

## divide and conquer

Ans; one)

۳) مجموعه ای از افرادی که در یک همانندی مشترک باشند.

## پذیر ادار متعارج

وادر) بـ(دـركـمعـناـضـيـةـ) اـسـارـاـنـجـيـهـ

میری اور حاصل فنر ادارہ وادار

Carry

لر ایم دو و هفتم فریب کشم - فقہا در فنون ثابت فرمایه کردند (۴) ،

رسی خنجر ک ملک (عاصی) ایشانو ترین همراه است → خدا را علیه ایشان می پنداشند و معنی آنها

• M Goss NL  $\rightarrow$  Saw!  $\rightarrow$  N

لـ كـ جـ مـ لـ

Can we do better?

perhaps the most important principle for the good algorithm designer is to refuse the content?)

(1940)

## Karatsuba

$$a \quad c \quad b \\ x \quad c \quad d$$

$$n = 10^{\frac{n}{2}} a + b$$

$$y = 10^{\frac{n}{2}} c + d$$

$$a \cdot y = 10^{\frac{n}{2}} ac + 10^{\frac{n}{2}} (acd + bc) + bcd$$

$$(10^{\frac{n}{2}})^2$$

نیز  $n$  اور  $c, d$  سے ممکنہ ترین طرز کا

point

$$T(n) = ET\left(\frac{n}{2}\right) + 10^{\frac{n}{2}}$$

Ex:  $a = 15 \times 10^5$

$$15 \times 10^5 \times 10^5$$

$$15 \times 10^5 \times 10^5$$

$$(a+b)(c+d) = ac + bd$$

$$+ (acd + bcd)$$

$$15 \times 10^5 \times 10^5$$

$$15 \times 10^5 \times 10^5 \rightarrow acd + bcd = (a+b)(c+d) - (ac + bd)$$

پہلے کوئی  
درست

$$\underbrace{15 \times 10^5 \times 10^5}_{A}, \underbrace{(a+b)(c+d)}_{B}, bd, ac \text{ میں } ①$$

$$B = A - bd - ac \quad ②$$

$$10^{\frac{n}{2}} ac + 10^{\frac{n}{2}} B + bd \quad ③$$

$$T(n) = ET\left(\frac{n}{2}\right) + 10^{\frac{n}{2}}$$

کوئی دوسرے طرز کا نہیں  
کوئی دوسرے طرز کا نہیں

ایسا کوئی دوسرے طرز کا نہیں  
ایسا کوئی دوسرے طرز کا نہیں

کوئی دوسرے طرز کا نہیں  
کوئی دوسرے طرز کا نہیں

input a two n-digit positive integers m and y  
output the product my

Assumption  $n$  is a power of 2

عمران و زوج همسندر صریح نمودند اگر اینها را می بینید

نیویورک نیویورک

سیار

فقط حولت اثنين (تفعيل) صوارب

تعميم الـ  $\omega$  order 5 (رسور)

if  $n=1$  then

## base case

compute my in one step and return the result

else

$a+b$  & = first and second halves of  $n$

$$c_1 \alpha = \sim$$

compute  $P B = a + b$  and  $q \otimes c + d$

## Using school e-school addition

recursively compute  $a \cdot c = a \cdot c + b \cdot d$

and  $P_{qB} = P_{qf}$

compute  $ad(bc) - ac(bd)$  using

## grade school addition

Compute  $10^n \cdot ac + 10^m \cdot abc \rightarrow bcf$

Using grade-school addition and return the result.

15/12

اگر آئندہ ترین کوئی زمان (نئے) حاصل نہ ہو تو دیر را نہ لے سار

مَوْلَانَ (شِعْرٌ) (أَنَّهُ كَرِيمٌ) وَرَحْمَانٌ (كَرِيمٌ)

وَرَدَتْ بِكَاهْ رَافِدْ كَاهْ

اُپر اور پس پردہ آنکھ کی اونٹیں کوں کھانے

الخطوة الأولى (الخطوة الأولى)  $\rightarrow$  ①

الخطوة الثانية (الخطوة الثانية)  $\rightarrow$  ②

الخطوة الثالثة (الخطوة الثالثة)  $\rightarrow$  ③

الخطوة الرابعة (الخطوة الرابعة)  $\rightarrow$  ④

الخطوة الخامسة (الخطوة الخامسة)

### Merge sort

الخطوة الخامسة (الخطوة الخامسة)  $\rightarrow$  ⑤

input array A of n distinct integers

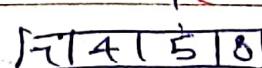
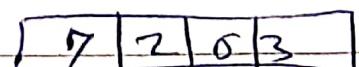
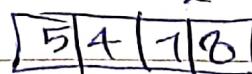
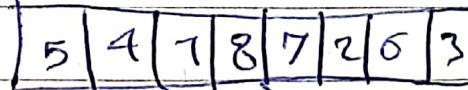
output a array with the same integers sorted from smallest to largest.

mergeSort(A<sub>0..n</sub>)  $\rightarrow$  mergeSort(A<sub>0..n/2</sub>)

(B = recursively sort first half of A

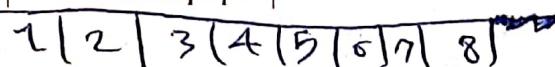
D<sub>B</sub> =  $\sim \sim \sim$  second  $\sim \sim$  A

return merge(C<sub>0..n</sub>)



merge

hirmandpaper



hirmandpaper

الخطوة الخامسة (الخطوة الخامسة)

(P(i)) / D / row

لیست کوچک سینه ای که دو دویس داشت، دویس اول را از اولین  $\leftarrow$  خود  
برداشت کرد و دویس دوم را از پایان داشت و دویس دوم را از پایان داشت

merge

inputs sorted Arrays C and D (length  $\frac{n}{2}$  each)

outputs sorted Array B (length  $n$ )

Simplifying assumption  $n$  is even

all zero

i\_B = 1

first index (i\_B) = R (second A)

$O(n)$  B  $\downarrow$  C  $\downarrow$

j\_D = 1

for K = 1 to n do

if  $C[i] < D[j]$  then

$B[K] = C[i]$

$i = i + 1$

else

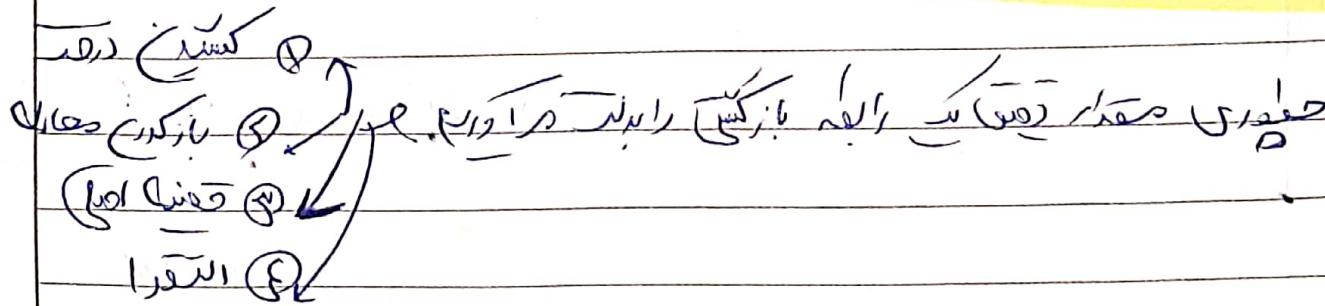
$B[K] = D[j]$

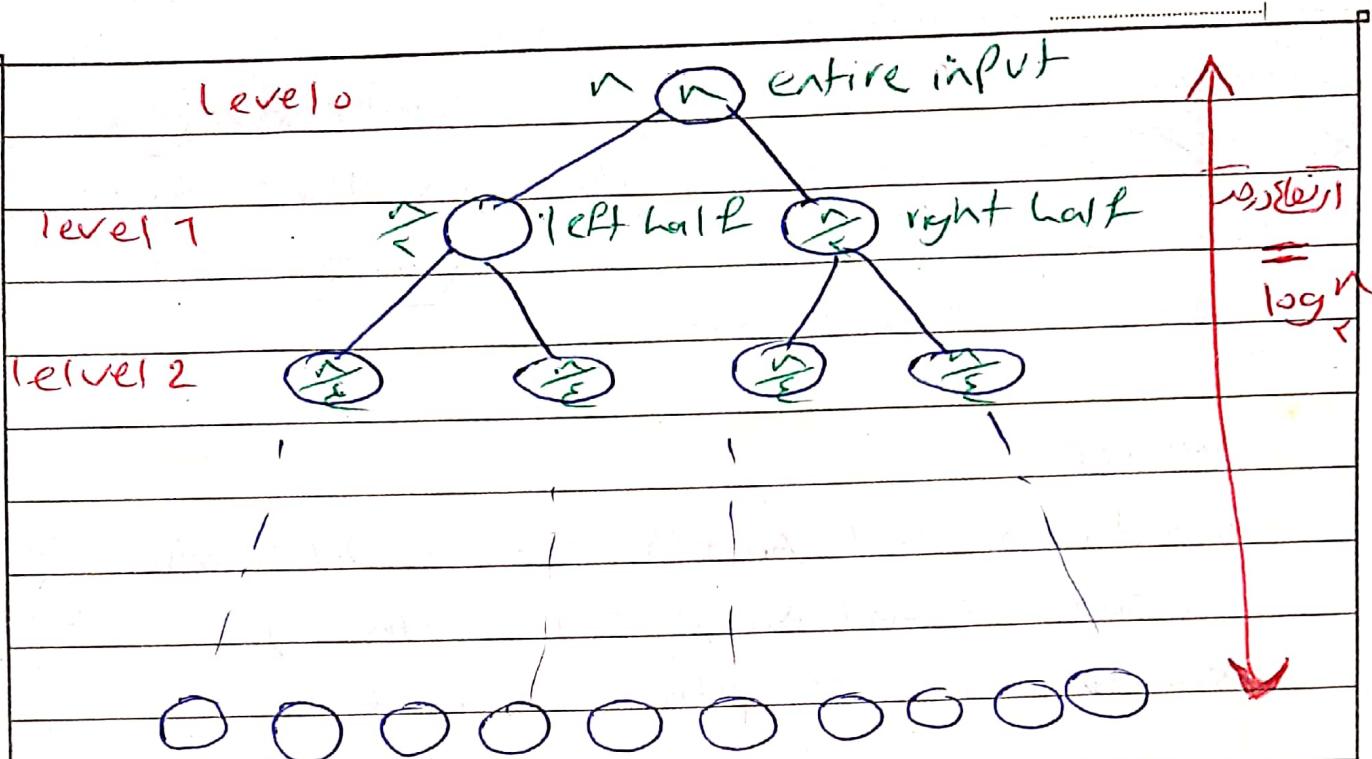
$j = j + 1$

( $i$ )  $\uparrow$  ( $j$ )  $\uparrow$

$T(n) = T(\frac{n}{2}) + O(n)$

mergesort merge



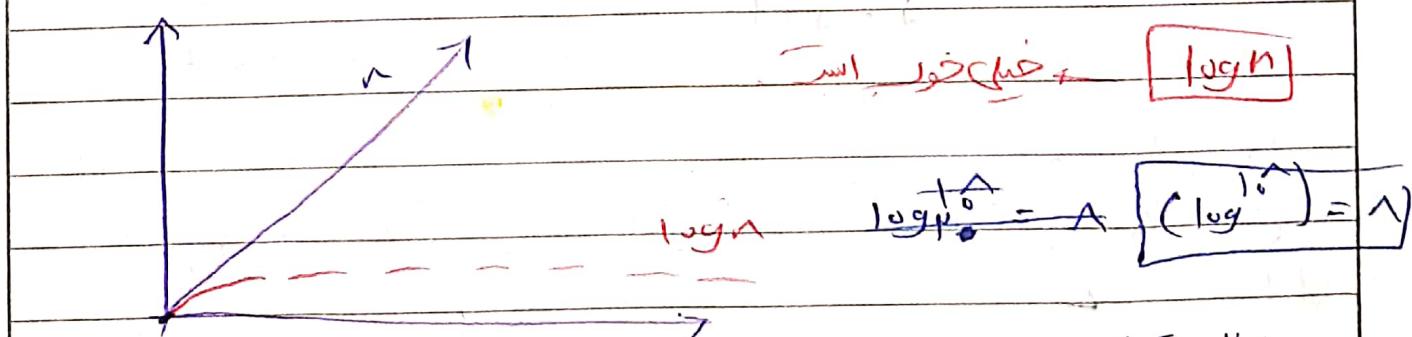


$i^j \in j$  level  $\log n$  time  $\Theta(n)$

$n \in j$  time  $\Theta(n)$

$\approx 10n \times \approx n$  time  $\Theta(n^2)$

$\approx 10n \times \approx n$  time  $\Theta(n^2)$



$$n^{1000} > 10^{10}$$

$$(10n)^{1000} < n^{1000} \times 10^{10}$$

اصل اول بتحصيل التورسم

اصل اول بتحصيل  $\Theta(n^2)$  حاله  $\rightarrow$  کارای زمان اعیا برای هر عددی  
اگر صد عدد زمان  $\Theta(n^2)$  باید آن توسم  $\Theta(n \log n)$  شده، کارایی  $\Theta(n^2)$  که زمان اعیا  
باشد لعدودی  $\Theta(n \log n)$  نباشد  $\Theta(n^2)$

$(O, \Omega, \Theta)$

اصل دوم بتحصيل محابی

صنفه از التورسم نوع  $\Theta$  آن توسم که زمان اعیا آن  $\Theta(n^2)$  حاله  
نسبت به  $n$  باشد و دوستی را کند

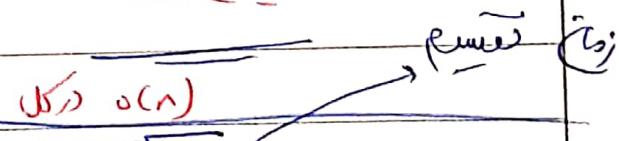
مثال آن توسم سبقتی اول اول بکار برد و دوستی  $\Theta(n^2)$  باشد

$O(\sqrt{n})$  ببراسی با خوبی  $\Theta(n^2)$  باشد

و دوستی  $\Theta(\sqrt{n})$  بکار برد و اس دوستی خوبی خواهیم کرد که آن نشان داد

maximum  $\rightarrow$  سبیل  $\log n$   $\rightarrow$  بکار برد و دوستی  $\Theta(n \log n)$

نمودار  $\Theta(n \log n)$   $\rightarrow$  این توسم



نمودار  $T(n) \leq T\left(\frac{n}{2}\right) + O(n) + O(1)$   $\rightarrow$  تعمیرات

نمودار  $T(n) \leq 2T\left(\frac{n}{2}\right) + O(n)$   $\rightarrow$  بحث در مورد حل زیر مجموعه

Karatsuba

birmandpaper

$3T\left(\frac{n}{2}\right) + O(n)$

ارقام جبار

بعض ادار

## standard Recurrence Form

Base case  $T(n)$  is at most a constant for all sufficiently small  $n$

$$T(n) = O(n)$$

$n < n_0$

General case for large values of  $n$

$$T(n) \leq c_1 T\left(\frac{n}{b}\right) + O(n^d)$$

$n > n_0$

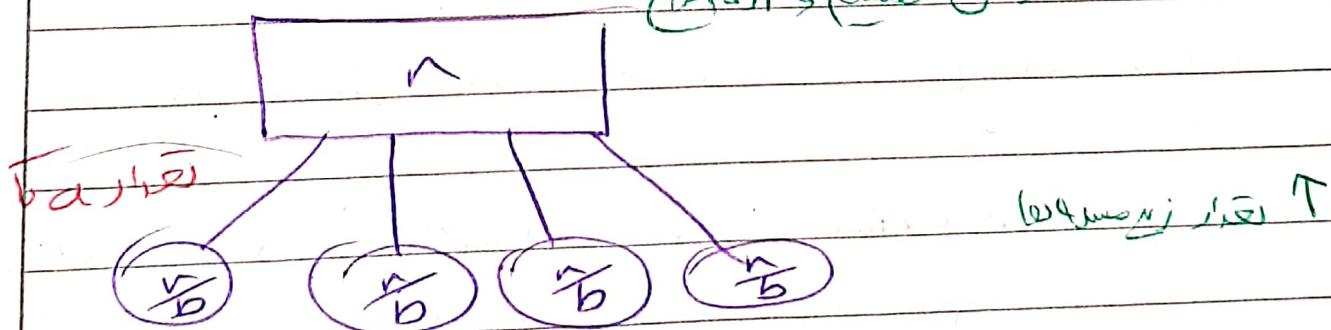
$\alpha$  - number of recursive calls

$b > 1$

$d > 0$

$b$  = input size shrinkage factor

$d$  = exponent in running time of the "combine" step



$$T(n) = c_1 T\left(\frac{n}{b}\right) + O(n)$$

## Master Theorem

$$T(n) \leq O(1)$$

لطفاً ملاحظة

$$T(n) = a T\left(\frac{n}{b}\right) + O(n^d)$$

صيغة كهف

لطفاً ملاحظة

$a > 1, b > 1$

$d > 0$

أمثلة

لطفاً ملاحظة

$$T(n) \leq \begin{cases} O(n^d \log n) & a = b^d \\ O(n^d) & a < b^d \end{cases}$$

$$a = b^d$$

$$a < b^d$$

أمثلة

كذلك

كذلك

كذلك

$$O(n^{\log_b a})$$

$$a > b^d$$

$$(a/b)^d$$

$$\log_b a = \frac{\log a}{\log b} \approx \frac{1}{1}$$

(أمثلة)

$$a = r = b = 1 \rightarrow d = 1, b = 1, a = 1 \rightarrow \text{Case 1}$$

$$T(n) = O(n \log n)$$

Case 1

$$d = 1 \rightarrow b = 1, a = 1 \rightarrow \text{Case 1}$$

$$a = 1 > b^d = 1$$

$$T(n) = O(n^{\log_1 1}) = O(n)$$

Case 3

$d=1$   $b=1$   $a=10^4$  p Karatsuba  $\tilde{O}(n^{1.5})$  (3)

$$a = c > b^d - \epsilon' \quad \xrightarrow{\text{case 3}} \quad T(n) = O(n^{\frac{\log_2 3}{2}}) = O(n^{1.59})$$

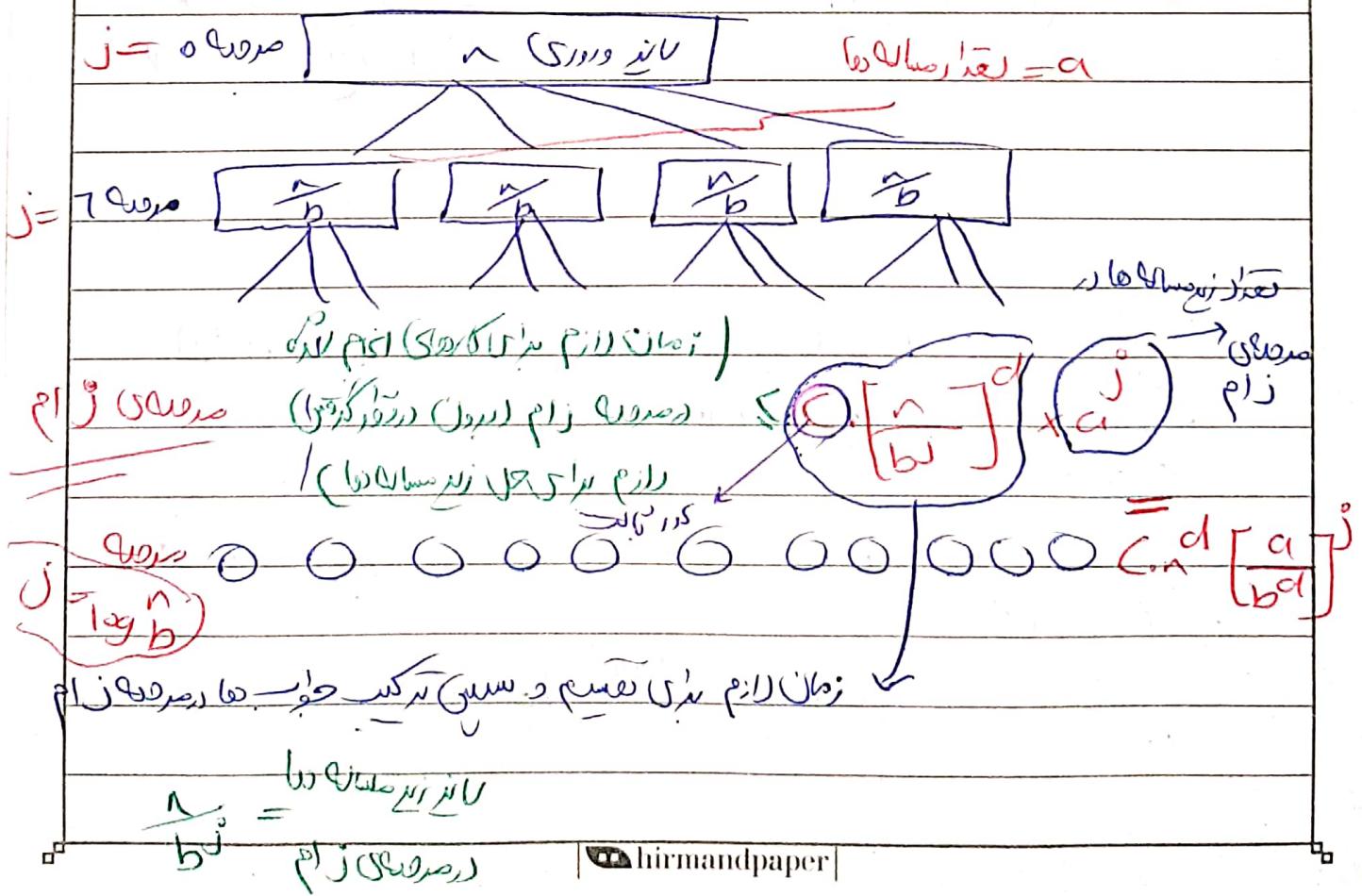
$$a=1-b=\frac{1}{4} \quad d=0, b=5 \quad , \quad a=1 \quad \text{G} \quad \text{G}_\text{left} \quad \text{G}_\text{right} \quad \textcircled{f}$$

*case 1*   $T(n) = O(n^0 \log n) = O(\log n)$

$$T(n) \geq T(\lceil n \rceil) + O(n^c) \in \Theta(n^{\lceil c \rceil})$$

case 2  $a < b^d$   $\Rightarrow a=1, b=2, d=5 \rightarrow T(n) = O(n^2)$

$$T(n) = \alpha T\left(\frac{n}{b}\right) + O(n^d) \text{ 경우 } \left( \text{부분문제 } \frac{1}{b} \right)$$



$$\text{زمان کار زن} = \sum_{j=0}^{\log_b^n} \text{عملیات های} = c n^d \sum_{j=0}^{\log_b^n} \left[ \frac{a}{b^d} \right]^j$$

$$T(n) = c n^d \sum_{j=1}^{\log_b^n} \left[ \frac{a}{b^d} \right]^j \rightarrow$$

Case 1:  $\frac{a}{b^d} = 1 \rightarrow T(n) = c n^d \times \log_b^n$   
 در این حالت زمان اجرا متناسب با  $n$  است.

Case 2:  $\frac{a}{b^d} < 1 \rightarrow T(n) = c n^d$

Case 3:  $\frac{a}{b^d} > 1 \rightarrow T(n) > c n^d = c n^{\log_b^n} = c n^{\log_b^a}$

$$n^{\log_b^n} = n^{\log_b^a} \quad \text{و) عملیات} \quad \text{ب) عملیات}$$

$$(e^{\log_b^n})^{\log_b^a} \quad \text{ب) عملیات}$$

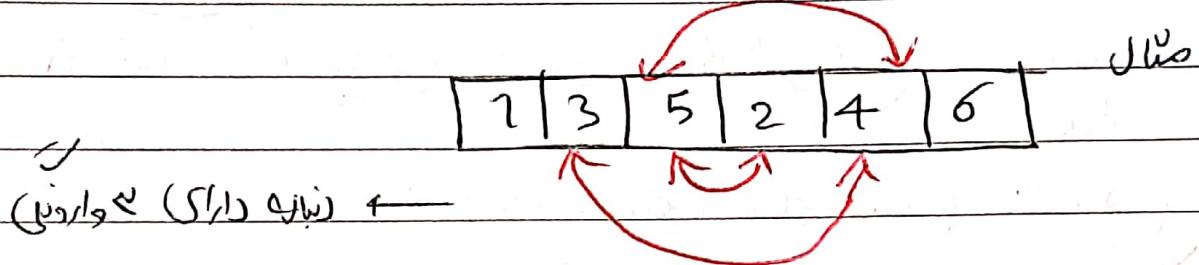
## SESSION 6

پیدا کردن تعداد وارونی نمایه از اندیار علیعی

و در (ج) پیدا نمایه از اندیار صهانی

صفحه ۸ تعداد وارونی هاک (نمایه و در)

$A[i] > A[j]$  و  $i < j$  را بگوییم (ج)



حواله نمایه بینیم و حداکثر تعداد وارونی که آن داشته باشد

حداکثر (n(n-1)) / 2 میباشد که اندیار مجموعه بردیست

حداکثر (n(n-1)) / 2 میباشد.

کاربرد فیلترینگ سیار کیم (صیغه عالی) برای فرآیند پیشگیری از اورج

نامناسب اندیار تعداد وارونی هاک دارند اینها را بگوییم خاصیت های اندیار تعداد وارونی هاک دارند

(brute force)

John (55 yrs) (JUL 4th)

## Brute force search for counting inversions

وقد يُؤدي إلى تضليل المُؤمنين

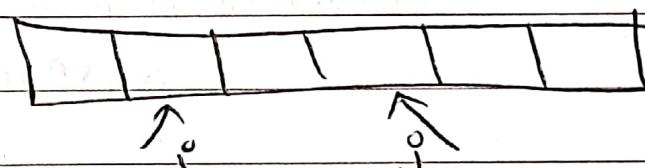
$$\text{numInv} = 0$$

for  $i = 1$  to  $n-1$  do

$$\text{numInv} = \text{numInv} + 1$$

return numInv

$O(n^k)$  باید باشد



$A[7] < A[5]$

6

+

(اویس دھنسوں ویلے رائے گاری) (عذر خارون) (وکی) (اویس)

عمر و میں اپنے حکایتی حیران کر دالے

~~Pop~~ ~~en~~

الـ زـ رـ الـ زـ

$$i \geq j > k$$

وارونیں (ولیتھی) اُرینیا) حصہ میں درجہ رال

د ب ج ز ه و ي أ ك م

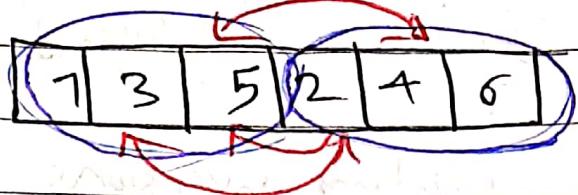
هیرمند پارس hirmandpaper

↑↑ |

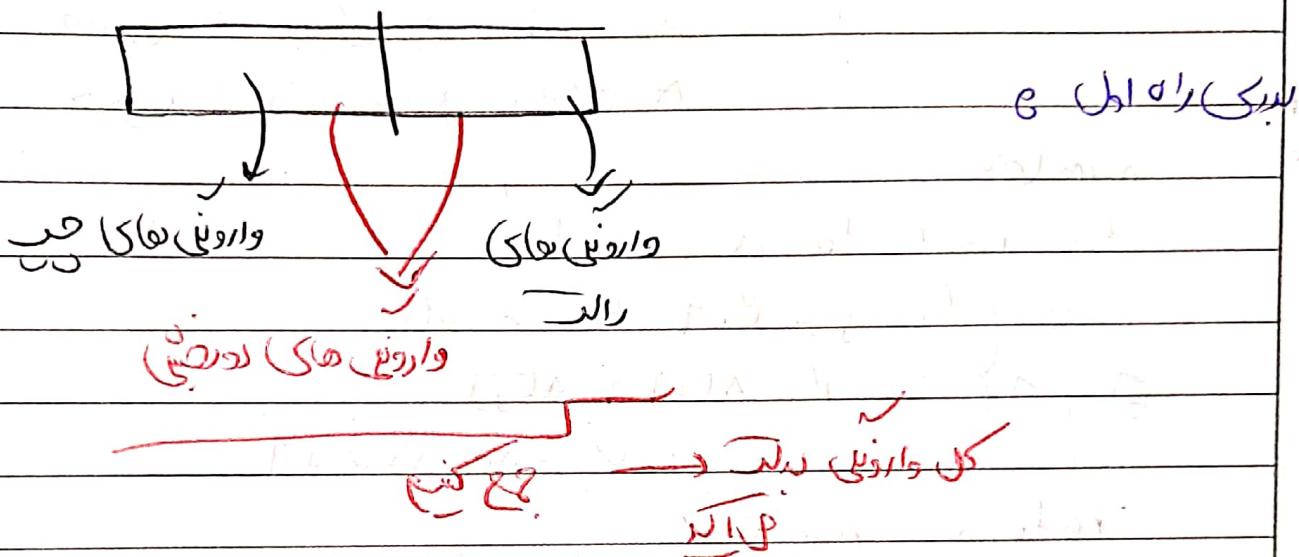
وَالْفُلَمْ (٤٢) وَالْمُكَبَّل

الله

↑ ↑ ↑



۳۲ داروینی (ورجینی)



داروینی کردن از اینجا ایجاد کردن داروینی A (ساخت B (خوبی) داروینی کردن داروینی B (خوبی))

if  $n=0$  or  $n=1$   
return 0

else

leftInv = CountInv(first half of A)

rightInv = CountInv(second half of A)

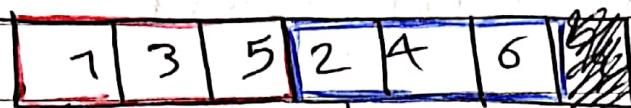
splitInv = countSplitInv(A)

return leftInv + rightInv + splitInv

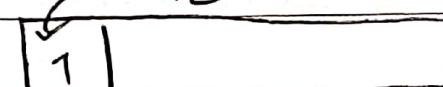
نحوه اولیه برای  $\sum_{i=1}^n$  داروینی (ورجینی) (عذر داروینی های)   
brute force

اگر بے اسٹم تعداد واریٹی (وختی) را صل (کر)  $\in \text{In merge sort}$

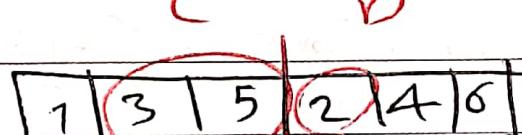
بایرانی (نیز)  $\in O(n^2)$



①



میں داری (وختی) کامل نہ وجود نہیں



②

داری کے تردید  
داری کے طبق مانند  
از بزرگترین 2



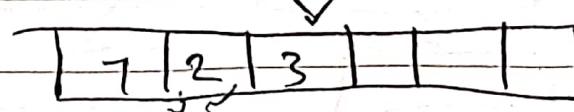
بے عدالتی (وختی) داری (وختی) داری

(وختی) وختی

③



$3 < 4$



میں داری (وختی) کامل نہ وجود نہیں

merge sort (sum)

$i=1, j=1$   $\text{splitInv} = 0$

for  $k=1$  to  $n$  do

if  $C[i] < D[j]$  then

$B[k] = C[i], i=i+1$

else

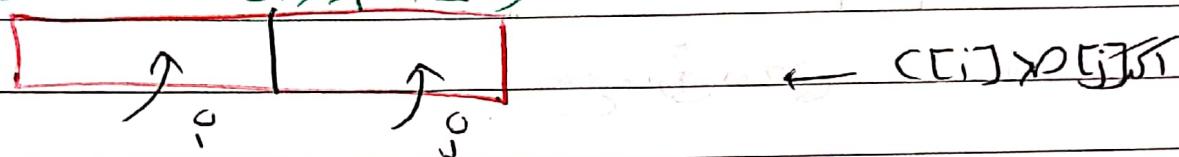
$B[k] = D[j], j=j+1$

$O(n)$

$\log n$

$\text{splitInv} = \text{splitInv} + (j - i + 1)$

return  $(B, \text{splitInv})$



بعد مراعات اهمیت کردن  $C$  در مسیر وارونه (عملیاتی)  $D$  را در  $B$  قرار دادیم.

پس از این مرحله  $\text{splitInv}$  چگونه باید باشد؟

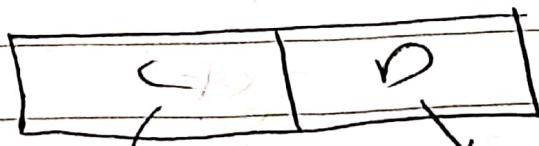
$$\text{splitInv} = \text{splitInv} + (j - i + 1)$$

splitInv برابر با  $j - i + 1$  است.

$$C = \underbrace{\dots}_{i} \rightarrow \underbrace{\dots}_{j} - (i-1)$$

sort & countInv

و فرمول



صریح کردیم که  $C$  و  $D$  مرتباً هستند و  $C$  و  $D$  مرتباً هستند.

if  $n=0$  or  $n=1$

return ( $A, 0$ )

else

$(C, \text{leftInv}) = \text{Sort\_and\_CountInv}$

$\rightarrow \text{first half of } A$

$(D, \text{rightInv}) =$

$\text{sort\_and\_countInv}(\text{second half of } A)$

تعارفی داده ترتیبی (Sort) بازگشته

$(B, \text{splitInv}) = \text{merge\_and\_countSplitInv}$

$(C_D)$

return  $(B, \text{leftInv} + \text{rightInv} + \text{splitInv})$

B ترتیبی داده

و این کار را

مجموع وارونه

با این سه ترتیبی داده

B  $\sim$  عبارتی بدهی (دی و اورجی) عبارتی (دی و اورجی) (دی و اورجی)

$T(n) = T(\frac{n}{2}) + O(n)$

$a = 1$      $b = 2$      $d = 1$

$\rightarrow a = b^d \rightarrow$

کاری کریم (این دویتی)  $O(n \log n)$

از این دویتی تعداد داده هایی که

لایه لایه بازگشتی

نَعْسَنَ وَخَلَقَ (الْجَنَّلِ) ۚ

## SESSION 7

## Quick Sort

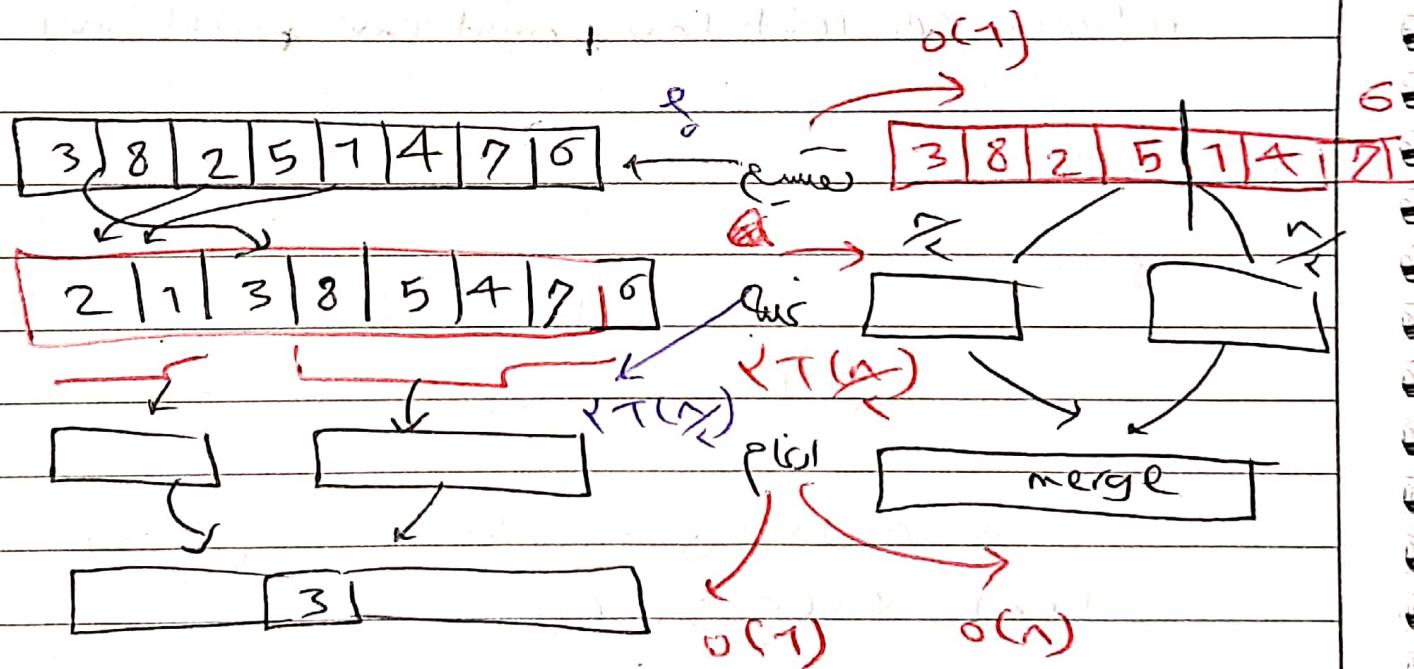
مودعہ ایکسپریس

ویژه) ۲ نیز دنباله از اداره صنایع باشند ترسیم رکوه

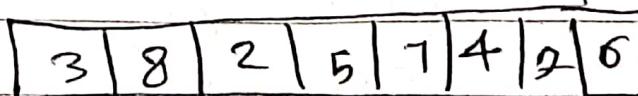
هـ ۸ صد و هشتاد و سی و چهارمین کوچه شهر

صریح‌ها  $\rightarrow$  صدیق (لاری) / عوی (بنایله) و عوی (اجامی) / لور  
از تقدیر نهان اعدا و قابل مفارستی با

~~বিক্রিয়ালী~~ quick sort ! merge sort



## المعنى (خبار)



نہ صورت تاریخی عمل میں لونا

لیست  $A$  و  $B$  داشته باشیم

آنکه  $A$  و  $B$  را از کوچک به بزرگ مرتب کنیم

اگر  $n \leq 1$  باشد

return

choose a pivot element  $p$

partition  $A$  around  $p$

recursively sort first part of  $A$

..... second .....

لیست

3	8	2	5	1	4	7	6
---	---	---	---	---	---	---	---

$\Theta(n)$

2	1	3	6	7	4	5	8
---	---	---	---	---	---	---	---

الخطوة 1

حافظة اولی

لیست  $A$  افتد که جایجا کنیم

$p$	$< p$	$> p$	$m$
-----	-------	-------	-----

$p$	$< p$	$> p$	$m$
-----	-------	-------	-----

$n > p$

$n < p$

8.16

W<sub>i</sub> ( $S_{\text{left}}$ )  $\rightarrow$  A ( $S_{\text{mid}}$ )  $\rightarrow$  B ( $S_{\text{right}}$ ) partition  
pivot  $\in$  B ( $S_{\text{right}}$ )

$p = A[i]$   
 $i = l+1$   
if  $A[j] > p$  do nothing

for  $j = l+1$  to  $r$  do  
if  $A[j] < p$  then  
swap  $A[j]$  and  $A[i]$   
 $i = i + 1$

swap  $A[l]$  and  $A[i-1]$  // place pivot correctly  
return  $i-1$

( $S_l$  pivot)

$O(n)$   $\Theta(l(i-l))$

in place  $\Leftrightarrow$  no swap

Quick sort ( $A, l, r$ )  
inputs array  $A$  of  $n$  distinct integers,  
left, right endpoints ( $l, r \in \mathbb{N}_1, 2, \dots, n$ )

post condition of elements of sub array

$A[l], A[l+1], \dots, A[r]$  are sorted from smallest to largest

if  $l \geq r$  then

return

$i = \text{choosePivot}(A, l, r)$   
swap  $A[l], A[i]$

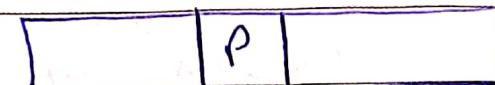
$j^o = \text{partition}(A, l, r)$  //  $j^o$  = new pivot position

quicksort( $A, l, j^o - 1$ ) // recurse on first part

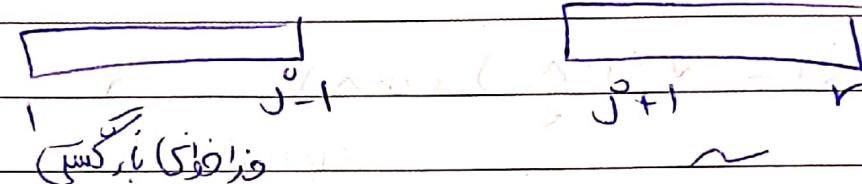
quicksort( $A, j^o + 1, r$ ) // recurse on second part

$l = 1$

$r = n$



partition( $A, l, r$ )  $\rightarrow$   $j^o$  found



وَزَانَتْ كِبِيرًا

session 8

pivot (pivot)

choose pivot()

input: array  $A$  of  $n$  distinct integers,  
left, right endpoints  $l, r \in \{1, 2, \dots, n\}$

output: index  $i \in \{l, l+1, \dots, r\}$

return  $i$  such that  $A[i] \leq A[j] \forall j \in [l, i]$

$$T(n) = T(n-1) + C_n \rightarrow O(n)$$

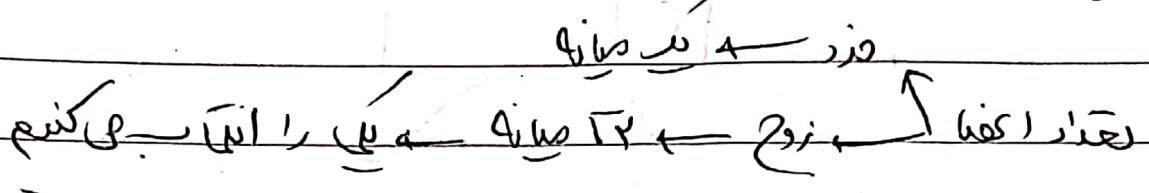
$$= T(n-2) + C(n-1) + C_n$$

$$= T(0) + C \sum_{i=1}^n i = n(n-1) + O(n^2)$$

Pivot  $\rightarrow$  il (سیکل)  $\leftarrow$  گز

return position of the median element  
of  $A[l], \dots, A[r]$

(جسے وہی جیسی ایک جگہ ہے جو میڈین (median) ہے اسے  
time کہا جاتا ہے)



$$T(n) = T\left(\frac{n}{2}\right) + O(n) \rightarrow O(n \log n)$$

(بھروسہ)

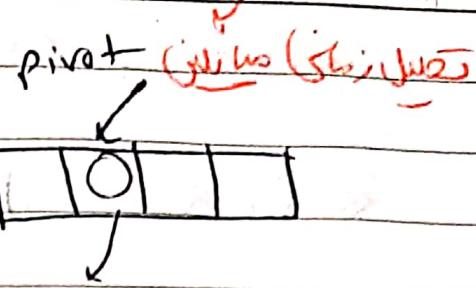
(کوئی جگہ جو بھی ہے) اس کو میڈین کہا جائے گا

Pivot (Random)  
return an element of  $A[l, l+1, \dots, r]$   
chosen uniformly at random

$< P$	$P$	$> P$
25 - 75%	$\approx$	

$$T(n) = T\left(\frac{n}{2}\right) + T\left(\frac{n}{2}\right) + O(n)$$

وہی جگہ جو ہے جس کا (جیسے) ہے  $\neq$  (جیسے) ہے



درگاه از ادار ۱ تا ۶ لغوند نیست

$$T(n) = \text{cost of pivots} +$$

حل مسأله با وفاده کنندگان

که روابط حاصل از زمان اوران

$$T(1)(n) = T(0) + T(n-1) + C_n$$

وقتی یک عدد اول بینف

pivot انتبار

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

:

$$T(n) = T(n-1) + T(0) + C_n$$

$$\downarrow \quad \text{وقتی یک عدد اول بینف} \quad = \quad \text{وقتی یک عدد اول بینف}$$

$$T(n) = \frac{1}{n} \sum_{i=1}^n T_i(n) = \frac{1}{n} \left[ \sum_{i=0}^{n-1} (T(i) + T(n-1-i)) + C_n \right]$$

$$= \frac{1}{n} \sum_{i=0}^{n-1} T(i) + C_n$$

$$n T(n) = C_n^n + \sum_{i=0}^{n-1} T(i)$$

نیز میتوانیم

$$(n-1) T(n-1) = C(n-1)^{n-1} + \sum_{i=0}^{n-2} T(i)$$

$$n T(n) = C n^n - C(n-1)^n + (n+1) T(n-1)$$

$C(n-1) \rightarrow$

$$\ll c'n + (n+1)T(n-1)$$

$$\ll c'n(n+1) \sum_{j=1}^{n+1}$$

$$\ll c'n(n+1) \int_{1/m}^{n+1} \frac{1}{x} dx$$

$$\rightarrow T(n) \leq c'(n+1) \log(n+1)$$

$$\rightarrow T(n) = O(n \log n)$$

اگر pivot را العارف

کنیم

$$T(n) = \frac{1}{n} \sum_{i=1}^n T_i(n)$$

حال که اگر نجات آنچه های (عکس) و این را فتح کنیم

از طرفی (هر دوی) از این 6 میانه (هر دوی) از این 2 ترکیب (log n) QuickSort (عکس)

از طرفی این ترکیب نیز احتمالی حافظه زیاد ندارد (پاره ایکس) (عکس)

session 9

(WIS)

(ب) مجموعی مسئلہ (WIS)

weighted independent set

وہی کوئی کٹا جاتے ہے کہ ہر رکی آنے والے نہیں

$$w \geq 0$$

$$v \in V$$

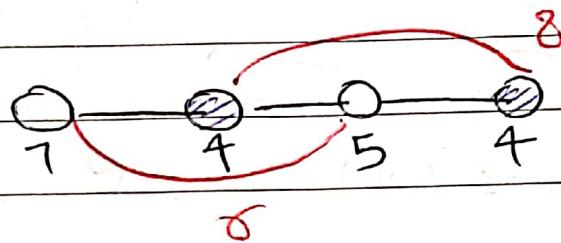
$$\sum_{v \in S} w_v$$

$$G = (V, E)$$

مترافق

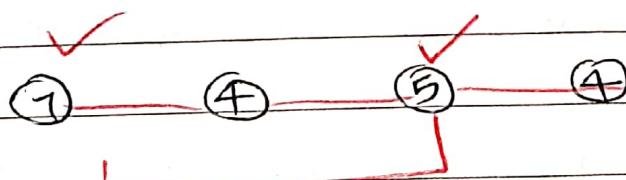
وہی کوئی مجموعی مسئلہ (WIS) میں از رکی (non-redundant) مفہوم فہرست میں

عنہ (ہر رکی) اڑان یا (وہی کوئی مجموعی مسئلہ (WIS) میں از رکی (non-redundant) مفہوم فہرست میں)



حل بیان کر کے  $T$  صلی ہونے والے مجموعی مسئلہ (WIS) کے

اوپر حصہ میں  $G$  میں دھوکا کاں اور سیر حصہ میں  $G$  میں دھوکا کاں اور سیر حصہ میں  $G$  میں دھوکا کاں اور سیر حصہ میں  $G$  میں دھوکا کا اسی طبقہ رکی تریخ تریخ فہرست رکی کر کے بے رکی قبیلہ و چھپاں نالہ



$$1+5=6 \rightarrow 6 < 8 \quad \text{X}$$

اوپر حصہ

$$S = \emptyset$$

sort vertices of  $V$  by weight

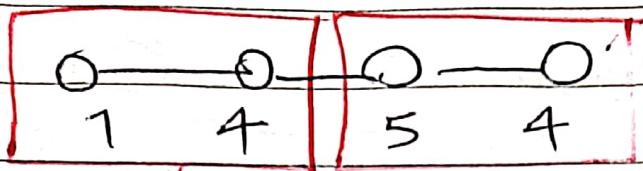
for each  $v \in V$ , in nonincreasing order of weight do

if  $S \cup \{v\}$  is an independent set of  $G$

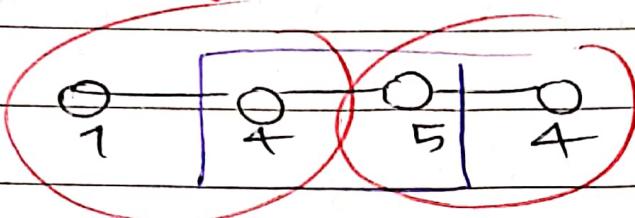
return  $S$

then  $S = S \cup \{v\}$

Algo, it uses same G but power less



الآن merge کرنا



4 5 merge کرنا

X ← merge( $S_1$ ,  $S_2$ ) (لزماً متساوية)  
کیم متساوية

WIS is Divide & Conquer

$G_1$  = first half of  $G$

$G_2$  = second half of  $G$

$S_1$  = recursively solve the WIS problem on

$G_1$

$S_2$  = ~ ~ ~ ~  $G_2$

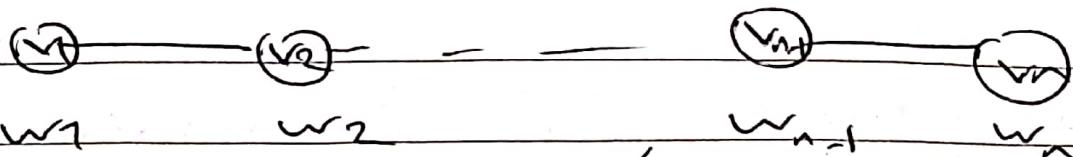
Combine  $S_1, S_2$  into a solution  $S$  for  $G$

return  $S$

وکیو اور) (بیس الگو ای وہی وہی جو  $G_1, G_2$  \*

MWS 8 for you, too

$$G = (V, E)$$



Jens C. J. Z.

الله جبارٌ وَّ عَزِيزٌ

لکھنؤ کے ساتھ ایک ایسا جگہ تھا جو اپنے بارے میں بے شکریتی کے لئے مشہور تھا۔ اسی جگہ پر ایک بڑا مکان تھا جو اپنے بارے میں بے شکریتی کے لئے مشہور تھا۔ اسی جگہ پر ایک بڑا مکان تھا جو اپنے بارے میں بے شکریتی کے لئے مشہور تھا۔

$$G_{N-1}$$

$\beta \cup_n \mathcal{S} \beta \cup_1 = \text{nil}$

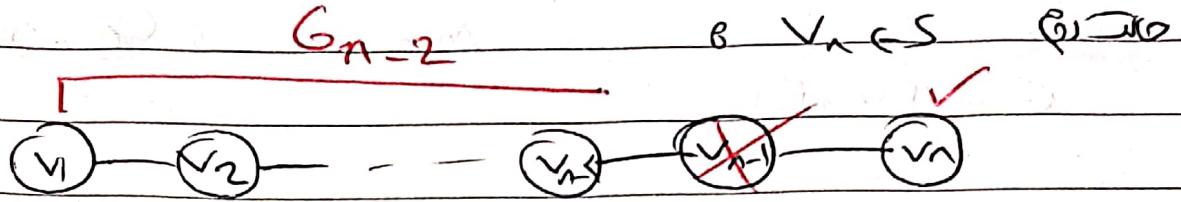


ک نیز مجموعہ از بھوکاری (رائے) میں دوسرے زیر مجموعہ کاری (سیکول) ارزانی رائے اعلان

الـ  $\max_{n \in \mathbb{N}} G_n$  يـ  $\rightarrow$   $\lim_{n \rightarrow \infty} G_n$

که در نظری معتبر است که برای گذاشت و متعاقن باشد از

۱۰۰٪ کمتر گزینه



বিবরণ করুন  $G_{n-2}$  যার সংস্করণ হ'ল  $\{1, 4, 5, 2, 3\} \cup \{n\}$

(از آنچه) کردن گزینه هایی را که ممکن است در مجموعه  $G_{n-2}$  باشند.

کس کے لئے  $G_{n-2}$  میں ایسا جو سلسلہ بای 6 باونز ہے جو  $G_n$  کے لئے ممکن ہے اسے  $G_n$  کا ٹکڑا کہا جائے گا۔

$$w^* + w_n \leq w \xrightarrow{w_n \rightarrow 0} w^* \leq w - w_n$$

$w^* \leq w - w_n$

III Gr. 1 Sk. mws w/ (7)

$$\text{الآن } G_n \text{ يعطى بـ} \sum_{i=1}^n w_i \text{ ونلاحظ أن } G_{n+1} = G_n + w_{n+1}$$

$i = 2, 3, \dots, n$   $w_i$  &  $\text{MIS}_i$

$$w_i = \max(w_{i-1}, w_{i-2} + \text{tw}_i)$$

$\text{MIS}_i \setminus v_i$

[SESSION 70]

MIS  $\cup \{v_i\}$

Input a Path Graph  $G$  with vertex set  $\{v_1, v_2, \dots, v_n\}$  and a non-negative weight  $w_i$  for each vertex  $v_i$ .

Output a maximum weight independent set of  $G$

if  $n=0$

return empty set

if  $n=1$

return  $\{v_1\}$

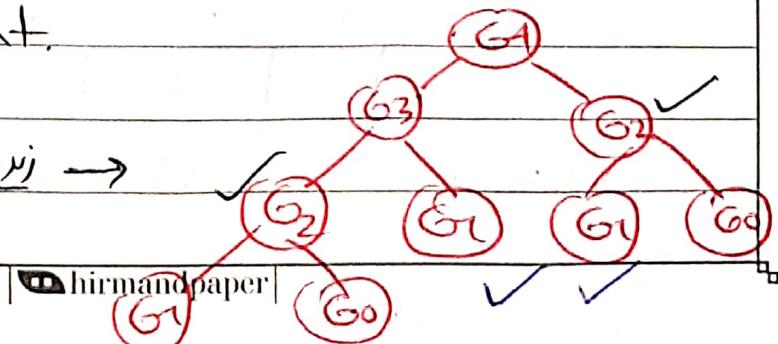
$S_1$  & recursively compute an MIS of  $G_{n-1}$

$S_2$  &  $\sim \sim \sim \sim G_{n-2}$

return  $S_1$  or  $S_2 \cup \{v_n\}$  whichever has higher weight.

(Figure 10)  $\rightarrow$   $S_1$  or  $S_2$

$O(n^2)$   $O(n^2)$



جواب: هر کار که نمایند میلر، از ذخیره کنن (Memoization) میگیرند

### memoization

ذخیره کنن  $\leftarrow$  پس از bottom-up برسی  $\leftarrow$  دنبالاً حساب کنن

WIS

$A = \text{length} - (n+1)$  array

$A[0] = 0$

$A[1] = w_1$

for  $i=2$  to  $n$  do

$A[i] = \max \{ A[i-1], A[i-2] + w_i \}$

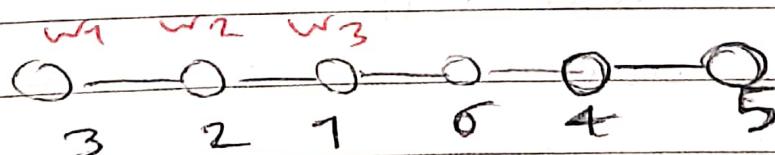
تویی راه ذخیره کنن بررسی

return  $A[n]$

نحوی: زیر مجموعه از

$O(n)$   $\leftarrow$  دسته بندی خوبی که  $O(n)$  است و  $O(1)$  است

$(O(1))$  بعنوان از  $O(n)$  است



$w_1$	$w_2$	$w_3$						$= 14$
3	2	1	6	4	5			

$$A[0] = 0$$

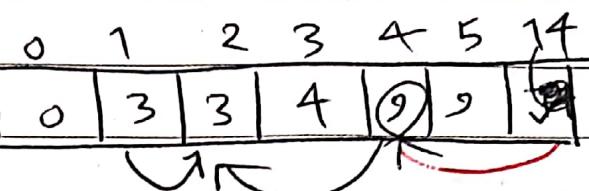
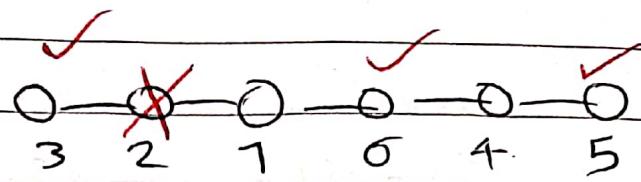
$$A[1] = 3$$

$$A[2] = \max\{ A[1], A[0] + w_2 \} = 3$$

$$A[3] = A[2] \max\{ A[2], A[1] + w_3 \} = 3$$

birmandpaper

(for  $i=0$  to  $n-1$ )



$$3 + 0 + 5 = 14$$

$$3 + 0 = 9 \quad 9 + 5 = 14$$

$s = 0$   
 $i = n \rightarrow 0, 1, 2, 3, 4, 5, 6, 7, 8, 9$

while  $i > 2$  do survive

if  $A[i-1] > A[i-2] + w_i$  then

$i = i - 1$   $\rightarrow$   $i = 0, 1, 2, 3, 4, 5, 6, 7, 8, 9$

survive  
else

$s = s \cup v_i$

$i = i - 2$

survive

if  $i = 1$  then

$s = s \cup \{v_i\}$

return  $s$

ریکارڈینگ کو اپنے لئے تیار کروں ۱) EDP  $\rightarrow$  (کوئی  
6<sub>0</sub>, 6<sub>1</sub>,  $\dots$ , 6<sub>n</sub>)

گلہیں کو اپنے لئے تیار کروں ۲)

کھس کو اپنے لئے تیار کروں ۳)

$A[i] = \max(A[i-1], A[i-2] + w_i)$

۴)  $A[n]$  از جا بے عیں از جا بے عیں

## Dynamic Programming

It is an optimization technique

Do you have something you can cache?

So you can use it.

is a way to solve problems by breaking it down into a collection of subproblems.

Solving each of those subproblems just once and storing their solutions in case next time the same subproblem occurs

first time you will solve it again → Caching

every backup

Opportunity cost of re-solving problem

Opportunity cost of

If you cache it (logically) = Memoization

function addTo80(n)

return n + 80;

y

for (y)

memoize

cache

let cache = {};

function memoizedAddTo80(n)

if (n in cache) → Non-existent

return cache[n];

else

cache[n] = n + 80;

y

y

return cache[n];

global scope is  $\text{c}(500)(5)$   
function memoizationAddTo80(a) {  
 let cache = {};  
 return function(n) {  
 if (n in cache) {  
 return cache[n];  
 } else {  
 console.log('long time');  
 cache[n] = n + 80;  
 return cache[n];  
 }  
 };  
}  
const memoized = memoizationAddTo80();  
console.log(memoized(5));

Dynamic programming = divide & conquer  
+ Memoization

- ① Can be divided into subproblems
  - ② Recursive Solution
  - ③ are there repetitive subproblems
  - ④ memoize subproblems

## Fibonacci & memoized solution

```
def fib(n, memo):
    if memo[n] != null: → O(1)
        return memo[n] → O(1) | O(1)

    if n == 1 or n == 2: → O(1)
        result = 1
    else:
        result = fib(n-1) + fib(n-2) → fib(n-1) + fib(n-2) = O(1)

    memo[n] = result
    return result
```

$$T(n) = \underbrace{\text{O(1)}}_{\text{خواهد بود}} \times \underbrace{\text{O(1)}}_{\text{خواهد بود}} = \text{O(1)}$$

لهم ای کر فرا فکر کن  
لهم ای کر جم خواهی کن  
لهم ای کر حکیم خواهی کن  
لهم ای کر راضی خواهی کن

$$\text{result} = \text{fib}(n-1) + \text{fib}(n-2)$$

لهم ای کر فرا فکر کن  
لهم ای کر جم خواهی کن  
لهم ای کر حکیم خواهی کن  
لهم ای کر راضی خواهی کن

$$T(n) = (n+1) \cdot O(1) = O(n)$$

لهم ای کر فرا فکر کن  
لهم ای کر جم خواهی کن  
لهم ای کر حکیم خواهی کن  
لهم ای کر راضی خواهی کن

## def fib\_bottom\_up Bottom Up Approach

```
if n == 1 or n == 2:
```

```
    return 1
```

```
bottom_up = new int[n+1]
```

```
bottom_up[1] = 1  
bottom_up[0] = 1
```

for i from 3 upto 18

$$\text{bottom\_vp}[i] = \text{bottom\_vp}[i-1] + \text{bottom\_vp}[i-2]$$

return bottom.vpcn)

O(n)

حاجات ( ) اخرين) ملخص جزء اول

تک فیبوناچی + برای اداره‌خانه نزدک صلی

6 - in bottom-up جون از تاریخ بارکس memorize با لایه لایه  
برای stack لایه لایه با آخر از آخر

## Knapsack problem

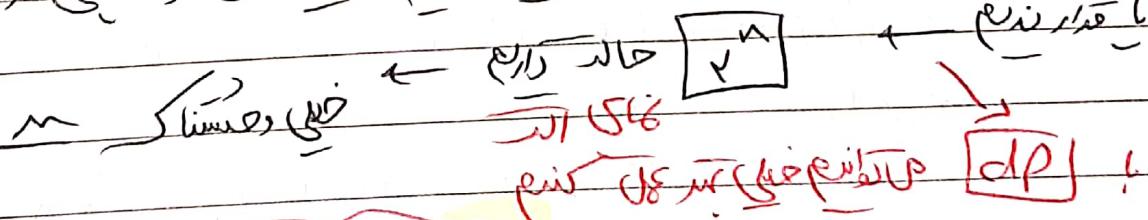
weight (kg)	1	2	4	2	5
value (\$)	5	3	5	3	2

$$n = 5$$

کوہ نیشنل پارک کے قریب میں ایک چھوٹا سا جنگل میں ایک چھوٹا سا چکنے والی داروں کا ایک مکان تھا۔

10 kg) WIND TIDE

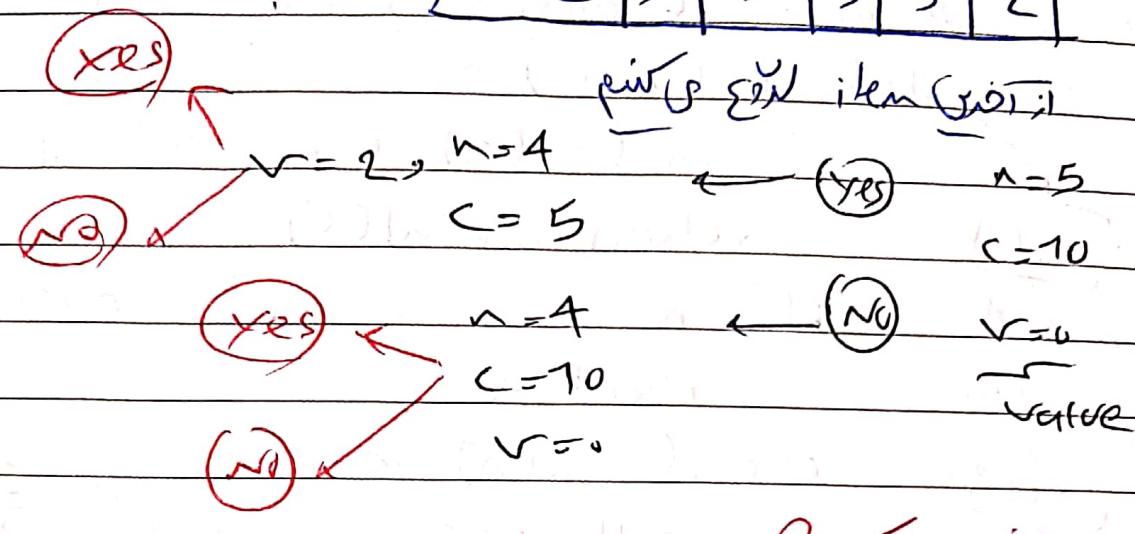
لے خارج کریں کہ  $Q_i$  کی value،  $\pi_i$  اور  $\alpha_i$  توک کرنے پر item  $i$  کا سعیم کیا تھا توی کو یہ دیکھ قرار دیں گے۔



## Dynamic Programming

- ① Recursive Solution
  - ② memorize intermediate Results
  - ③ bottom up approach

	weight	1	2	4	2	5
	value	5	3	5	3	2



عن طریق قسمی هر دوی

naive recursive Solution (code)

def KS( $n, c$ ):

if  $n == 0$  or  $c == 0$  & base case

result = 0

KS(5, 10)

else if  $w[n] > 0$  & که وزن فعلی

result = KS( $n-1, c$ ) که این قاعده کافی نیست

else

result ←  $\max\{result, v[n] + KS(n-1, c-w[n])\}$

tmp1 = KS( $n-1, c$ ) → آنکه زیرا

tmp2 =  $v[n] + KS(n-1, c-w[n])$  → اینکه زیرا

result =  $\max\{tmp1, tmp2\}$

return result

این (تولید)  $\max$  میان

این حسنه کنیم و این قدر را برمیگردانیم

## dp Solution

// initialize  $\text{arr}[n][c] = \text{undefined}$  \*

def ks(n, c) :

    if  $\text{arr}[n][c] \neq \text{undefined}$  : \*

        return  $\text{arr}[n][c]$

$n, c$

    if  $n == 0$  or  $n == 0$  // base case

        result = 0

        مقدار

    else if  $w[n] > c$

        result =  $\text{ks}(n-1, c)$  \*

$n-1$

    else :

        tmp1 =  $\text{ks}(n-1, c)$

        tmp2 =  $v[n] + \text{ks}(n-1, c - w[n])$

        result =  $\max(\text{tmp1}, \text{tmp2})$  \*

$\text{arr}[n][c] = \text{result}$  \*

    return result.

رسیجی ویسی

کوئی بھائی نہیں

کوئی بھائی نہیں، لیکن

$(n+1)(c) =$

لیکن کوئی نہیں

$O(nC)$

## Longest Common Subsequence Problem

P = "BATD"

PN = یعنی لگن لگن

Q = "ABACD"

یا اسکی

→ "BAD"

NP problems

① recursion

② memorize Intermediate results

③ Bottom-up

hirmandpaper

## Recursive solution

$LCS(p_0, q_0)$

(Case 1)  $\epsilon$

$$p_0 = " \boxed{p_1} m "$$

$$q_0 = " \boxed{q_1} m "$$

( $m$  میں جو کوئی بھی)

اپنے LCS  $\underline{\underline{n}}$

میں  $q_1 \neq p_1$

میں  $m$  کا

$\rightarrow LCS(p_0, q_0) =$

$1 + LCS(p_1, q_1)$

$\boxed{n}$  اسی

(Case 2)  $\theta$

$$p_0 = " \boxed{p_1} m "$$

$$q_0 = " \boxed{q_1} y "$$

( $m$  میں آخر صفا در)

$LCS(p_0, q_0) = \max \{ LCS(p_1, q_0), \text{ ایسی حال میں } \}$

$LCS(p_0, q_1) \}$

$p_1 \neq q_1$  کی وجہ سے

Recursive solution

def  $LCS(p, q, n, m)$

if  $n == 0$  or  $m == 0$  // base case

result = 0

series کیا else if  $p[n-1] == q[m-1]$

result = 1 +  $LCS(p, q, n-1, m-1)$

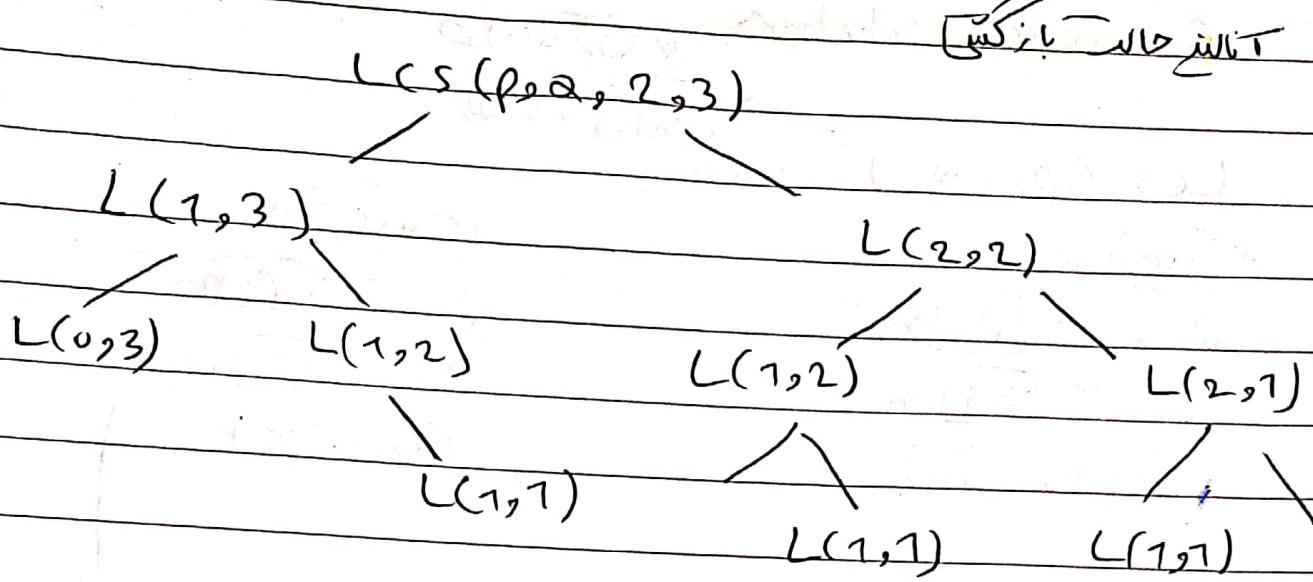
else if  $p[n-1] \neq q[m-1]$

series کیا tmp1 =  $LCS(p, q, n-1, m)$

tmp2 =  $LCS(p, q, n, m-1)$

result =  $\max \{ tmp1, tmp2 \}$

return result



(ii) duplicate (مکانیزم)

$p = "AA"$  e جس

$q = "BBB"$

$O(n^m)$

$O(n^m)$

memorize

initialize  $arr[n][m]$  to undefined

def LCS(p, q, n, m)

if  $arr[n][m] \neq \text{undefined}$  :

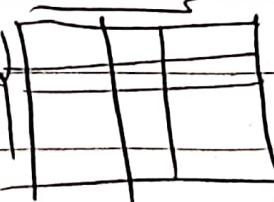
return  $arr[n][m]$

if  $n == 0$  or  $m == 0$  :

result = 0

else if  $p[n-1] == q[m-1]$  :

arr[n][m] = result + 1



else if

$arr[n][m] = result$

وہی ایسا

return result

hirmandpaper

کو

$O(nm)$

→ اسی  $O(1)$  کے لئے  $O(nm)$  میں بار بار پڑھ لے جائیں

## Bottom up solution

$$P = "AA" \rightarrow \text{LCS}(P, Q, 2, 3)$$

$L(0,0)$	$m=0$	$m=1$	$m=2$	$m=3$
$n=0$	0	0	0	0
$n=1$	0	1	1	1
$n=2$	0	1	2	2

لطفاً فرضی کرو که  $P$  و  $Q$  دو زیرمتوازن سکونتی هستند  
 پس  $L(n, m) = \text{substring}(P[0:m], Q[0:n])$

Find sets of numbers that add up to 70

$\rightarrow [2, 4, 6, 10]$  set which adds 22

$[2, 4, 6, 10] \rightarrow$  this  $\{6, 7, 9\}$   
 $\{2, 4, 7, 9\}$

$\{6, 7, 9\} \rightarrow$   $\{10, 4\}$   
 $\{10, 4\} \rightarrow$   $\{9\}$   
 $\{9\} \rightarrow$   $\{6\}$   
 $\{6\} \rightarrow$   $\{4\}$

$\boxed{10} \leftarrow$   $\{10, 6, 4, 2\}$   $\{10, 6, 4, 2\}$   $\{10, 6, 4, 2\}$

$O(n^m)$  time complexity for subset sum

def count\_sets(arr, total)  $\rightarrow$  T114

$\boxed{10}$   $\downarrow$  return rec(arr, total, arr.length - 1)

$\boxed{2, 4, 6}$   $\downarrow$  def rec(arr, total, index)  $\rightarrow$  T114

$\uparrow$  if total == 0:  $\boxed{1}$

$\boxed{1}$   $\downarrow$   $i=0$

return 1  $\uparrow$   $\boxed{2, 4, 6, 10}$

else if total < 0:  $\boxed{0}$

$\boxed{0}$   $\downarrow$   $i=1$

return 0

$\boxed{0}$   $\downarrow$   $i=2$

$\uparrow$  else if  $i > \text{length}$ :  $\boxed{0}$

$\boxed{0}$   $\downarrow$   $i=3$

return 0

$\boxed{0}$   $\downarrow$   $i=4$

$\uparrow$   $\boxed{0}$   $\downarrow$   $i=5$

$\boxed{0}$   $\downarrow$   $i=6$

$\boxed{0}$   $\downarrow$   $i=7$

$\boxed{0}$   $\downarrow$   $i=8$

$\boxed{0}$   $\downarrow$   $i=9$

$\boxed{0}$   $\downarrow$   $i=10$

$\boxed{0}$   $\downarrow$   $i=11$

$\boxed{0}$   $\downarrow$   $i=12$

$\boxed{0}$   $\downarrow$   $i=13$

$\boxed{0}$   $\downarrow$   $i=14$

$\boxed{0}$   $\downarrow$   $i=15$

$\boxed{0}$   $\downarrow$   $i=16$

$\boxed{0}$   $\downarrow$   $i=17$

$\boxed{0}$   $\downarrow$   $i=18$

$\boxed{0}$   $\downarrow$   $i=19$

$\boxed{0}$   $\downarrow$   $i=20$

$\boxed{0}$   $\downarrow$   $i=21$

$\boxed{0}$   $\downarrow$   $i=22$

$\boxed{0}$   $\downarrow$   $i=23$

$\boxed{0}$   $\downarrow$   $i=24$

$\boxed{0}$   $\downarrow$   $i=25$

$\boxed{0}$   $\downarrow$   $i=26$

$\boxed{0}$   $\downarrow$   $i=27$

$\boxed{0}$   $\downarrow$   $i=28$

$\boxed{0}$   $\downarrow$   $i=29$

$\boxed{0}$   $\downarrow$   $i=30$

$\boxed{0}$   $\downarrow$   $i=31$

$\boxed{0}$   $\downarrow$   $i=32$

$\boxed{0}$   $\downarrow$   $i=33$

$\boxed{0}$   $\downarrow$   $i=34$

$\boxed{0}$   $\downarrow$   $i=35$

$\boxed{0}$   $\downarrow$   $i=36$

$\boxed{0}$   $\downarrow$   $i=37$

$\boxed{0}$   $\downarrow$   $i=38$

$\boxed{0}$   $\downarrow$   $i=39$

$\boxed{0}$   $\downarrow$   $i=40$

$\boxed{0}$   $\downarrow$   $i=41$

$\boxed{0}$   $\downarrow$   $i=42$

$\boxed{0}$   $\downarrow$   $i=43$

$\boxed{0}$   $\downarrow$   $i=44$

$\boxed{0}$   $\downarrow$   $i=45$

$\boxed{0}$   $\downarrow$   $i=46$

$\boxed{0}$   $\downarrow$   $i=47$

$\boxed{0}$   $\downarrow$   $i=48$

$\boxed{0}$   $\downarrow$   $i=49$

$\boxed{0}$   $\downarrow$   $i=50$

$\boxed{0}$   $\downarrow$   $i=51$

$\boxed{0}$   $\downarrow$   $i=52$

$\boxed{0}$   $\downarrow$   $i=53$

$\boxed{0}$   $\downarrow$   $i=54$

$\boxed{0}$   $\downarrow$   $i=55$

$\boxed{0}$   $\downarrow$   $i=56$

$\boxed{0}$   $\downarrow$   $i=57$

$\boxed{0}$   $\downarrow$   $i=58$

$\boxed{0}$   $\downarrow$   $i=59$

$\boxed{0}$   $\downarrow$   $i=60$

$\boxed{0}$   $\downarrow$   $i=61$

$\boxed{0}$   $\downarrow$   $i=62$

$\boxed{0}$   $\downarrow$   $i=63$

$\boxed{0}$   $\downarrow$   $i=64$

$\boxed{0}$   $\downarrow$   $i=65$

$\boxed{0}$   $\downarrow$   $i=66$

$\boxed{0}$   $\downarrow$   $i=67$

$\boxed{0}$   $\downarrow$   $i=68$

$\boxed{0}$   $\downarrow$   $i=69$

$\boxed{0}$   $\downarrow$   $i=70$

$\boxed{1}$   $\downarrow$   $i=71$

$\boxed{1}$   $\downarrow$   $i=72$

$\boxed{1}$   $\downarrow$   $i=73$

$\boxed{1}$   $\downarrow$   $i=74$

$\boxed{1}$   $\downarrow$   $i=75$

$\boxed{1}$   $\downarrow$   $i=76$

$\boxed{1}$   $\downarrow$   $i=77$

$\boxed{1}$   $\downarrow$   $i=78$

$\boxed{1}$   $\downarrow$   $i=79$

$\boxed{1}$   $\downarrow$   $i=80$

$\boxed{1}$   $\downarrow$   $i=81$

$\boxed{1}$   $\downarrow$   $i=82$

$\boxed{1}$   $\downarrow$   $i=83$

$\boxed{1}$   $\downarrow$   $i=84$

$\boxed{1}$   $\downarrow$   $i=85$

$\boxed{1}$   $\downarrow$   $i=86$

$\boxed{1}$   $\downarrow$   $i=87$

$\boxed{1}$   $\downarrow$   $i=88$

$\boxed{1}$   $\downarrow$   $i=89$

$\boxed{1}$   $\downarrow$   $i=90$

$\boxed{1}$   $\downarrow$   $i=91$

$\boxed{1}$   $\downarrow$   $i=92$

$\boxed{1}$   $\downarrow$   $i=93$

$\boxed{1}$   $\downarrow$   $i=94$

$\boxed{1}$   $\downarrow$   $i=95$

$\boxed{1}$   $\downarrow$   $i=96$

$\boxed{1}$   $\downarrow$   $i=97$

$\boxed{1}$   $\downarrow$   $i=98$

$\boxed{1}$   $\downarrow$   $i=99$

$\boxed{1}$   $\downarrow$   $i=100$

$\boxed{1}$   $\downarrow$   $i=101$

$\boxed{1}$   $\downarrow$   $i=102$

$\boxed{1}$   $\downarrow$   $i=103$

$\boxed{1}$   $\downarrow$   $i=104$

$\boxed{1}$   $\downarrow$   $i=105$

$\boxed{1}$   $\downarrow$   $i=106$

$\boxed{1}$   $\downarrow$   $i=107$

$\boxed{1}$   $\downarrow$   $i=108$

$\boxed{1}$   $\downarrow$   $i=109$

$\boxed{1}$   $\downarrow$   $i=110$

$\boxed{1}$   $\downarrow$   $i=111$

$\boxed{1}$   $\downarrow$   $i=112$

$\boxed{1}$   $\downarrow$   $i=113$

$\boxed{1}$   $\downarrow$   $i=114$

$\boxed{1}$   $\downarrow$   $i=115$

$\boxed{1}$   $\downarrow$   $i=116$

$\boxed{1}$   $\downarrow$   $i=117$

$\boxed{1}$   $\downarrow$   $i=118$

$\boxed{1}$   $\downarrow$   $i=119$

$\boxed{1}$   $\downarrow$   $i=120$

$\boxed{1}$   $\downarrow$   $i=121$

$\boxed{1}$   $\downarrow$   $i=122$

$\boxed{1}$   $\downarrow$   $i=123$

$\boxed{1}$   $\downarrow$   $i=124$

$\boxed{1}$   $\downarrow$   $i=125$

$\boxed{1}$   $\downarrow$   $i=126$

$\boxed{1}$   $\downarrow$   $i=127$

$\boxed{1}$   $\downarrow$   $i=128$

$\boxed{1}$   $\downarrow$   $i=129$

$\boxed{1}$   $\downarrow$   $i=130$

$\boxed{1}$   $\downarrow$   $i=131$

$\boxed{1}$   $\downarrow$   $i=132$

$\boxed{1}$   $\downarrow$   $i=133$

$\boxed{1}$   $\downarrow$   $i=134$

$\boxed{1}$   $\downarrow$   $i=135$

$\boxed{1}$   $\downarrow$   $i=136$

$\boxed{1}$   $\downarrow$   $i=137$

$\boxed{1}$   $\downarrow$   $i=138$

$\boxed{1}$   $\downarrow$   $i=139$

$\boxed{1}$   $\downarrow$   $i=140$

$\boxed{1}$   $\downarrow$   $i=141$

$\boxed{1}$   $\downarrow$   $i=142$

$\boxed{1}$   $\downarrow$   $i=143$

$\boxed{1}$   $\downarrow$   $i=144$

$\boxed{1}$   $\downarrow$   $i=145$

$\boxed{1}$   $\downarrow$   $i=146$

$\boxed{1}$   $\downarrow$   $i=147$

$\boxed{1}$   $\downarrow$   $i=148$

$\boxed{1}$   $\downarrow$   $i=149$

$\boxed{1}$   $\downarrow$   $i=150$

$\boxed{1}$   $\downarrow$   $i=151$

$\boxed{1}$   $\downarrow$   $i=152$

$\boxed{1}$   $\downarrow$   $i=153$

$\boxed{1}$   $\downarrow$   $i=154$

$\boxed{1}$   $\downarrow$   $i=155$

$\boxed{1}$   $\downarrow$   $i=156$

$\boxed{1}$   $\downarrow$   $i=157$

$\boxed{1}$   $\downarrow$   $i=158$

$\boxed{1}$   $\downarrow$   $i=159$

$\boxed{1}$   $\downarrow$   $i=160$

$\boxed{1}$   $\downarrow$   $i=161$

$\boxed{$

که مقدار که دستم از دفعات کمتر است ←  $\leftarrow$  جمع جوابی کن

else if  $total < curr[i]$  که نهایتی کن  
وارد کردن کردن  
return  $rec(arr, total, i-1)$

else اگر نهایتی کن  
return  $rec(arr, total - curr[i], i-1) +$   
 $rec(arr, total, i-1)$   
که نهایتی کن و در نظر بگیر

پس که  $total$  را درست و اوارد کن و بخواهیں

def count\_sets\_dp(arr, total):  
mem = {} // empty dict or hash table  
return dp(arr, total, arr.length - 1, mem)

def dp(arr, total, i, mem):  
key = str(total) + ' ' + str(i)  
if key in mem:  
return mem[key]  
if total == 0:  
return 1  
else if total < 0:  
return 0  
else if i < 0:  
return 0  
else if total < curr[i]:  
to\_return = dp(arr, total - curr[i], i-1, mem)  
else:  
total - curr[i] ← جوابی کن  
total ← پرسی  
(dp(arr, total - curr[i], i-1, mem) +  
dp(arr, total, i-1, mem))

pair (جفت) از این دو عناصر پار

• 1 arguments

mem [key] = to - return

return to - return

memory

Time  $\propto$  تعداد ارجاعات  $\times$  میزان

فیکر call

call

$\Rightarrow$  (junks) [2, 4, 6, 10] جمعیت نیست

total = 10

(10) 10  $\times$  4  $\rightarrow$  40

Time =  $10 \times 4$

total  $\rightarrow$  مجموع ارجاعات

Time  $\propto$  total  $\times n \times 2 \times o(1)$

$\underline{o(n \times total)}$

$\underline{o(n \times o(1))}$   $\rightarrow$  مجموع ارجاعات

اولین ارجاع که در داده لورا

$n \times n \times total + 1 \times o(1)$

فیکر call

## SESSION 13

گیر  $v_1, v_2, \dots, v_n$

لی  $s_1, s_2, \dots, s_n$

پر کول علاوه  $\left( \begin{array}{l} \text{کو} \\ \text{ل} \end{array} \right)$

$S \subseteq \{1, 2, \dots, n\}$

$$\max \sum_{i \in S} v_i, \quad \sum_{i \in S} s_i \leq C$$

کوچکتر از زیاد نمایی  $(S \subseteq \{1, 2, \dots, n\})$  ①

$v_{i,C} =$  بزرگترین دارایی اول و

$\rightarrow O(n^2)$

$O(n^2) \rightarrow O(n^2)$  پر کول فریضی

پس از کوچکترین دارایی نمایی  $(S \subseteq \{1, 2, \dots, n\})$  ②

$v_{0,C} = 0$   $O(1)$

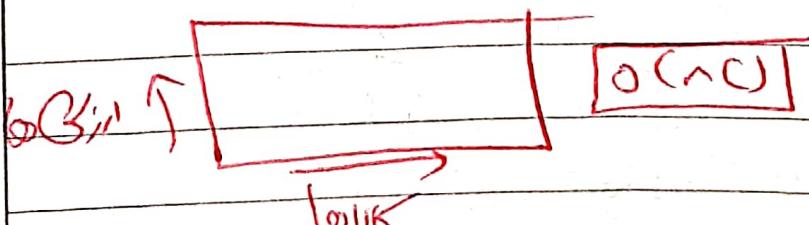
$$v_{i,C} = \max \{v_{i-1,C}, v_{i-1,C} + v_i\}$$

بزرگترین دارایی

+  $v_i$

$n-1$  (ستون) دارایی

$O(1) \rightarrow v_{n,C} \quad O(n)$  جمع ③



## Sequence alignment

(68%) (51%)

وادری و (نیاز)  $x \rightarrow y$   $\leftarrow$  (نیاز)  $y \rightarrow x$   $\leftarrow$  (نیاز)  $x \rightarrow y$   $\leftarrow$  (نیاز)  $y \rightarrow x$

لهم اجعلنا ملائكة حسنة [gap] (سورة ٢٠)

$\ell \rightarrow \min (S, \ell)$

four (four) (four)

$x + yap$  —   
 $y + yap$  — 

روی چالانه اور کے مکان

is  $\mu$  called  $\mu \rightarrow m$  ①

(ستقردات  $y' = y - kx_n$  و  $x' = x - kx_n y$ ) أك

Perd 0876 Sintex 636 per min x x per  
mm hr . x' x' (cm)

الله رب العالمين

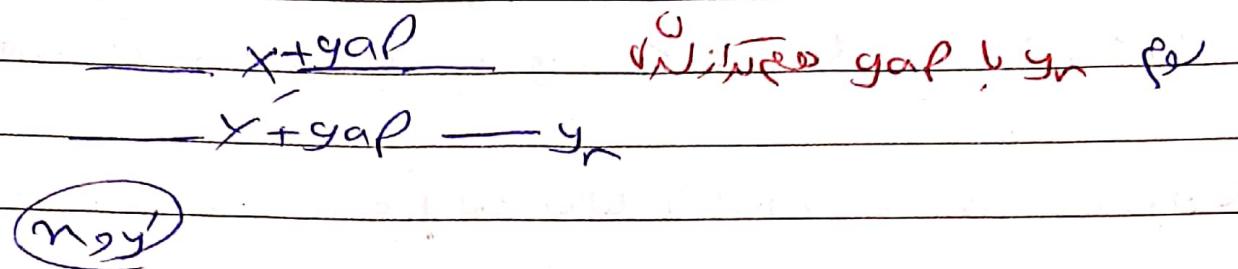
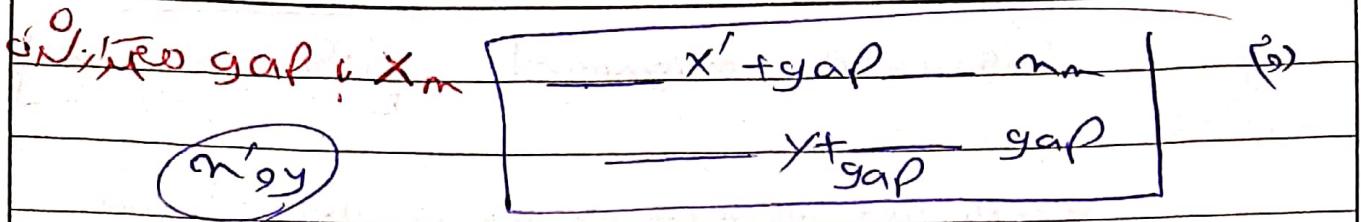
$p' < p$   $\alpha$   $\rightarrow$   $x, x'$   $\in$   $P'$   $\cup$   $P''$

$\text{X}_m\text{Y}_n = P' + \alpha$  ~~(for 1.5 types)~~

نبلد صغار.

1

16



عکس از اینجا

$$X = m_1 m_2 - \text{gap} \rightarrow \text{SIN Bar} \rightarrow \text{میانگین} \\ \text{نیز میانگین} \rightarrow \text{این نیز} . Y = y_1 y_2 - y_n$$

$y_1, y_2, \dots, y_n$  از اینجا میانگین  $x'$  است  $\text{SIN Bar} \rightarrow \text{۱}$    
 میانگین

gap,  $x_m \sim x, x' \sim$  ۲

gap,  $y_n \sim y, x \sim$  ۳

$$x' = x \text{ (نمای)} \quad y' = y \text{ (نمای)} \rightarrow \text{نمای}$$

$$x_i = m_1 m_2 \dots m_i \quad y_j = y_1 y_2 \dots y_j$$

$$P_{i,j} = \min_{j \in \{1, 2, \dots, n\}} x_j, x_i$$

$$P_{i,j} = \min_{j \in \{1, 2, \dots, n\}}$$

$$\text{۱} \quad P_{i-1,j-1} + \alpha_{\text{نمای}}$$

$$1 \leq i \leq m \quad \text{و SIN}$$

$$1 \leq j \leq n$$

hirmandpaper

$$\text{۲} \quad P_{i,j} + \alpha_{\text{gap}}$$

$$\text{۳} \quad P_{i,j-1} + \alpha_{\text{gap}}$$

$$P_{i,0} = P_{0,i} = i \cdot \alpha_{gap}$$

$$A = (m+1) \times (n+1) \quad (\text{size}) \text{ w.r.t}$$

for  $i=0$  to  $m$  do

$$A[i][0] = i \cdot \alpha_{gap}$$

ans[0]

for  $j=0$  to  $n$  do

$$A[0][j] = j \cdot \alpha_{gap}$$

for  $i=1$  to  $m$  do

for  $j=1$  to  $n$  do

$O(mn)$

$$A[i][j] =$$

$\alpha_{gap}$

( $\text{size}$ ) w.r.t.

min

$$A[i-1][j-1] + \alpha_{gap}$$

$$A[i-1][j] + \alpha_{gap}$$

$$A[i][j-1] + \alpha_{gap}$$

return  $A[m][n]$

Tallest Subproblem

SESSION 75 & 76

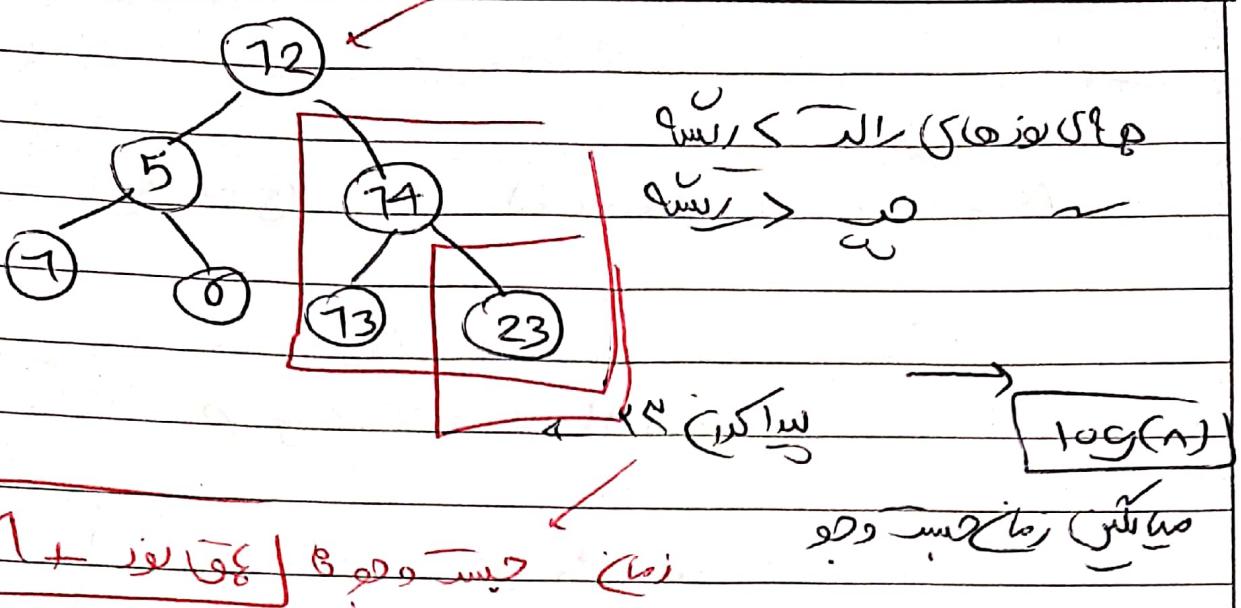
(Recursion) (size (size - 1))

size (size - 1), size (size - 1) ←

size (size - 1)

size (size - 1) ←

size ←



دستور کمینجی بینری درخت که عمقی و یا بالاتر  
درخت گویی

و درستی بینری رکورد های دیرگذشتی فرعی نامی

$$P_1, P_2, \dots, P_n \quad K_1, K_2, \dots, K_n \quad B \text{ (بروز)}$$

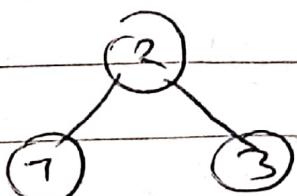
برای جستجوی باشیکی تابعی کاری کردن با این روش

$$\sum_{i=1}^n p_i \times ( \text{جستجوی } K_i )^T$$

$T = \max_{i=1}^n K_i$

$T = 2, K_i = 6$

$$T = 2, K_i = 6 + 1$$



$$2 \times 0/1 + 1 \times 0/1 + 2 \times 0/1 =$$

$$7/2$$

$$K_i = 1, 2, 3$$

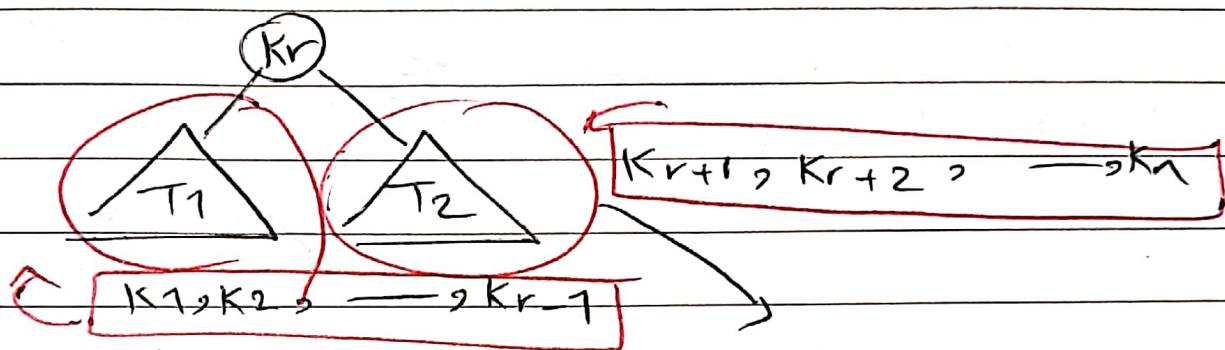
$$p_i = 0/8, 0/7, 0/7$$

- 1
- 2
- 3

$$7 \times 0.8 + 0.7 \times 2 + 0.1 \times 3 =$$

٧(٠٨) + ١(٢) + ٣(٠١) = ٧.٣

لهم حمد لله رب العالمين وبارئه رب كل كائن (س)



لهم حمد لله رب العالمين

لهم حمد لله رب العالمين (س)

Kr-1 تKr (س)

Kr ت Kr+1 ا

wq, j =  $\frac{w_{j-1}}{K_{j-1}} \dots \frac{w_1}{K_1}$   
زمانی دارای مجموع مصالحی در مجموع زمانی دارای مجموع مصالحی

(جواب) دارای مجموع مصالحی دارای مجموع مصالحی  
و فضای پر را  $P_1, P_2, \dots, P_r$  فضای پر را  $P_{r+1}, P_{r+2}, \dots, P_n$

و  $P_{r+2}, \dots, P_r$  دارای مجموع مصالحی دارای مجموع مصالحی  
نیز دارای مجموع مصالحی دارای مجموع مصالحی ۱  
و فضای پر را  $P_1, P_2, \dots, P_{r-1}$  فضای پر را  $P_r$

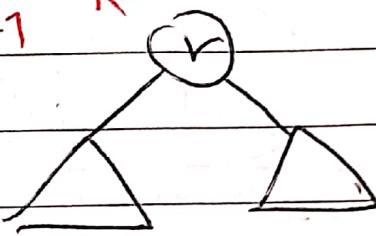
$P_{r+1}, \dots, P_n$  دارای مجموع مصالحی دارای مجموع مصالحی ۲  
و فضای پر را  $P_{r+1}, \dots, P_n$  دارای مجموع مصالحی دارای مجموع مصالحی

$w_{i,j}^0 = \min_{0 \leq r \leq j} (w_{i,r-1} + w_{r+1,j})$

$$w_{i,j}^0 = \sum_{r=1}^j w_{i,r-1}$$

$$w_{i,j}^0 = \min_{r \in \{1, \dots, n\}} (w_{i,r-1} + w_{r+1,j}) + \sum_{k=1}^r p_k$$

for  $i \in \{1, \dots, n\}$ ,  $j \in \{1, \dots, n\}$



$$A = (n+1) \times (n+1)$$

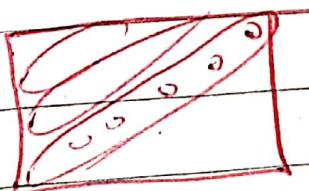
$$\text{for } i=0 \text{ to } n \text{ do } i^0 = j^0 + 1$$

for  $i=1$  to  $n+1$  do

$$A[i][i-1] = 0$$

for  $s=0$  to  $n+1$  do

for  $i=1$  to  $n-s$  do  
     $A[i][i+s] = \min_{k=i}^{i+s} (A[i][k-1] + A[k+1][i+s])$



$$\sum_{k=i}^{i+s} p_k + \min_{r=i}^{i+s} (A[i][r-1] + A[r+1][i+s])$$

now step 5:

return  $A[1][n]$

caser

1 2 3 4

$\frac{3}{8}$   $\frac{3}{8}$   $\frac{1}{8}$   $\frac{1}{8}$

8 Jus

$\frac{1}{8} \times 5$   $\text{Ans}$   
 $(5 \times 8) = 40$

4	$\frac{7}{8}$	1	$\frac{3}{8}$	$\frac{1}{8}$	0
3	$\frac{7}{8}$	$\frac{5}{8}$	$\frac{1}{8}$	0	
2	$\frac{2}{8}$	$\frac{3}{8}$	0		
1	$\frac{3}{8}$	0			
0	0				

$O(n^2)$

j ↑

1 2 3 4

$$A[1,2] = (\cancel{\frac{3}{8}} \leftarrow \cancel{\frac{3}{8}}) + \textcircled{1}$$

$$m = \min \left\{ \begin{array}{l} A[1,1] + A[3,2] \\ A[1,0] + A[2,2] \end{array} \right. \rightarrow m = \frac{3}{8}$$

$$= \frac{3}{8} \times 3 = \textcircled{\frac{3}{8}}$$

$\frac{3}{8}$

## Bellman-Ford Algorithm

## # Single source shortest path

## # graph algorithm

## Dijkstra vs Bellman Ford

⑦ dijkstra and bellmanford doesn't find shortest path for graphs with negative edge weight cycle.

اگر دو صفحے دال کروں تو

مکان سخت

② dijkstra can't detect if graph has negative edge weight cycle while bellman ford

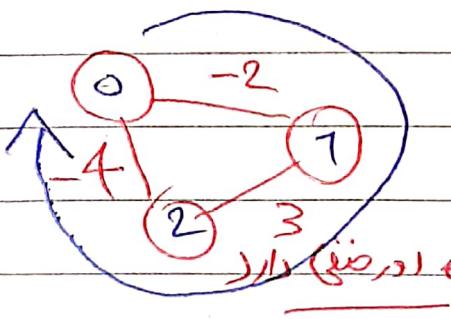
Can

وَلَا يَرْجِعُونَ مِنْ قَبْلِ دِرْلَكْ

طراز bellman-ford

درانی صورت برپا گردد که گراف (وصف) دارکوب کوچکتری صنعت را

مکانیزم

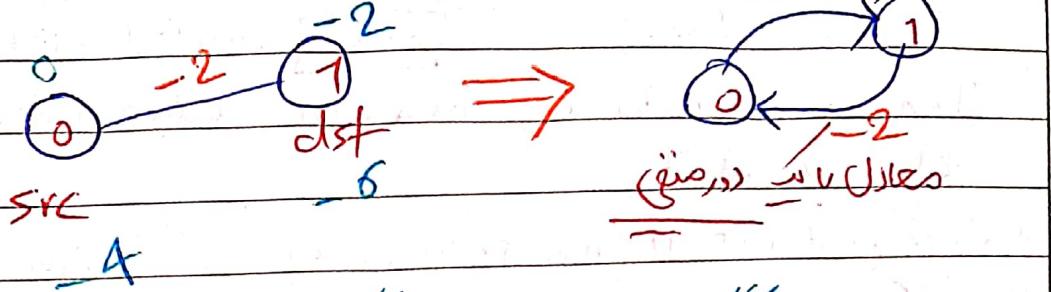


$$-2 + 3 - 4 = \boxed{-3}$$

$T$  negative edge weight sick

الآن نحن في المراحل الثالثة والرابعة bellman-ford

undirected graph



directed  
graph  
-2

اگر مسیر میں چوڑا ہے تو (دیکھ بجہ) جو کوئی مسیر کا انتہا نہیں ہے اسے مسیر کا انتہا کہا جاتا ہے۔ مثلاً مسیر 0 → 1 → 2 → 0 کا انتہا 0 ہے اور مسیر 0 → 1 → 2 → 3 → 0 کا انتہا 3 ہے۔

wiki (یہ انتہا کو دیکھ بجہ) میں اسے انتہا کہا جاتا ہے۔ note!!

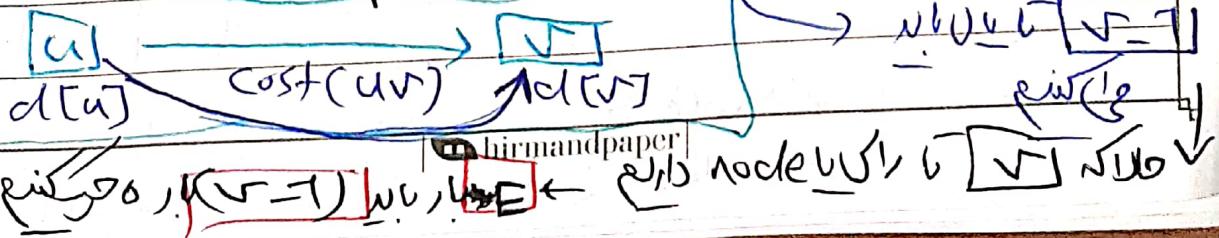
\* bellman ford doesn't find shortest path for undirected graph if it has any negative edge

(یہ اسی bellmanFloyd کا مطلب ہے کہ اگر کوئی مسیر میں چوڑا ہے تو اسے دیکھ بجہ کر دیں) **Bellman Ford Algo**

① initialize all the distance values as  $\infty$  except source

② repeat  $(V-1)$  times

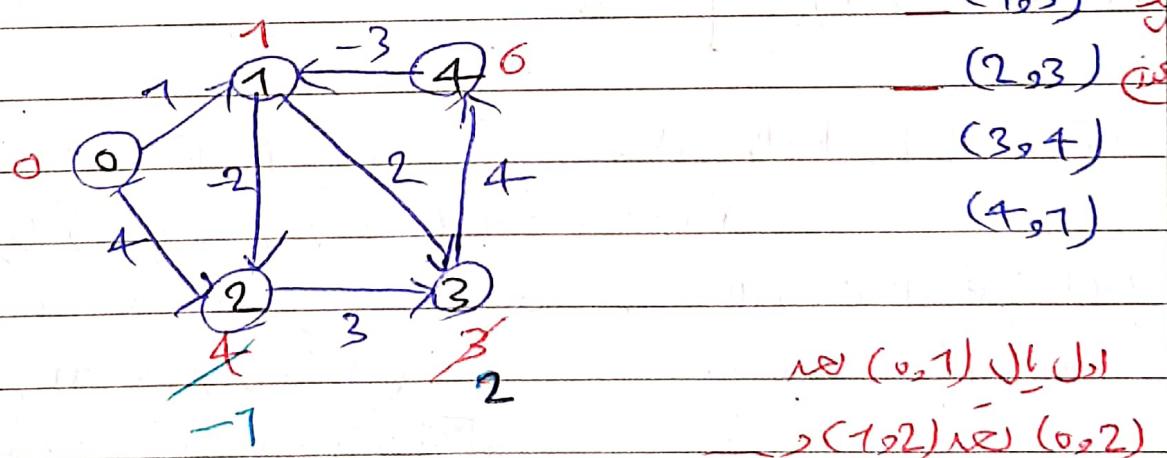
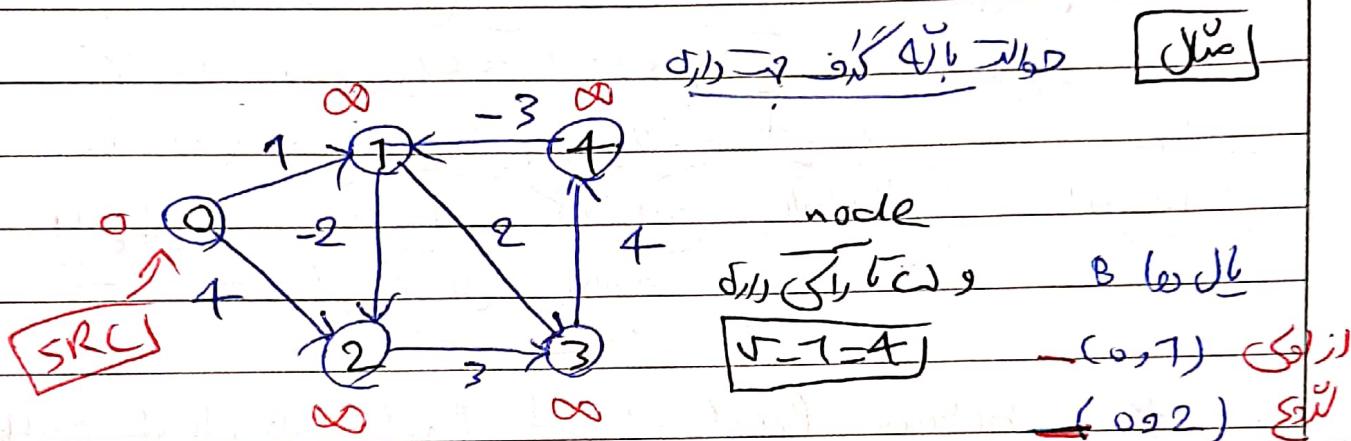
(Step 1) if  $d[u] + \text{cost}(uv) < d[v]$  then update  $d[v]$   
vertices (edge) else skip



③ Relax all the vertices one more time.

if we find any new shortest distance value  
then → we have negative edge weight cycle  
else we don't

لے سکتے ہیں اس کا نام مکانیکی صورت ہے۔



First relax

<u>النهايات المقتضبة</u>	0	1	2	3	4
$\frac{1}{x}$	0	1	-7	2	6

وَالْمُؤْمِنُونَ هُمُ الْمُفْلِحُونَ

$\leftarrow$  ~~reduces~~  $\rightarrow$  (6) Nach ~~reduces~~ iteration  $\bar{y}$  ist  $\bar{x}_1, \dots, \bar{x}_n$

جرون مارش (Marsh)  $\rightarrow$   $(V-1) \times E$

time complexity  $\in O(VE)$  Bellman Ford

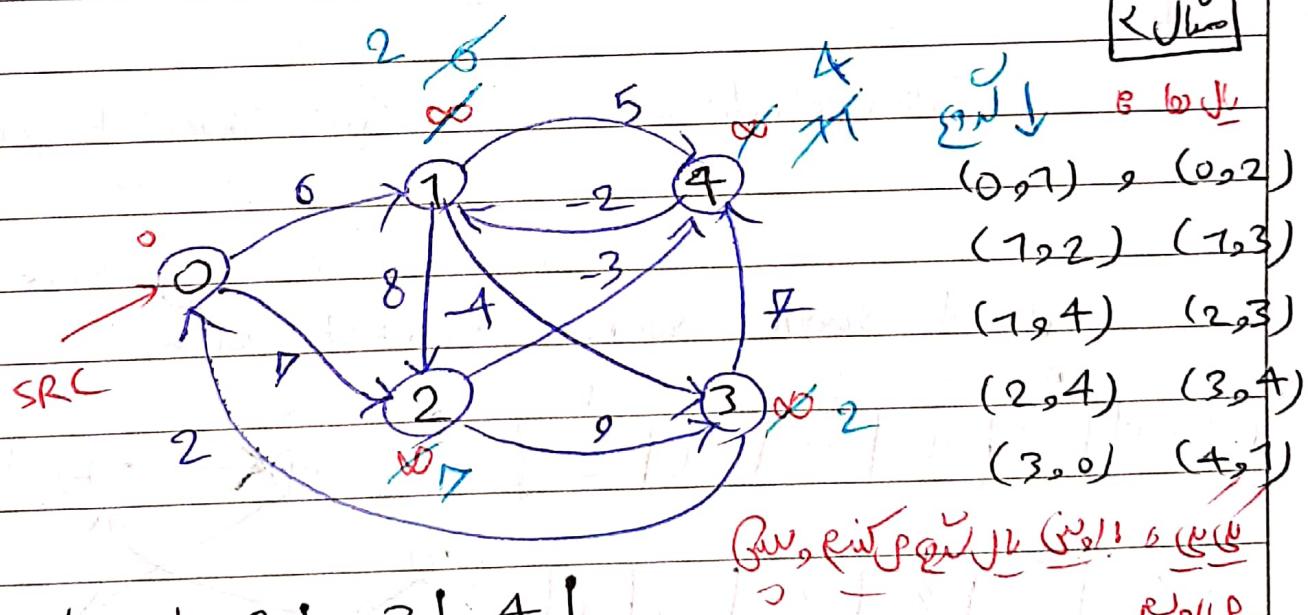
$\in O(E \log V)$  Dijkstra

Dijkstra (العاشر)  $\in O(V^2)$  Dijkstra  $\leftarrow$  يعتمد على كيفرن، وهو كالصنف الثاني في جون خطي.

Bellman Ford  $\leftarrow$  جون خطي

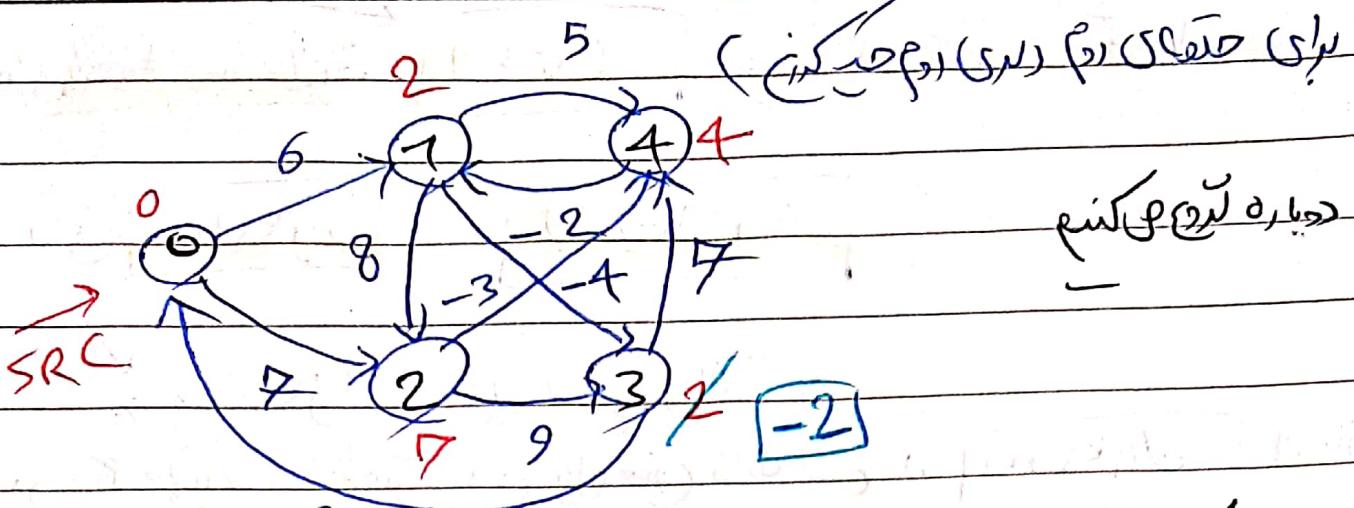
Relax call  $E$  edges  $\in (V-1)$  times

$(E) \times (V-1)$



0	1	2	3	4
0	$\infty$	$\infty$	$\infty$	$\infty$
0	2	7	2	4
0	2	7	-2	4

First Parent 0 1 2 3 4  
 (3) Cust 0 5 7 2 17  
 hirmudpaper 2



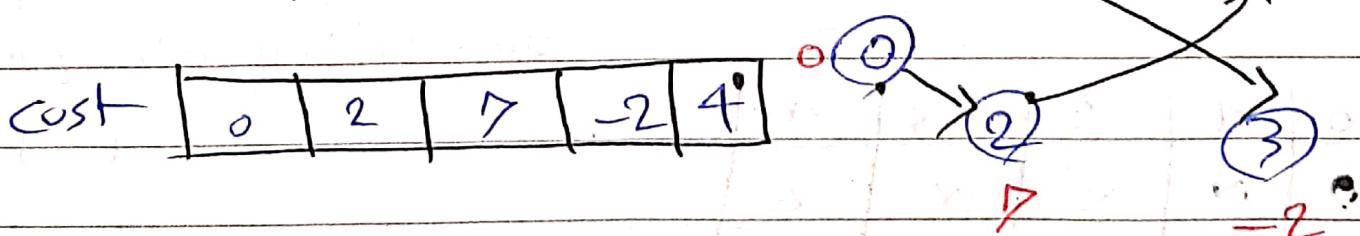
	0	1	2	3	4
parent	-1	0	0	1	2
cost	0	6	7	2	11

int(f(Src) to Node) (ijkstra(SN) for shortest)

Initial update → previous shortest path

Print single source shortest path

	0	1	2	3	4
parent	-1	4	0	1	2
cost	0	2	7	-2	4



```
void bellmanFord( vector<edges> &Edges) {  
    }
```

```
    int parent[v]; // store shortest path structure  
    int cost_parent[v]; // cost of the node to  
    vector<int> value(v, INT_MAX); parent  
    // keeps shortest path values to edge weight  
    // each vertex from source
```

```
    parent[0] = -1;
```

```
    value[0] = 0;
```

```
// include (v-1) edges to cover all v-vertices
```

```
    bool updated;
```

```
    for(int i=0; i<v-1; i++)
```

```
        updated = false;
```

```
        for(int j=0; j<E; j++)
```

```
            int u = Edges[j].src;
```

```
            int v = Edges[j].dst;
```

```
            int wt = Edges[j].wt;
```

```
            if( value[u] != INT_MAX &&
```

```
                value[u] + wt < value[v] )
```

```
                value[v] = value[u] + wt;
```

```
                parent[v] = u;
```

```
                cost_parent[v] = value[v];
```

```
                updated = true;
```

```
            }
```

```
            if(updated == false)
```

```
                break;
```

If now check by relaxing once more time if we have a negative edge cycle

for (int j=0; j < E (if updated == true; j++))

int u = Edges[j].src;

int v = Edges[j].dst;

int wt = Edges[j].wt;

if (value[u] != INT\_MAX and  
value[u] + wt < value[v])

cout << "Graph has negative cycle";

return;

If print shortest Path

for (int i=1; i < V; i++)

cout << "U->V is " << parent[i] << " -> " << i <<

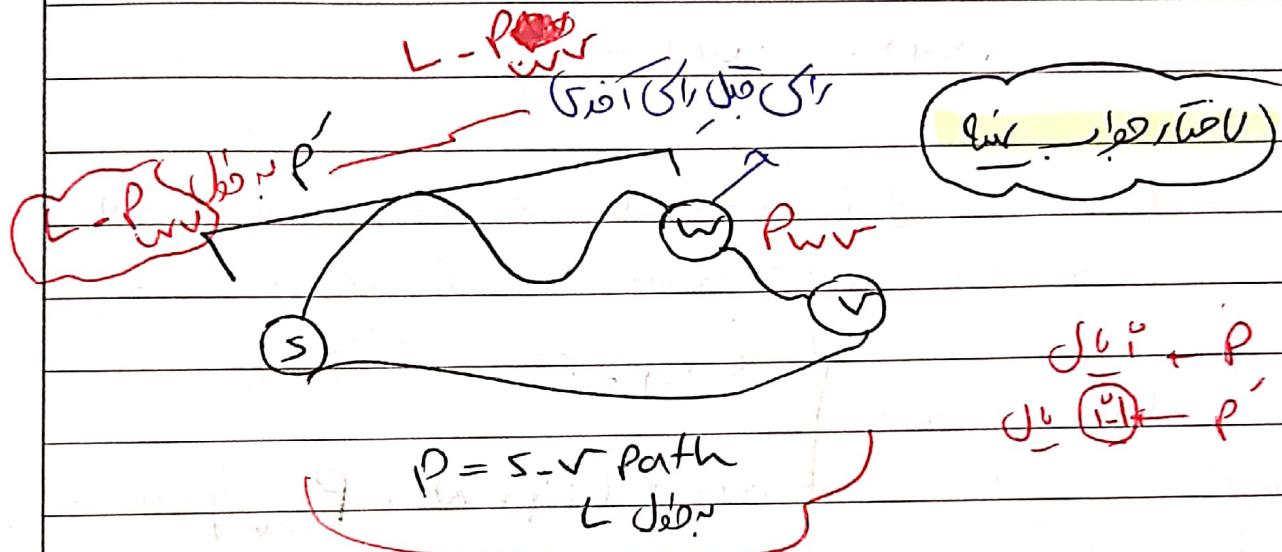
" cos to reach " << parent[i] << " from source" <<

value[i] << "\n";

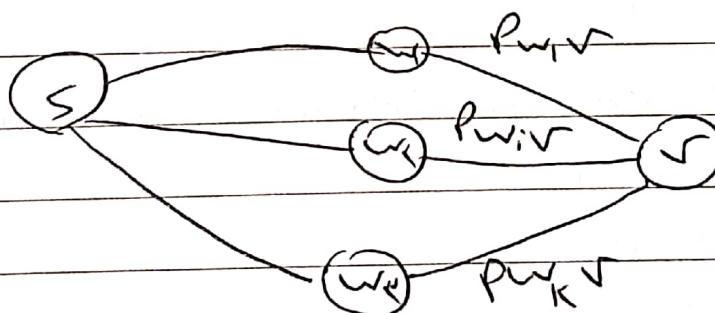
Bellman forced  $\rightarrow$  university

و کاف و سر و بای هم میل داشتند و مجموعه ای می خواستند که همه این میل ها را در یک مجموعه جمع کنند.

هرو ۸ کوچکترین قابل نبایر را از منبع  $\pi$  برداشت کنیم گرفت (۱۵)



L پر کوئاں اور صیوان کے سال



$$S_{\text{min}} = \min_i \left( \frac{\text{air} - \bar{v}_i}{S_{\text{air}}} + L_{\text{wind}} \right)$$

(ووضع) کندارم و اگر خود را کنده  
از هر دفعه که کوتاه تری صورت داشته باشد  
زیرا مسافتی که کوتاه تری صورت داشته باشد  
از دفعه دیگر باشد

(راهنمایی دریافتی)  $\rightarrow$   $w \in \mathbb{R}$  می باشد و  $w = \sum_{i=1}^n w_i x_i$  است که  $x_i \in \mathbb{R}^d$  می باشد. این را می توان به صورت  $w = \sum_{i=1}^n w_i x_i$  نوشت.



$$L_{i,v} = \min_{\substack{w \in V \\ (i,w) \in E}} L_{w,v} + P_{wv} \quad \text{②} \Rightarrow$$

$L_{i,v} = \min_{\substack{w \in V \\ (i,w) \in E}} L_{w,v} + P_{wv}$

$L_{i,v} = \min_{\substack{w \in V \\ (i,w) \in E}} L_{w,v} + P_{wv}$

$L_{i,v} = \min_{\substack{w \in V \\ (i,w) \in E}} L_{w,v} + P_{wv}$

$A_6 = (n+1) \times n$  two dimensional array

base case ( $i=0$ )

$$A[0][s] = 0$$

for each  $\sqrt{f_3}$  do

$$A[0][\sqrt{ }]=+\infty$$

for i=1 to n do

if  $\text{ATi}[v] \neq \text{for } v \in V \setminus \{v\}$  stable = True

$\Delta \subseteq \{1\} \cup \{j\}$

`stable = false`

if (stable)

return A[i-1][v]

 hirnandpaper

A[i][j] =  hirmandp

$$A[i][j] \cdot 0 =$$

hirmandpaper

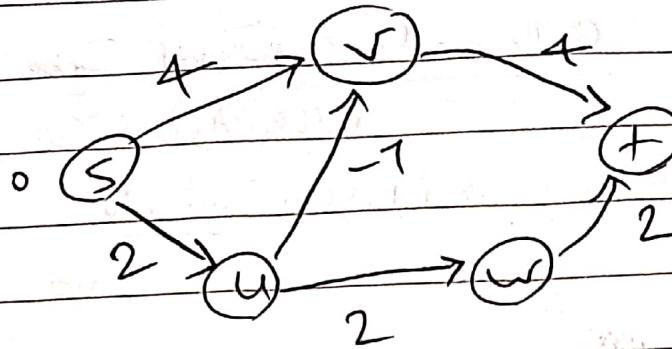
$\min_{\mathbf{A}} \mathbf{A}(\mathbf{i}, \mathbf{j}) \mathbf{J}(\mathbf{F})$

—  
—

case 1

1995 -

## Case 2



# بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

مکالمہ

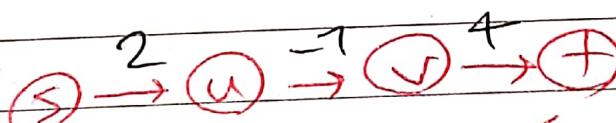
۱۰۸

51/1

	-	-	v	v	v	+	+∞	+∞	8	5	5
w	--	u	u	u	w	+∞	+∞	+∞	4	4	4
u	-	s	(s)	s	u	+∞	2	2	2	2	2
v	-	s	u	(u)	u	v	+∞	4	1	1	1
s	-	-	-	-	-	s	0	0	0	0	0
	0	1	2	3	4		0	1	2	3	4

$$A_{1,0}S = \min_{\{0,1\}} \{A_{0,0}, S\}$$

$$A_1, \sqrt{-\min} \left( \begin{array}{c} A_{0,2} \\ A_{0,3} \end{array} \right) + \begin{pmatrix} 4 \\ -5 \end{pmatrix}$$



$$2 - 1 + 4 = \boxed{5}$$

$\circ(n) \text{ for } \{ \}$

زمانی احتمال

o(n) for( ) K

$$\min \{ \text{cost}_1, \text{cost}_2 \}$$

$$\frac{O(n \times n \times n)}{8(n^4)}$$

9

نحوه وارد و نزدیک

$$\text{Time Complexity: } O(n^m) \times \sum_{r \in R} \sqrt{\log(\log(r))} = O(nm)$$

مذکور شده است که در اینجا بحث در مورد خودکار و اتوماتیک راهنمایی (Bellman Ford) می‌شود.

برای کمینه کردن مسافت میان نodes از اینجا پس از معرفی مفهوم Routing table می‌باشد که در اینجا معرفی شد.

### Dijkstra Algorithm

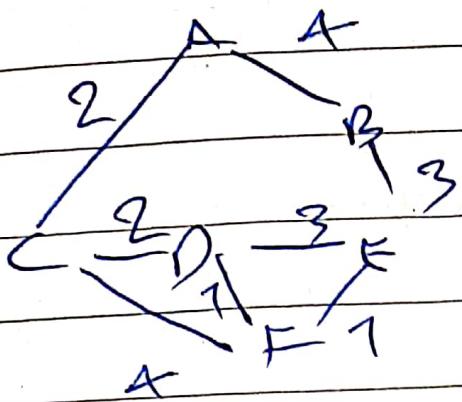
۱) در (جهد کمینه کردن) مسافت  $\rightarrow$  دور جای را ویند کنم و دور  $\rightarrow$  می‌گشتم که کمترین مقدار از اولین دوری که ویند کردم را اطلاع داشتم.

۲) وقتی که رستم لاین (دوری) که قراره ویند کنم و رکھم  $\rightarrow$  مساله می‌گشتم.

Start node

۳) باید دور (دور چشم)  $\rightarrow$  فاصله ای این دور را از اولین دور ۰ باقی بگذارم (یا لاین) که بادن دور (دور چشم) می‌گشتم.

۴) اگر کافیست از زور کردن آگرین کمتر از فاصله (یا لاین) صیغه کوچه را باید دور همینه کنم.



وے جو E, A کی مسافت (جنہیں)

pick the smallest  $\rightarrow [A]$

visited

[A, C, B, D, F, E]

vertex	shortest dist from A	previous
A	0	
B	$\infty$ 4	A, B null
C	$\infty$ 2	B, C A
D	$\infty$ 4	C, null A
E	$\infty$ 7 6	D, null C
F	$\infty$ 6 5	E, null B - F, null C, D

y

visited ہی تو پڑھ کر ادن (ورک) کی میز کر دیں

پہلی (جس کی) 'x' نام

وے جو E, A کی مسافت (جنہیں)

A  $\rightarrow$  B

(previous next, previous E, A)

E  $\rightarrow$  F  $\rightarrow$  D  $\rightarrow$  C  $\rightarrow$  A  $\rightarrow$  [6]  
فہاں

کوچکتری فاصله از مسیر

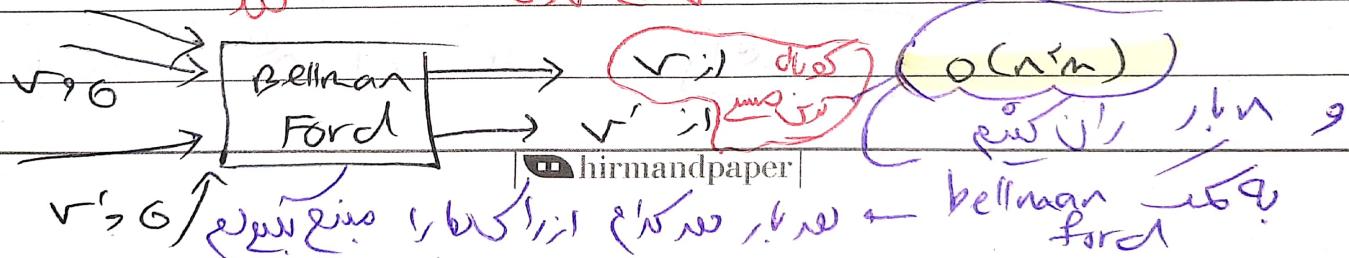
G(R)  $\rightarrow$  کافی نیست در مواردی هم که  $G = \{v\}$

dist(u, v)

کوتاهتری فاصله می بیند و می خواهد

کوتاهتری

کوتاهتری



نادر رایج

$O(n^2)$

رایج

$O(n^2)$

نادر

hirmandpaper

رایج

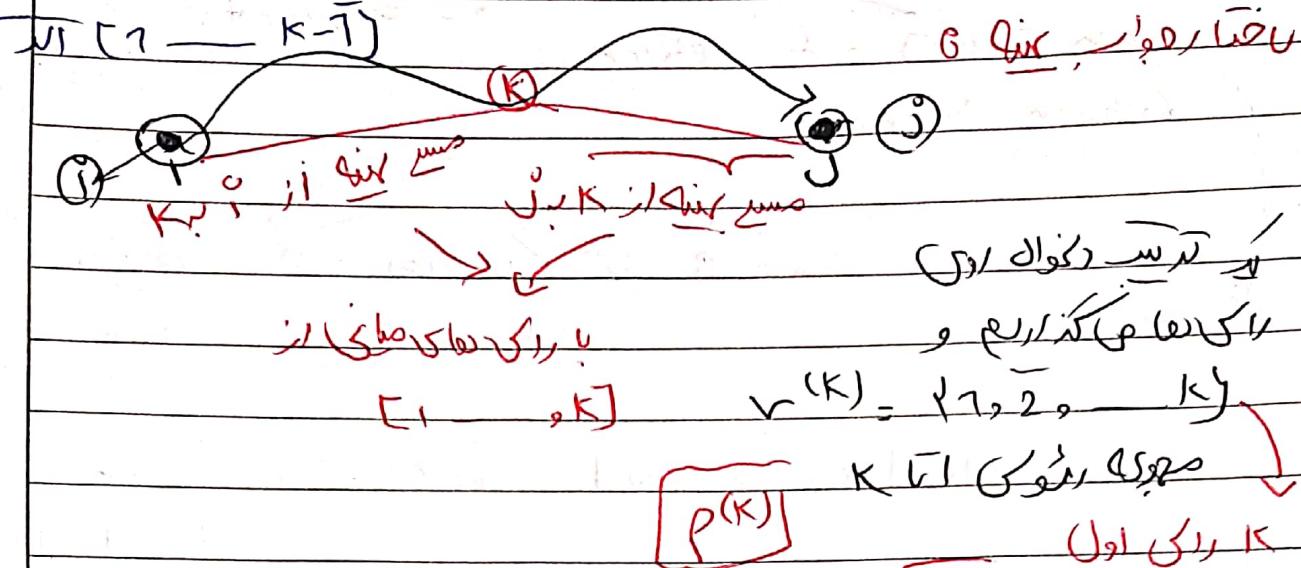
$O(n^2)$

نادر

$O(n^2)$

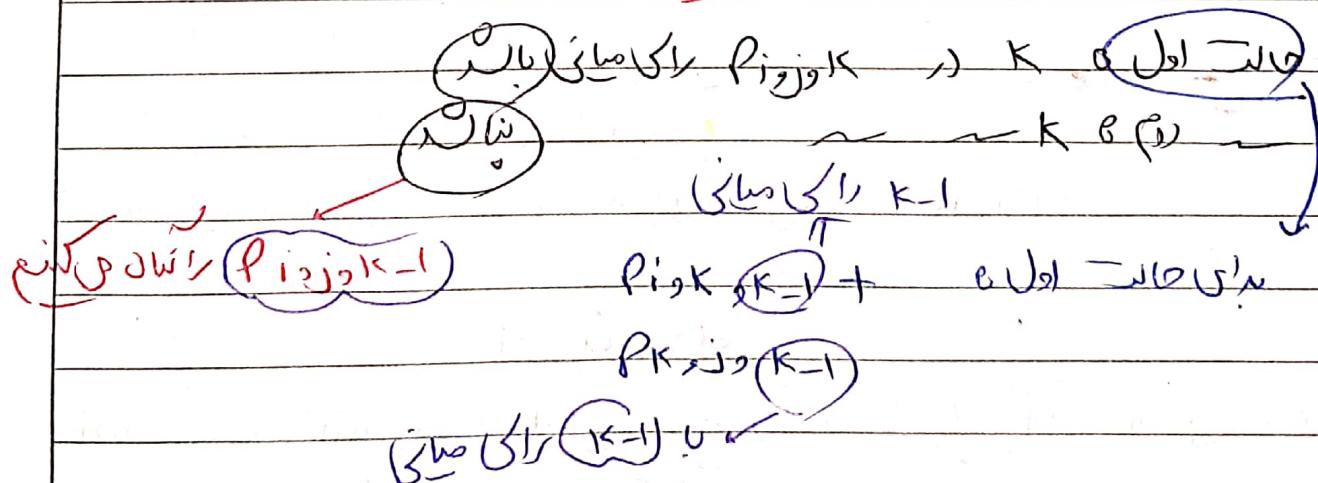
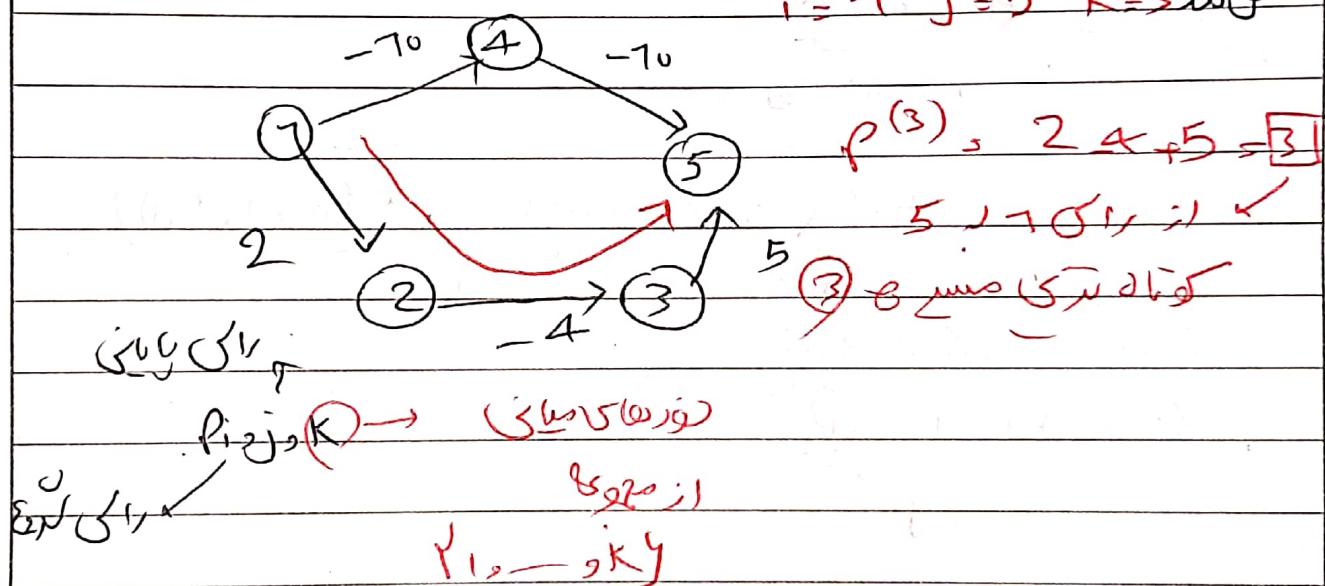
رایج

۷ بانفر (ورصف) نهایی از



مثال ۸ کوتاه ترین مسیر که تنها ۱۰ رند اول به عنوان راک میانی اتفاق اف

$$i = 1, j = 5, k = 3$$



$$O(n^k) \leftarrow v_{i,j,k} \leq n^k$$

## Floyd-warshall الورس

$A_{ij}^k$  هي المسافة بين  $v_i$  و  $v_j$  باعتماد مسار من خلال كوهنرني

$$A_{ij}^k = \underbrace{\text{محل كوهنرني}}_{\text{مس}}$$

$$A_{ij}^{i,j,0} = \begin{cases} 0 & i=j \\ \infty & (ij) \notin E \\ +\infty & (ij) \in E, i \neq j \end{cases}$$

$$A_{ij}^k = \min$$

$$A_{ij}^{i,j,k-1}$$

$$A_{ij}^{i,j,k-1} + A_{kj}^{k,j,k-1}$$



input directed graph  $G = (V, E)$  in adjacency list or adjacency matrix representation.

and a real-valued length  $l_e$  for each edge  $e \in E$

output dist( $v, w$ ) for every vertex pair  $(v, w) \in V$   
or a declaration that  $G$  contains a negative cycle.

$v = \{1, 2, \dots, n\}$

$O(n^2)$

$K$  subproblems ( $K$  indexed from 0,  $v, w$  from 1)

$\boxed{U(v,w); K}$

$\boxed{1; 1 w \text{ or}}$

base case ( $K=0$ )

for  $v = 1$  to  $n$  do

for  $w = 1$  to  $n$  do

if  $v=w$  then

$$A[0][v][w] = 0$$

else if  $(v, w)$  is an edge of  $G$  then

$$A[0][v][w] = L_{vw}$$

else

$$A[0][v][w] = +\infty$$

for  $K=1$  to  $n$

for  $v = 1$  to  $n$

for  $w = 1$  to  $n$

$$A[K][v][w] =$$

$$\min(A[K-1][v][w], A[K-1][v][K]$$

$$+ A[K-1][K][w])$$

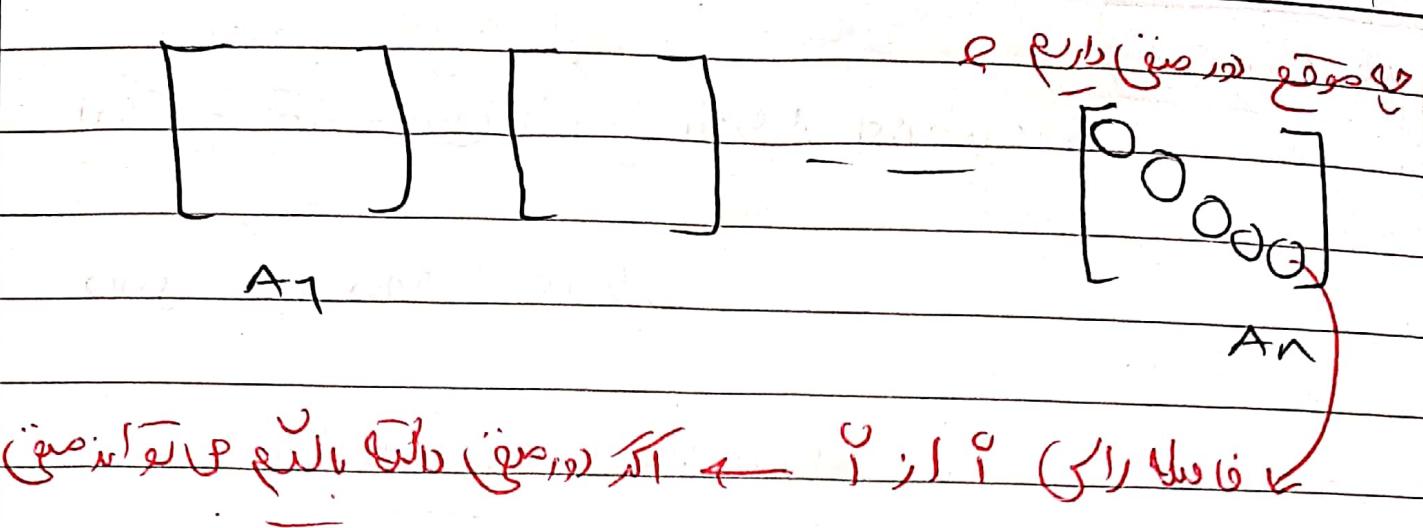
for  $v = 1$  to  $n$

~~if  $A[n][v][v] < 0$~~

~~return "negative cycle"~~

return  $\{A[n][v][w]\} \quad v, w \leftarrow V$

~~PS, PC~~



(٩) (مروج)

Bellman Ford یا پرسکو (پرسکو) کوئنٹری کسی شے پر میں کیا

لهم انت خلي اذنی (عاشر) کے لئے تبریزی (تاریخ) نہ دفع

وهي اى صناعة بـ صناعي وتجهيز اى اسلوب راي  
الساده كتب (توصيم از اول (دباهه حصار کتب))

$$v^{(K)} = \sqrt{K} \cdot \underbrace{\frac{1}{\sqrt{K}} \sum_{k=1}^K v_k}_{\text{میانگین مقدار}} \quad \downarrow$$

## NP-completeness

指的是许多计算问题的集合，如果其中一个问题是NP完全的，则所有其他问题都是NP完全的。

Turing halting problem

given a program and an input, whether the program will eventually halt when run with that input or will run forever.

halting problem (whether a given program will stop given a particular input)

### NP complete

are problems whose status is unknown.

No polynomial-time algorithm has yet been discovered for any NP complete problem, nor has anybody been able to proof that no polynomial-time algorithm exist for any of them.

( $\leq$ ) is NP-complete if it can be reduced to another problem in polynomial time.

P

deterministic Turing machine

can solve all NP-complete problems in polynomial time.

(NP)

→ We interested in decision problems (L<sub>dec</sub>)  
Non-deterministic Turing machine

Polynomial (b)  $\rightarrow$  جملہ

- NP (L<sub>dec</sub>)  $\neq$  P  $\neq$  NP  $\star$
- (b) 1) deterministic machine جو ایسے کام کر سکے  $\star$
  - (b) 2) non-deterministic جو ایسے کام کر سکے polynomial machine (2) جو Polynomial

(NP-complete)

are the hardest problems in [NP] set.

a decision problem L is NP-complete if

①  $L \in NP$

وہ NP-complete ہے جو (L) کو حل کر سکے  
(L) کو حل کرنے والے کم تر ہے لیے

② every problem in [NP] is reducible to L

in Polynomial time

$L \rightarrow$  NP-complete  $\rightarrow$  L

$L \in NP \Rightarrow L \rightarrow$  L ①

$L \rightarrow$  is the prob L  $\rightarrow$  NP  $\rightarrow$  others ②

Polynomial

(NP-hard)

کی تا NP-hard ہوں گے

(نیوردیکل) یعنی، وہ سچے کوئی

ٹھوڑا کم راستے کی وجہ سے

NP-hard  $\rightarrow$  جو B Turing halting problem

NP  $\rightarrow$  جو B vertex covering problem

(P  $\neq$  NP)

جو & shortest path problem

Decision vs optimization problems

discussing the difficulty of decision problems  
is often really equivalent to discussing the  
difficulty of optimization problems

e vertex cover problem

sized vertex set

is, is, is, is (سچے کوئی نہیں)

optimization

(S<sub>1</sub>, S<sub>2</sub>)

e (جواب) decision

(S<sub>1</sub>, S<sub>2</sub>)

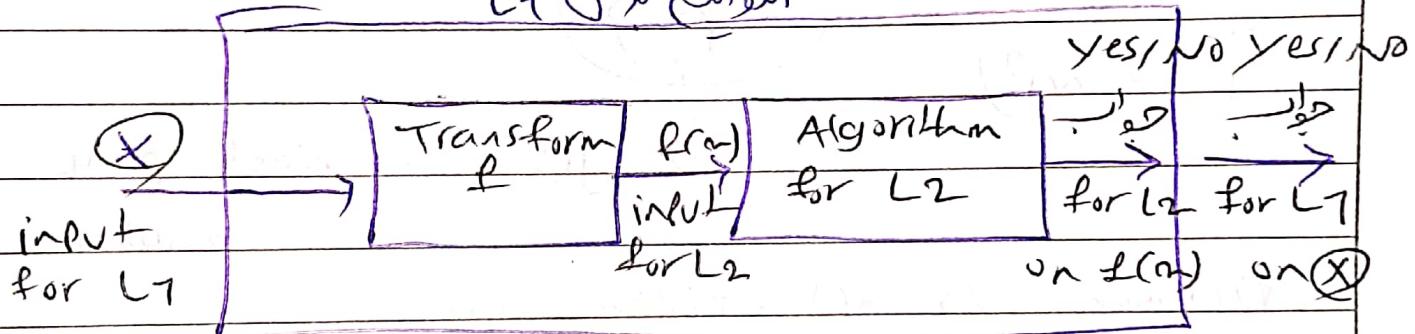
is, is, is, is (سچے کوئی نہیں)

e  $\boxed{K}$  (جواب) ہے

## Reduction

اگر  $L_2$  یک decision problem باشد و  $L_1$  یک many-one reduction را از  $L_2$  به  $A_1$  بدهد، آنگاه  $A_1$  کمتر نیست از  $L_1$ .  
 این ایشان را reduction می‌نامند.  
 این ایشان را many-one reduction می‌نامند.

(عن تعریف  $L_1$  کمتر از  $L_2$  است)  $L_1$  کمتر از  $L_2$  است اگر  $L_1$  را می‌توان بجهت  $L_2$  حل کرد.



حذا را NP reduction می‌نامند.

reduction که آن  $L_1$  کمتر از  $L_2$  است را many-one reduction می‌نامند.  
 این آن باید حل مسائل  $L_1$  را حل مسائل  $L_2$  را باید حل کرد.

حذا را NP-completeness که reduction است.

$L_1$  را NP-complete می‌نامند اگر  $L_1$  کمتر از  $L_2$  باشد.

کمتر از  $L_2$  باشد.

و  $L_2$  NP-complete باشد و  $L_1$  کمتر از  $L_2$  باشد.

و  $L_1$  کمتر از  $L_2$  باشد.

the class  $[NP]$  is the set of all functions  $f$  for which, given any  $m$  and  $y$ , you can check in polynomial time a whether or not  $f(m,y)$  is true

exists  $(P \leq K)$   $\exists$

exists  $(L \leq K)$   $\exists$

the class  $P$  is the set of all functions  $f$  for which, given any  $m$  you can find in polynomial a value  $y$  for which  $f(m,y)$  is true

\* all functions in  $P$  are also in  $NP$ \*

$P \subseteq NP \rightarrow \text{Co-NP}$

### ① shortest path

given weighted graph  $G$ , nodes  $s$  and  $t$  in  $G$ , and value  $K$ , is there a path  $p$  from  $s$  to  $t$  such that  $\text{weight}(p) \leq K$

### ② traveling salesperson

given completed weighted graph  $G$  and value  $K$ , is there a hamilton cycle  $c$  visiting every node in  $G$  such that  $\text{weight}(c) \leq K$ ?

### ③ minimum spanning tree.

given weighted graph  $G$  and value  $K$ , is there a spanning tree  $t$  such that  $\text{weight}(t) \leq K$

#### ④ vertex cover

given graph  $G$  and value  $K$  is there a vertex cover  $S$  for  $G$  such that  $|S| \leq K$ ?

exist or not

these are  $\boxed{NP}$  because we can check in polynomial time

① whether  $P$  is a path in  $G$  with weight  $\leq K$

②  $\sim \sim \sim \sim$  cycle  $\sim$  visiting every node and has weight  $\leq K$

③  $\sim +$  is a spanning tree of  $G$  with weight  $\leq K$

④ whether  $S$  is a vertex cover of  $G$  and has  $\leq K$  nodes

## SESSION 27

ویری (V) کا، تینی (T) کا، زمکی (Z) کا، باریل (B) کا، زمکی (Z) کا، ویری (V) کا،

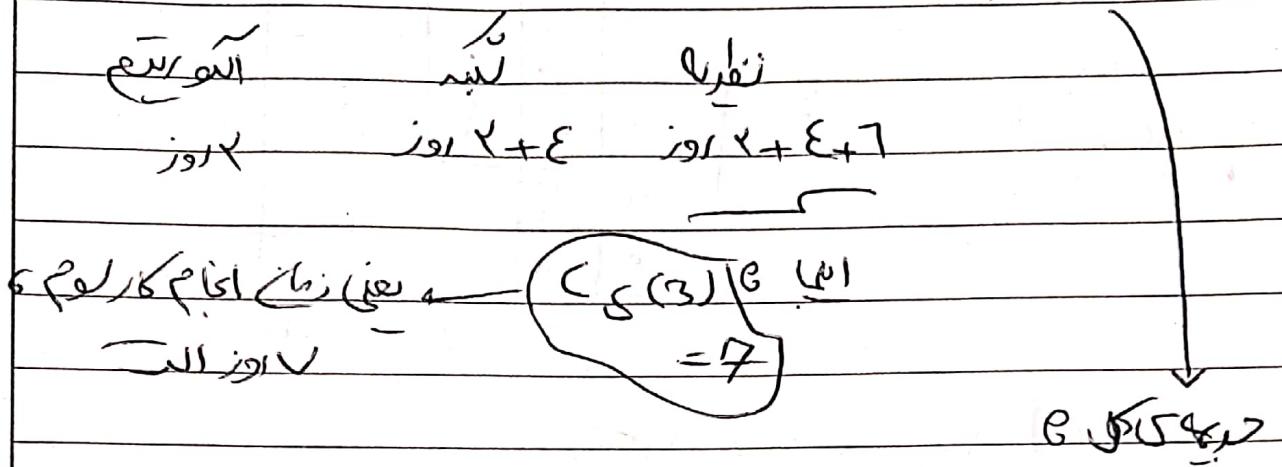
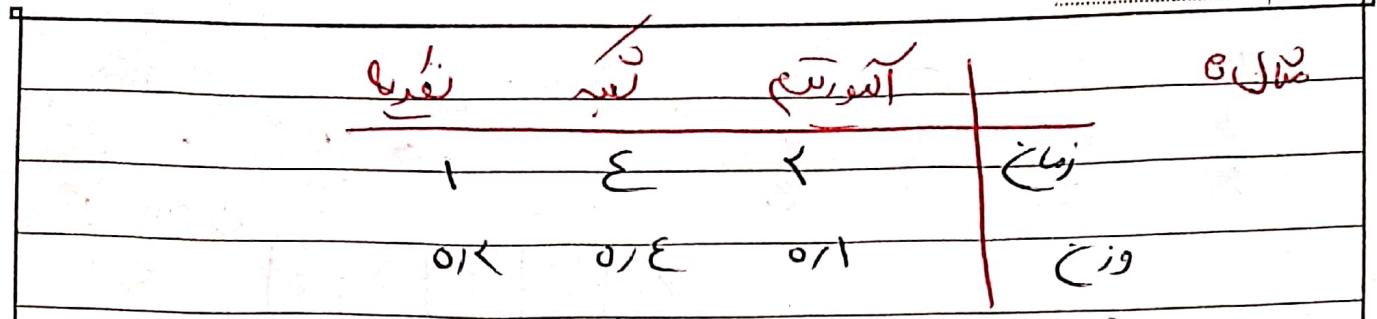
ویری (V) کا، تینی (T) کا، زمکی (Z) کا، باریل (B) کا، زمکی (Z) کا، ویری (V) کا،

$$\min \sum_{j=1}^n w_j C_j(s) \rightarrow \text{زمکی (Z) کا رام کرنا}$$

ویری (V) کا، تینی (T) کا، زمکی (Z) کا، باریل (B) کا،

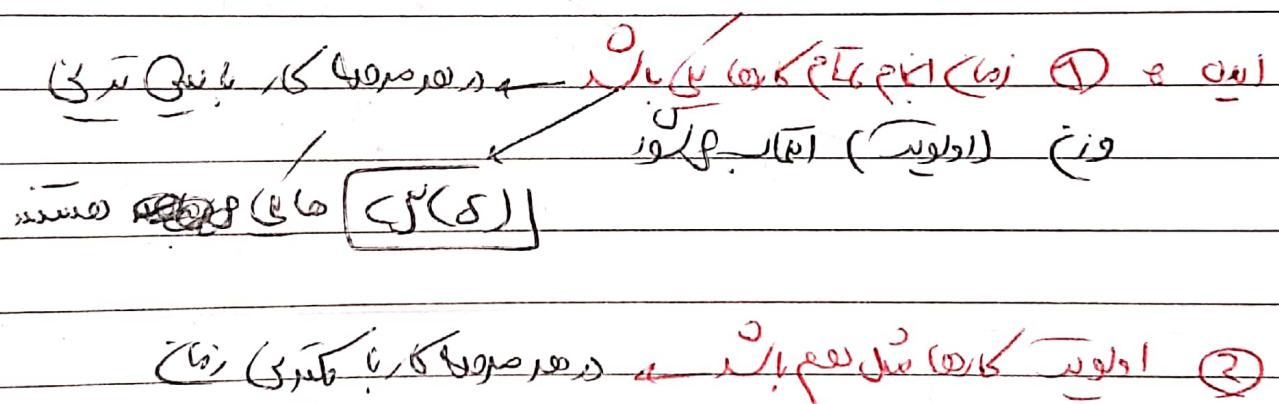
ویری (V) کا، تینی (T) کا، زمکی (Z) کا، باریل (B) کا،

$C_j(s)$



$$2 \times 0/8 + 4 \times 0/4 + 1 \times 0/1$$

(نفیری لور در مکرری زمان صحن)



صلی ب

	کار (ج)	کار (ج)
$C_5(5)$	$L_1 = 5$	$L_2 = 2$
ادلویی	$W_1 = 3$	$W_2 = 7$

$$5 \times 3 + (5 \times 2)^{x_1} = 22$$

نحوه کار نفیری

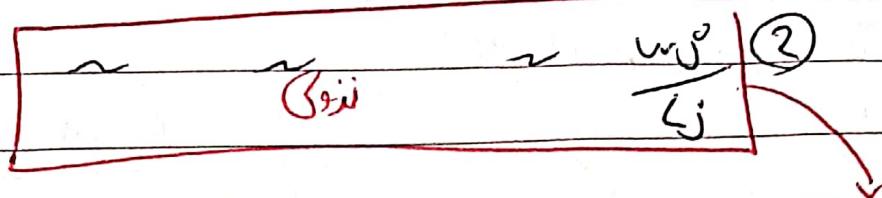
کار نفیری

$$2 \times 1 + (2 \times 5) \times 3 = 23$$

صلی ب

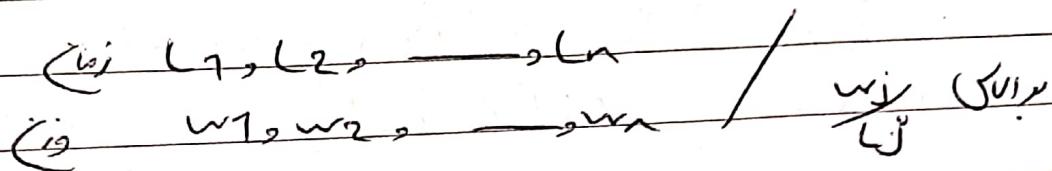
$$w_2 - l_2 = -1 \quad w_1 - l_1 = -2 \quad \xrightarrow{\quad} \quad X^{\circ}_{\rho_1 \rho_2}$$

~~لهم إني نذرت نفسي لكتابك سليم~~



exchange

و ن (ج) ب

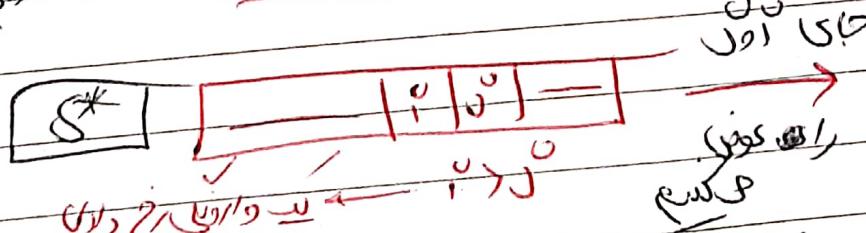
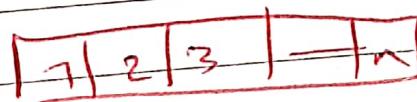


مذکور صدیقہ کنون

$$\frac{w_1}{L_1} > \frac{w_2}{L_2} > \dots > \frac{w_n}{L_n}$$

اول صفائی نہیں بھائی کہ لڑا و دریس کے لئے بلکہ اپنے

كـ مـ طـ بـ حـ حـ



دکان صورت (اریاضی کشم) با حلب جاہ اور گلہری میرے نامہ

$$S^* = S$$

هر طبق سطح ای خوب است

اگر ویرایش حرفی که مانع از این

$$\sum_{\delta} w_i c_i(\delta)$$



بودن آن ایستاد

$$S_1^* - S_2^*$$



$$= \left[ L_j x w_i - L_i x w_j \right]$$

که درست باشد ای این عبارت نیز

$$\frac{w_j}{L_j} > \frac{w_i}{L_i} \rightarrow L_i w_j - L_j w_i > 0$$

$$c_i(S_1^*) = c_i(S^*) + L_i \quad \text{اگر} = l$$



$$c_j(S_1^*) = c_j(S^*) - L_j$$

این کار نیز درست نیست زیرا  $S_1^*$  ایستاد

این کار نیز درست نیست زیرا  $S_1^*$  ایستاد

این کار نیز درست نیست زیرا  $(S^*)$  جایگزین  $(S)$  شد

این کار نیز درست نیست زیرا  $S^*$  ایستاد

و (align)  $\rightarrow$  (sort)  $\rightarrow$  ای اگر این کار