$SS(i, t^*) = \begin{cases} \text{True} & \text{if } t^* = 0 \\ \text{False} & \text{if } i = 0 \text{ and } t^* \neq 0 \\ SS(i-1, t^*) & \text{if } t^* \neq 0 \\ \vee\ SS(i-1, t^* - A[i]) & \end{cases}$$

$SS(n, t)$ — output.

$T(n) = 2T(n-1) + 1$

$= O(2^n)$.

$i = 1 \longrightarrow n$

$t^* = 0 \longrightarrow n = \sum a[i]$

$SS(i, t^*)$

$SS[i, t^*] \quad i \left[\begin{array}{c} \\ t^* \\ \\ \\ n \end{array} \right]$

$SS[n+1][w]$

$SS(a[1 \ldots n])$

{ Base case

For i=1 to n
  for t^* = 0 to W
    $SS[i, t^*] = SS[i-1, t^*] \vee SS[i-1, t^* - A[i]]$

Running time = $O(nW)$

Space Complexity = $O(nW)$

$\leq W$

Input n numbers

Input n log W

$n * 2^{\log w} = w$

max absolute value $\leq W$
and # number is n
$O(nW)$

+ pseudo polynomial time

+ Polynomial time

* running is polynomial in the input length.