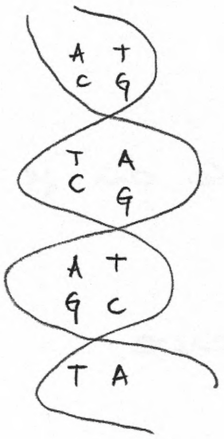
DINAMIČKO PROGRAMIRANJE

- PROBLEM: usporediti sličnost DNA dva organizma



- razmotati polinukleotidne lance građene od 4 vrste nukleotida

A - adenin
T - timin
C - citozin
G - guanin

odgovor: koliko je dugačak najdulji zajednički podniz?

A C T T C A G

6

G A C T C A G

A C T T C A G

3

G G C T T T A

Problem:

Za dana 2 niza

$$X = \langle x_1, \ldots, x_m \rangle \; ; \; Y = \langle y_1, \ldots, y_n \rangle$$

treba pronaći zajednički podniz od X i Y tako da je njegova duljina najveća.

skraćeno: LCS - engl. longest common subsequence

Primjer:

$X = \langle A, C, T, T, C, A, G \rangle$ $\quad lcs(X, Y) = 6$

$Y = \langle G, A, C, T, C, A, G, T \rangle$ $\quad lcs(X, U) = 3$

$U = \langle G, G, C, T, T, T \rangle$ $\quad LCS(X, Y) = \langle A, C, T, C, A, G \rangle$

Naivno rješenje:

- provjeriti svaki podniz od X je li podniz od Y
- uzeti najdulji

Algoritam:

Ulaz: X, Y

Izlaz: $Z = LCS(X, Y)$

```
c = 0
Za svaki podniz S od X
    if S podniz od Y onda
        then if |S| > c then
            Z ← S;  c ← |S|
        end if
    end if
kraj za svaki
Vrati Z
```

Složenost algoritma???

$m = |X|$, $n = |Y|$

Koliko ima podnizova od X?

2^m podnizova

Koliko "košta" provjera je li S podskup od Y

$O(n)$

Odgovor: $O(n 2^m)$

$|X|, |Y| \approx 12\,650\,000$ (kromosom 5)

Broj koraka: $12650000 \times 2^{12650000} \approx \underline{\underline{10^{10^6}}}$

Broj atoma u svemiru: $\underline{\underline{10^{80}}}$

<u>Ideja:</u> - koristiti tehniku dinamičkog programiranja
- rješenje potproblema koristiti za rješenje problema

<u>Oznake:</u>

$X_i = \langle x_1, \ldots, x_i \rangle$ prefiks od X

$Y_i = \langle y_1, \ldots, y_i \rangle$ -II- -II- Y

$F[i,j] := lcs(X_i, Y_j)$

$F[m,n] := lcs(X, Y)$

Primjer:

$\boxed{x_7 = y_6}$

A C T T C A $\boxed{G}$
G A C T C A $\boxed{G}$

$z_6 = x_7 = y_7$

A C T T C A
G A C T C A

$Z_5 = LCS(X_6, Y_6)$

$\boxed{x_7 \neq y_6}$

A $\boxed{C\ T\ T}$ C A $\cancel{G}$
G G $\boxed{C\ T}$ $\boxed{T}$ $\boxed{T}$

$z_3 \neq x_7$

$Z = LCS(X_6, Y)$

$z_3 \neq y_6$

A $\boxed{C\ T}$ T C A $\boxed{G}$
G G $\boxed{C\ T\ G}$ $\cancel{T}$

$Z = LCS(X, Y_5)$

aključak:

orem (o optimalnoj podstrukturi LCS problema)

eka su $X = \langle x_1, \dots, x_m \rangle$ i $Y = \langle y_1, \dots, y_n \rangle$ nizovi i $Z = \langle z_1, \dots, z_k \rangle$ lo koji LCS od X i Y. Tada vrijede sljedeće tvrdnje:

1. Ako je $x_m = x_n$ onda je $z_k = x_n = x_m$ i $Z_{k-1} = LCS(X_{m-1}, Y_{n-1})$
2. Neka je $x_m \neq y_n$. Ako je $z_k \neq x_m$ onda je $Z = LCS(X_{m-1}, Y)$
3. Neka je $x_m \neq y_n$. Ako je $z_k \neq y_n$ onda je $Z = LCS(X, Y_{n-1})$

oz. (Leiserson, Rivest, Stein 2nd edition, 351. str)

Interpretacija teorema:

- $x_m \neq y_n$
  - $z_k \neq x_m$: $Z = LCS(X_{m-1}, Y)$
  - $z_k \neq y_n$: $Z = LCS(X, Y_{n-1})$

  rješenje će biti dulji podniz
- $x_m = y_n$

  $\Downarrow$

  $Z_{k-1} = LCS(X_{m-1}, Y_{n-1})$

  x_m dodati na kraj niza Z_{k-1}

$$\boxed{c[i,j] = \begin{cases} 0 & \text{, ako je } i=0 \text{, ili } j=0 \\ c[i-1, j-1] + 1 & \text{, } i,j>0 \quad x_i = y_j \\ \max(c[i, j-1], c[i-1, j]) & \text{, } i,j>0 \quad x_i \neq y_j \end{cases}}$$

- najprije ćemo izračunati duljinu najdužeg zajedničkog podniza

```
LCS-LENGTH (X, Y)
  m ← X.length
  n ← Y.length
  for i ← 1 to m
     do c[i, 0] ← 0
  end for
  for j ← 1 to n
     do c[0, j] ← 0
  end for
  for i ← 1 to m
     do for j ← 1 to n
        do if x_i = y_j
           then c[i, j] ← c[i-1, j-1] + 1
                b[i, j] ← "↖"
           else if c[i-1, j] ≥ c[i, j-1]
                then c[i, j] ← c[i-1, j]
```

| $b[i,j] \leftarrow$ "↑"

else $f[i,j] \leftarrow f[i,j-1]$

| $b[i,j] \leftarrow$ "←"

end if

return f and b

X\Y		G	A	C	T	C	A	G	T
	0	0	0	0	0	0	0	0	0
A	0	0	1	1	1	1	1	1	1
C	0	0	1	2	2	2	2	2	2
T	0	0	1	2	3	3	3	3	3
T	0	0	1	2	3	3	3	3	4
C	0	0	0	2	3	4	4	4	4
A	0	0	1	2	3	4	5	5	5
G	0	1	1	2	3	4	5	6	6

$lcs(X,Y) = f[7,8] = 6$

rekonstrukcija najduljeg zajedničkog podniza

```
PRINT LCS(b, X, i, j)
   if i=0 or j=0
       then return
   if b[i,j] = "↖"
       then PRINT-LCS (b, X, i-1, j-1)
            print: x_i
   else if b[i,j] = "↑"
       then PRINT-LCS(b, X, i-1, j)
   else PRINT-LCS(b, X, i, j-1)
```

Rezultat na prethodnom primjeru: ACTCAG

Vrijeme izvršavanja: LCS-LENGTH – $\Theta(mn)$

PRINT-LCS – $O(m+n)$

- LCS - primjer optimizacijskog problema
- optimizacijski problem - svako rješenje ima pridruženu vrijednost
  cilj je pronaći rješenje s najvećom/najmanj vrijednošću
- dinamičko programiranje koristimo u rješavanju uglavnom optimizacijskih problema
- ključni koraci:

1. Okarakterizirati strukturu optimalnog rješenja
2. Rekurzivno definirati vrijednost optimalnog rješenja
3. Izračunati vrijednost optimalnog rješenja (koristeći bottom-up pristup)
4. Rekonstruirati optimalno rješenje na osnovu izračunatih informacija

Primjer (LCS)

1. Teorem o optimalnoj podstrukturi LCS problema
2. Rekurzivna relacija za računanje optimalne vrijednosti
3. LCS-LENGTH algoritam i tablica
4. PRINT-LCS algoritam (i tablica)

Ulančano množenje matrica

Problem:

ulaz: niz matrica $A_1, \dots, A_n$, $n \in \mathbb{N}$ za koje je definiran produkt

$$A_1 A_2 \cdots A_n$$

izlaz: izračunati produkt $A_1 A_2 \cdots A_n$ koristeći najmanji mogući broj operacija množenja skalara

- asocijativnost množenja matrica:

$$(A_1(A_2(A_3 \cdot A_4)))$$
$$(A_1((A_2 A_3) A_4))$$
$$((A_1 A_2)(A_3 A_4))$$
$$((A_1(A_2 A_3)) A_4)$$
$$(((A_1 A_2) A_3) A_4)$$

- uočimo da raspored zagrada ima utjecaj na broj operacija

Primjer:

$$\underset{10 \times 100}{A_1} \times \underset{100 \times 5}{A_2} \times \underset{5 \times 50}{A_3}$$

dva načina: $((A_1 A_2) A_3)$

$$(A_1 \times A_2)$$
$$10 \cdot 100 \cdot 5 = \underline{5000}$$
$$\underset{10 \times 5}{(A_1 \times A_2)} \times \underset{5 \times 50}{A_3}$$
$$10 \cdot 5 \cdot 50 = \underline{2500}$$

Ukupno: 7500 op.

$A_1 (A_2 A_3)$

$(A_2 \times A_3)$

$100 \times 5 \cdot 50 = \underline{25000}$

$A_1 \times (A_2 \times A_3)$

$10 \times 100 \qquad 100 \times 50$

$10 \cdot 100 \cdot 50 = \underline{50000}$

Ukupno: 75000 op.

Zaključak: prvi pristup je 10 puta brži.

Iznake:

$A_1 \quad A_2 \quad \ldots \quad A_n$

$p_0 \times p_1 \quad p_1 \times p_2 \quad \ldots \quad p_{n-1} \times p_n$

matrica A_i ima dimenziju

roj različitih načina postavljanja zagrada

$(\underbrace{A_1 \ldots A_k}_{\text{broj načina ovdje}}) \otimes (\underbrace{A_{k+1} \ldots A_n}_{\text{broj načina ovdje}}) \qquad k = 1, \ldots, n-1$

$$P(n) = \begin{cases} 1 & , n = 1 \qquad (A_1) \\ \sum_{k=1}^{n-1} P(k)P(n-k) & , n \geq 2 \end{cases}$$

Može se pokazati da je $P(n) \in \Omega(2^n)$ (D.Z).

Brute-force metoda ne prolazi!

(Struktura optimalnog postavljanja zagrada

$$\underbrace{\underbrace{\boxed{A_i \cdots A_{k-1}}}_{\text{optimalno}} \; \underbrace{\boxed{A_k \cdots A_j}}_{\text{optimalno}}}_{\text{optimalno}}$$

Rekurzija:

$m[i,j] \leftarrow$ minimalan broj skal. množenja za računanje produkta $A_i \cdots A_j$ $\quad 1 \le i \le j \le n$

rješenje (ono što tražimo) je $m[1,n]$

$i = j \Rightarrow A_i \cdots A_i \Rightarrow m[i,i] = 0 \quad 1 \le i \le n$

$i < j \Rightarrow A_i \cdots A_j \Rightarrow m[i,j] = m[i,k] + m[k+1,j] + p_{i-1}\, p_k\, p_j$

$$\underbrace{A_i \cdots A_k}_{m[i,k]} \quad \underbrace{A_{k+1} \cdots A_j}_{m[k+1,j]} \qquad i \le k < j$$

$$\underbrace{\underbrace{A_i \cdots A_k}_{p_{i-1} \times p_k} \quad \underbrace{A_{k+1} \cdots A_j}_{p_k \times p_j}}_{p_{i-1}\, p_k\, p_j \text{ op.}}$$

Rekurzivna relacija:

$$m[i,j] = \begin{cases} 0 & , \; i = j \\ \min\limits_{i \le k < j} \{ m[i,k] + m[k+1,j] + p_{i-1}\, p_k\, p_j \} & , \; i < j \end{cases}$$

Oznaka: neka je $s[i,j] = \bar{k}$ takav da je

$$m[i,j] = m[i,\bar{k}] + m[\bar{k}+1,j] + p_{i-1}\, p_{\bar{k}}\, p_j$$

pristup računanja $m[i,j]$ je <u>tabularizacija</u> (bottom-up)
pristup kao kod LSC-a.

- broj množenja ovisi samo o dimenzijama

$$p = \langle p_0, p_1, \ldots, p_n \rangle \qquad p.\text{length} = n+1$$

MATRIX-CHAIN-ORDER (p)

```
n ← p.length - 1
for i ← 1 to n
    m[i,i] ← 0
for l ← 2 to n
    for i ← 1 to n - l + 1
        j ← i + l - 1
        m[i,j] ← ∞
        for k ← i to j-1
            q ← m[i,k] + m[k+1,j] + p_{i-1} p_k p_j
            if q < m[i,j]
                m[i,j] ← q
                s[i,j] ← k
            end if
        end for
    end for
end for
return m, s
```

$A_i \ldots A_j$

$p_{i-1} \ldots p_j$

dulјine

ima ih $j-(i-1) = \underline{j-i+1}$

$A_i \ldots A_{\underbrace{i+l-1}_{j}}$

<u>Primjer:</u>

matrica	A_1	A_2	A_3	A_4	A_5	A_6
dim	30×35	35×15	15×5	5×10	10×20	20×25

i\j	1	2	3	4	5	6
1	0	15750	7875	9375	11875	15125
2		0	2625	4375	7125	10500
3			0	750	2500	5375
4				0	1000	3500
5					0	5000
6						0

i\j	1	2	3	4	5	6
1	x	1	1	3	3	3
2		x	2	3	3	3
3			x	3	3	3
4				x	4	5
5					x	5
6						x

$m[1,2] = \overset{=0}{m[1,1]} + \overset{=0}{m[2,2]} + p_0 p_1 p_2 = 30 \cdot 35 \cdot 15 = 15750$

$m[2,3] = \overset{=0}{m[2,2]} + \overset{=0}{m[3,3]} + p_1 p_2 p_3 = 35 \cdot 15 \cdot 5 = 2625$

$m[3,4] = \quad \vdots \qquad \vdots \qquad + p \vdots$

$m[4,5] = \quad \vdots \qquad \vdots \qquad \vdots$

$m[5,6] = \overset{=0}{m[5,5]} + \overset{=0}{m[6,6]} + p_4 p_5 p_6 = 10 \cdot 20 \cdot 25 = 5000$

$m[1,3] = m[1,$

$$m[1,3] = \min \begin{cases} m[1,1] + m[2,3] + p_0 p_1 p_3 = 0 + 2625 + 5250 = 7875 \\ m[1,2] + m[3,3] + p_0 p_2 p_3 = 15750 + 0 + 2250 = 18000 \end{cases}$$

$s[1,3] = 1$

Konstrukcija optimalnog rješenja:

```
PRINT-OPTIMAL-PARENS (s, i, j)
  if i == j
      print A_i
  else print "("
      PRINT-OPTIMAL-PARENS (s, i, s[i,j])
      PRINT-OPTIMAL-PARENS (s, s[i,j]+1, j)
      print ")"
  end if
```

Poziv:

PRINT-OPTIMAL-PARENS (s, 1, 6)

$$((1)(2\ 3))((4\ 5)(6))$$

$$(A_1\ (A_2 A_3))\ ((A_4 A_5)\ A_6)$$

Vrijeme izvršavanja:

MATRIX-CHAIN-ORDER:

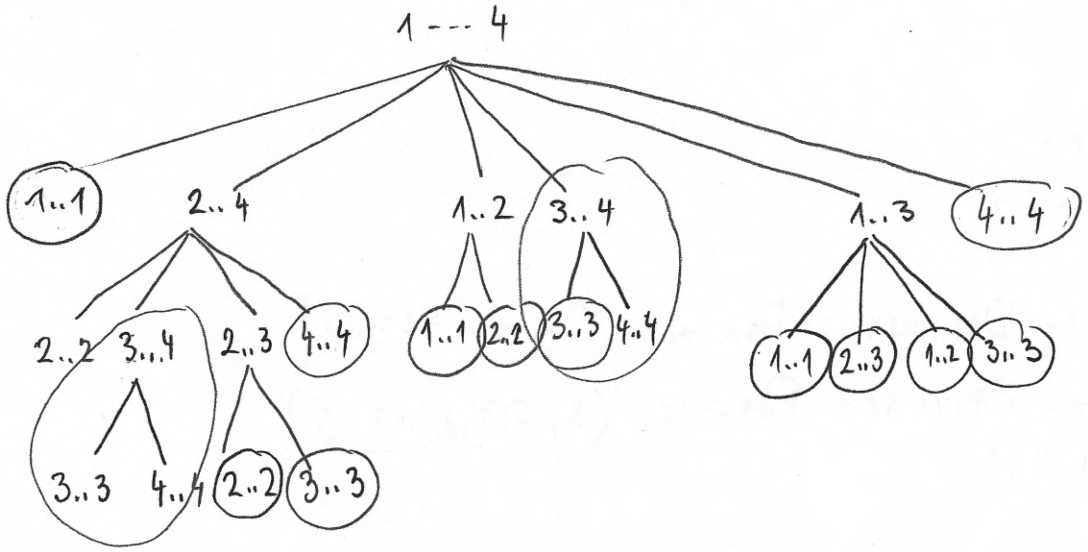
3 for petlje

$$T(n) = \sum_{l=2}^{n} \sum_{i=1}^{n-l+1} \sum_{k=i}^{i+l-2} \Theta(1) = O(n^3)$$

D.Z. $\Theta(n^3)$ (odnosno $T(n) \in \Omega(n^3)$)

prostorna složenost: $\Theta(n^2)$

Razmotrimo rekurzivan pristup (nije tabularizacija)



```
RECURSIVE-MATRIX-CHAIN (p, i, j)
  if i == j
     return 0
  m[i, j] ← ∞
  for k ← i to j-1
        q = RECURSIVE-MATRIX-CHAIN (p, i, k)
            + RECURSIVE-MATRIX-CHAIN (p, k+1, j)
            + p_{i-1} p_k p_j
        if q < m[i, j]
           m[i, j] ← q
        end if
  end for
return m[i, j]
```

RECURSIVE - MATRIX-CHAIN (p, i, j)

Vrijeme izvršavanja: (donja međa)

$T(1) \geq 1$

$$T(n) \geq 1 + \sum_{k=1}^{n-1} \left(T(k) + T(n-k) + 1 \right)$$

ekvivalentno

$$T(n) \geq 2 \sum_{i=1}^{n-1} T(i) + n$$ (stablo rekurzije + metoda supstitucije)

Pretp: $T(n) = \Omega(2^n)$ $\quad T(n) \geq 2^{n-1}, n \in \mathbb{N}$

Baza: $T(1) \geq 2^{1-1} = 2^0 = 1$

Pretpostavka: $T(k) \geq 2^{k-1}, \ k \leq n$

Korak: $$T(n+1) \geq 2 \sum_{i=1}^{(n+1)-1} T(i) + n+1$$

$$\geq 2\sum_{i=1}^{n-1} 2^{i-1} + n+1$$

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

$$= 2(2^{n}-1) + (n+1)$$

$$= 2^{n+1} - 2 + n+1 = 2^{n+1} + (n-1)$$

$$\geq \underline{\underline{2^{n}}} \quad [= 2^{(n+1)-1} \quad \Rightarrow \quad \underline{\underline{T(n) = \Omega(2^{n})}}$$

-ukoliko „pamtimo" rješenje potproblema koji se preklapaju dolazimo do pristupa koji se zove memoizacija

```
MEMOIZED-MATRIX-CHAIN (p)
  n ← p.length - 1
  for i ← 1 to n
      for j ← 1 to n
          m[i,j] ← ∞
      end for
  end for
  return LOOKUP-CHAIN (m, p, 1, n)

LOOKUP-CHAIN (m, p, i, j)
  if m[i,j] < ∞
        return m[i,j]
  end if
  if i == j
        m[i,j] ← 0
  else  for k ← i to j-1
```

```
q ← LOOKUP-CHAIN (m, p, i, k)
    + LOOKUP-CHAIN (m, p, k+1, j)
    + p_{i-1} p_k p_j
if q < m[i,j]
|    m[i,j] ← q
end if
return m[i,j]
```

Vrijeme izvršavanja: $O(n^3)$??? (analiza D.Z)