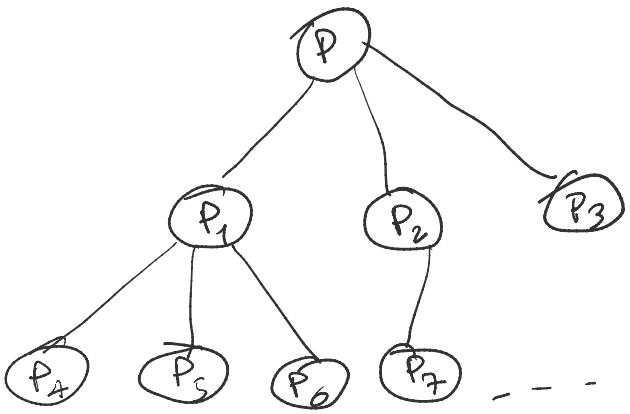
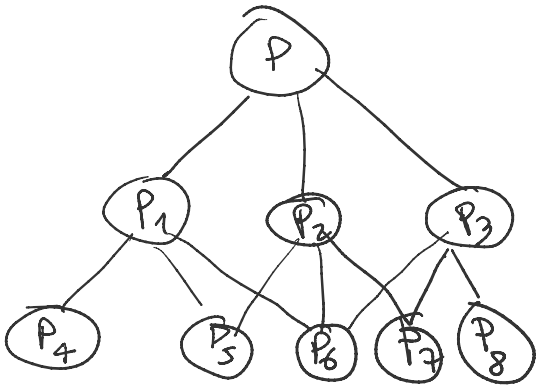
Lezione 28 aprile

Tuesday, April 28, 2020 4:15 PM

Programmazione Dinamica

Simile al D. et I.





Ottimizzazione

min o max

Alg. di Progr. Dinamica definito in 4 passi

1) Uguale caratterizzazione della struttura sia del problema generale che dei sottoproblemi
2) Definizione dei sottoproblemi elementari
3) Regola ricorsiva
4) Strategia memorizzazione delle soluzioni

4) Strategia memorizzazione delle soluzioni parziali

$$\begin{cases} F_0 = 0 \\ F_1 = 1 \\ F_i = F_{i-1} + F_{i-2} \end{cases}$$

$$F_n = \frac{1}{\sqrt{5}}\left(\left(\frac{1+\sqrt{5}}{2}\right)^n - \left(\frac{1-\sqrt{5}}{2}\right)^n\right)$$

ma ...

F_{18} presenta degli errori

```
Fib(n):
if (n ≤ 1) return n;
   else return Fib(n-1) + Fib(n-2)
```

È un buon programma ??

$$T(n) = T(n-1) + T(n-2) + \Theta(1) \geq 2T(n-2) + \Theta(1)$$

$$\vdots$$

$$T(n) \geq 2^{n/2}$$ esponenziale in n!

Progr. Din. Fib2(n); tabella F[0..n]

Progr. Din. Fib2 (n); tabella $F[0..n]$

```
F[0] = 0; F[1] = 1
for (i = 2; i ≤ n; i++)
    F[i] = F[i-1] + F[i-2];
return F[n];
```

$\Theta(n)$

$\Theta(n)$ spazio

```
Fib-migliore (n):
if (n <= 1) return n;
else {
    a = 0; b = 1;
    for (i = 2; i ≤ n; i++){
        c = a + b;
        a = b;
        b = c;
    }
return c;
```

$\Theta(n)$ tempo

spazio $\Theta(1)$

Longest common subsequence LCA

A: ANALISI

a: A D C A A B (m)

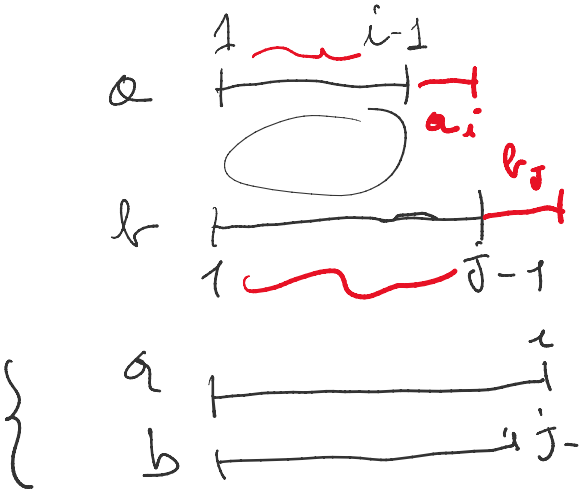
b: D A A B D C D A A C A C C B A (n)

Bioinformatica
DNA
Lunghezza della sequenza di lunghezza massima

1) $LCS(a[1..i], b[1..j])$ sotto pr.
$LCS(a[1..m], b[1..n])$ pr. gen.

2) $LCS(\emptyset, b[1..j]) = 0$
$LCS(a[1..i], \emptyset) = 0$

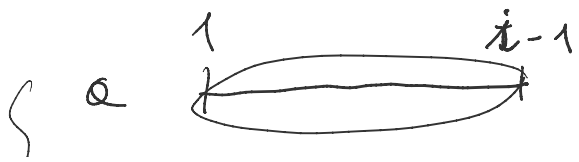
3)

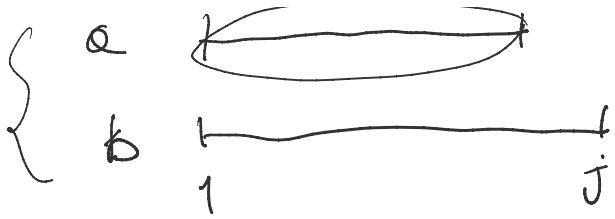


se $a_i = b_j$

$LCS(i,j) = LCS(i-1, j-1) + 1$

$LCS(i,j) = LCS(i, j-1) + 1$





$$LCS(i,j) \begin{cases} 0 & \text{se } i=0 \text{ o } j=0. \\ LCS(i-1,j-1)+1 & \text{se } a_i = b_j \\ \max(LCS(i,j-1), LCS(i-1,j)) & \text{else} \end{cases}$$

4) Usiamo

array $L[0..m, 0..n]$ per memorizzare le soluzioni parziali.

a = AMICA $\quad m = 5$

b = MATEMATICA $\quad n = 10$

L

	∅	M	A	T	E	M	A	T	I	C	A
∅	0	0	0	0	0	0	0	0	0	0	0
A	0	0	1	1	1	1	1	1	1	1	1
M	0	1	1	1	1	2	2	2	2	2	2
I	0	1	1	1	1	2	2	2	3	3	3
C	0	1	1	1	1	2	2	2	3	4	4
A	0	1	2	2	2	2	3	3	3	4	5

procedo per righe

L[m,n] = lunghezza della LCS

L[m,n] = lunghezza della LCS

tempo $\Theta(m \cdot n)$

spazio? $\Theta(m \cdot n)$

lunghezza ? spazio $\Theta(n)$ oppure $\Theta(n)$

sequenza $\Theta(m \cdot n)$

Ricostruzione sequenza LCS
dopo il calcolo di L

W = nuovo array di dim L[m,n]

```
i = m;
j = n;
k = L[m,n]
```


```
while (i>0) && (j>0) {
    if a[i] == b[j] {
        W[k] = a[i];
        k--;
        i--;
        j--;
    } else if (L[i,j] == L[i-1,j]) i--;
        else j--;
```

```
} print (w);
```