Dynamisk programmering

Flere eksempler

Eksempel 1: Længste fælles delsekvens

Alfabet = mængde af tegn:

$\{$a,b,c,...,z$\}$, $\{$A,C,G,T$\}$, $\{$0,1$\}$

Eksempel 1: Længste fælles delsekvens

Alfabet = mængde af tegn:

$\{a,b,c,\ldots,z\}$, $\{A,C,G,T\}$, $\{0,1\}$

Streng = sekvens $x_1 x_2 x_3 \ldots x_n$ af tegn fra et alfabet:

helloworld

GATAAATCTGGTCTTATTTCC

00101100101010001111

Eksempel 1: Længste fælles delsekvens

Alfabet = mængde af tegn:

$\{a,b,c,\ldots,z\}, \quad \{A,C,G,T\}, \quad \{0,1\}$

Streng = sekvens $x_1 x_2 x_3 \ldots x_n$ af tegn fra et alfabet:

helloworld

GATAAATCTGGTCTTATTTCC

00101100101010001111

Delsekvens = delmængde af tegnene i streng, i uændret rækkefølge:

G A T A A A T C T C

G A T C

Længste fælles delsekvens

Fælles delsekvens for to strenge:

X G A T A A A T C T C

G A T C

Y A G T T A T T C

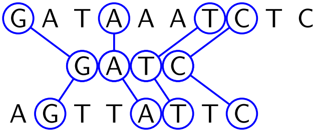
Eller blot:

X G A T A A A T C T C

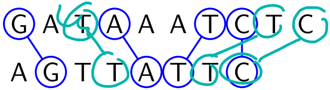
Y A G T T A T T C

Længste fælles delsekvens

Fælles delsekvens for to strenge:



Eller blot:



Længste fælles delsekvens (Longest Common Subsequence, LCS):

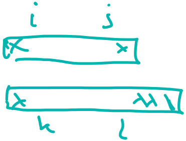
Givet to strenge

$$X = x_1 x_2 x_3 \dots x_m$$
$$Y = y_1 y_2 y_3 \dots y_n$$

af længde m og n, find en længste fælles delsekvens for dem.

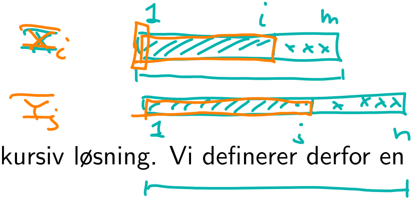
Længden af denne kan ses som et mål for similaritet mellem strenge (f.eks. dna-strenge).

Rekursiv løsning?



Vi vil arbejde på at lave en rekursiv løsning. Vi definerer derfor en størrelse for del-problemer:

Rekursiv løsning?



$X = X_m$

$Y = Y_n$

Vi vil arbejde på at lave en rekursiv løsning. Vi definerer derfor en størrelse for del-problemer:

- $X_i = x_1 x_2 x_3 \ldots x_i$ for $1 \leq i \leq m$.
- $Y_j = y_1 y_2 y_3 \ldots y_j$ for $1 \leq j \leq n$.
- X_0 og Y_0 er den tomme streng.
- lcs(i, j) er længden af længste fælles delsekvens af X_i og X_j.

Rekursiv løsning?

Vi vil arbejde på at lave en rekursiv løsning. Vi definerer derfor en størrelse for del-problemer:

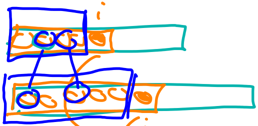
- $X_i = x_1 x_2 x_3 \ldots x_i$ for $1 \leq i \leq m$.
- $Y_j = y_1 y_2 y_3 \ldots y_j$ for $1 \leq j \leq n$.
- X_0 og Y_0 er den tomme streng.
- $\mathrm{lcs(i, j)}$ er længden af længste fælles delsekvens af X_i og X_j.

Vi vil gerne finde $\mathrm{lcs(m, n)}$.

Mere generelt: Vi søger en rekursiv formel for $\mathrm{lcs(i, j)}$.

Basistilfælde: Det er klart at $\mathrm{lcs(0, j)} = \mathrm{lcs(i, 0)} = 0$.

Optimale delproblemer I



Formel for lcs(i, j):

$\boxed{\text{Case I: } x_i = y_j}$

k

Observation: en fælles delsekvens Z for X_i og Y_j af længde k består af

f.ex

- Et sidste tegn z_k.
- En streng $\boxed{Z'} = z_1 z_2 z_3 \ldots z_{k-1}$ af længde $k-1$, som må være en fælles delsekvens af X_{i-1} og Y_{j-1} (tegnene i Z skal komme i samme rækkefølge som i X og Y, så kun sidste tegn i Z har mulighed for at være x_i og y_j).

k − 1

Optimale delproblemer I

Formel for lcs(i, j): Case I: $x_i = y_j$

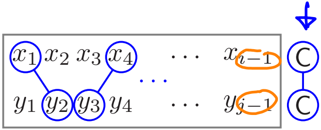
Observation: en fælles delsekvens Z for $\underbrace{X_i \text{ og } Y_j}$ af længde k består af

- Et sidste tegn z_k.
- En streng $\boxed{Z'} = z_1 z_2 z_3 \ldots z_{k-1}$ af længde $k-1$, som må være en fælles delsekvens af X_{i-1} og Y_{j-1} (tegnene i Z skal komme i samme rækkefølge som i X og Y, så kun sidste tegn i Z har mulighed for at være x_i og y_j).

$\underbrace{(k-1)} + 1$

Den essentielle egenskab (optimale delproblemer) for Case I:

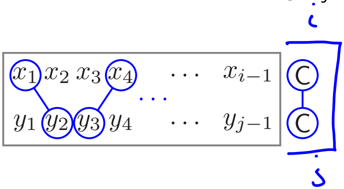
Hvis Z er en længste fælles delsekvens for for X_i og Y_j, må Z' være en længste fælles delsekvens af X_{i-1} og Y_{j-1}. For hvis der fandtes en længere fælles delsekvens for X_{i-1} og Y_{j-1}, kunne den tilføjes tegnet x_i $(= y_j)$ og blive en længere fælles delsekvens for X_i og Y_j.



Optimale delproblemer I

Af den essentielle egenskab haves i Case I ($x_i = y_j$):

- $\mathrm{lcs}(i, j) = \mathrm{lcs}(i-1, j-1) + 1$
- En længste fælles delsekvens for X_{i-1} og Y_{j-1} tilføjet tegnet x_i ($= y_j$) er en længste fælles delsekvens for X_i og Y_j.



Optimale delproblemer II

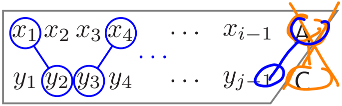
Formel for $\mathrm{lcs(i,j)}$: $\boxed{\text{Case II: } x_i \neq y_j}$

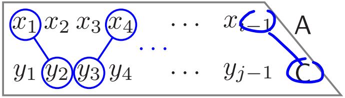
Observation: en fælles delsekvens $Z = z_1 z_2 z_3 \ldots z_k$ for X_i og Y_j kan ikke have z_k værende en parring af x_i og y_j (da disse jo er forskellige).

Så Z må være en fælles delsekvens for *enten* X_{i-1} og Y_j *eller* for X_i og Y_{j-1} (eller evt. begge).

Den essentielle egenskab (optimale delproblemer) for Case II:

> Hvis Z er en længste fælles delsekvens for X_i og Y_j, må den være en længste fælles delsekvens for enten X_{i-1} og Y_j eller for X_i og Y_{j-1} (eller evt. begge). For hvis der fandtes en længere fælles delsekvens for enten X_{i-1} og Y_j eller for X_i og Y_{j-1}, ville denne også være en længere fælles delsekvens for X_i og Y_j.





Optimale delproblemer II

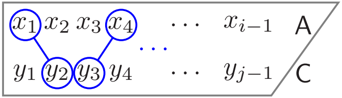
Lad T_1 være en længste fælles delsekvens for X_{i-1} og Y_j, og lad T_2 være en længste fælles delsekvens for X_i og Y_{j-1}.

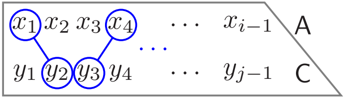
Af den essentielle egenskab i Case II ($x_i \neq y_j$) haves at blandt T_1 og T_2 er der (mindst) en som er en længste fælles delsekvens for X_i og Y_j.

Ingen af T_1 og T_2 kan være længere end den længste fælles delsekvens for X_i og Y_j (da de begge er delsekvense af X_i og Y_j).

Så af den essentielle egenskab haves i Case II ($x_i \neq y_j$):

- $\mathrm{lcs}(\mathrm{i},\mathrm{j}) = \max(\mathrm{lcs}(\mathrm{i}-1,\mathrm{j}), \mathrm{lcs}(\mathrm{i},\mathrm{j}-1))$
- Hvis $\mathrm{lcs}(\mathrm{i}-1,\mathrm{j}) \geq \mathrm{lcs}(\mathrm{i},\mathrm{j}-1)$, er en længste fælles delsekvens for X_{i-1} og Y_j også en længste fælles delsekvens for X_i og Y_j. Et symmetrisk udsagn gælder for "$\leq$" og X_i og Y_{j-1}.





Rekursiv formel for $\mathsf{lcs}(i, j)$

Alt i alt har vi fundet flg. rekursive formel for $\mathsf{lcs}(i, j)$:

$$\mathsf{lcs}(i,j) = \begin{cases} 0 & \text{if } i = 0 \text{ or } j = 0 \\ \mathsf{lcs}(i-1, j-1) + 1 & \text{if } i, j > 0 \text{ and } x_i = y_j \\ \max(\mathsf{lcs}(i-1, j), \mathsf{lcs}(i, j-1)) & \text{if } i, j > 0 \text{ and } x_i \neq y_j \end{cases}$$

Rekursiv formel for $\mathsf{lcs}(i,j)$

Alt i alt har vi fundet flg. rekursive formel for $\mathsf{lcs}(i,j)$:

$$\mathsf{lcs}(i,j) = \begin{cases} 0 & \text{if } i = 0 \text{ or } j = 0 \\ \mathsf{lcs}(i-1,j-1)+1 & \text{if } i,j > 0 \text{ and } x_i = y_j \\ \max(\mathsf{lcs}(i-1,j),\mathsf{lcs}(i,j-1)) & \text{if } i,j > 0 \text{ and } x_i \neq y_j \end{cases}$$

Den giver anledning til en naturlig, simpel rekursiv algoritme.

MEN: det er nemt at se at der er gentagelser blandt delproblemers delproblemer.

Så samme delproblemer bliver gentagne gange beregnet forskellige steder i rekursionstræet, og køretiden bliver meget dårlig.

Kan evt. løses med memoization: hav en tabel med plads til svaret på alle de mulige delproblemer $\mathsf{lcs}(i,j)$, og gem svaret når det er beregnet første gang. Siden, slå det bare op.

Dynamisk programmering: udfyld i stedet direkte denne tabel bottom-up på struktureret måde.

Dynamisk programmering

Dynamisk programmering: udfyld tabel over $\mathsf{lcs(i,j)}$ bottom-up på struktureret måde.

$i \backslash j$	0	1	2	$\cdot$	$\cdot$	$\cdot$	n
0							
1							
2							
$\cdot$							
$\cdot$							
m							

$$\mathsf{lcs(i,j)} = \begin{cases} 0 & \text{if } i = 0 \text{ or } j = 0 \\ \mathsf{lcs(i-1,j-1)+1} & \text{if } i,j > 0 \text{ and } x_i = y_j \\ \mathsf{max(lcs(i-1,j),lcs(i,j-1))} & \text{if } i,j > 0 \text{ and } x_i \neq y_j \end{cases}$$

Dynamisk programmering

Dynamisk programmering: udfyld tabel over lcs(i, j) bottom-up på struktureret måde.

$i \backslash j$	0	1	2	·	·	·	n
0	0	0	0	0	0	0	0
1	0						
2	0						
·	0						
·	0						
m	0						

$$\mathrm{lcs}(i,j) = \begin{cases} 0 & \text{if } i = 0 \text{ or } j = 0 \\ \mathrm{lcs}(i-1, j-1) + 1 & \text{if } i, j > 0 \text{ and } x_i = y_j \\ \max(\mathrm{lcs}(i-1, j), \mathrm{lcs}(i, j-1)) & \text{if } i, j > 0 \text{ and } x_i \neq y_j \end{cases}$$

Dynamisk programmering

Dynamisk programmering: udfyld tabel over lcs(i, j) bottom-up på struktureret måde.

j $\qquad$ y_j $\quad Y$

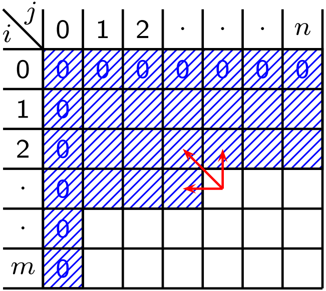
X $\quad x_i$ $\qquad i$

$i \backslash j$	0	1	2	·	·	·	n
0	0	0	0	0	0	0	0
1	0						
2	0						
·	0						
·	0						
m	0						

$$\mathrm{lcs(i,j)} = \begin{cases} 0 & \text{if } i = 0 \text{ or } j = 0 \\ \mathrm{lcs(i-1, j-1)} + 1 & \text{if } i, j > 0 \text{ and } x_i = y_j \\ \max(\mathrm{lcs(i-1, j)}, \mathrm{lcs(i, j-1)}) & \text{if } i, j > 0 \text{ and } x_i \neq y_j \end{cases}$$

Dynamisk programmering

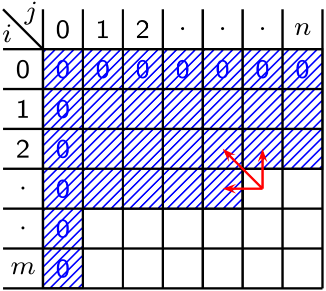
Dynamisk programmering: udfyld tabel over $\mathsf{lcs(i,j)}$ bottom-up på struktureret måde.



$$\mathsf{lcs(i,j)} = \begin{cases} 0 & \text{if } i = 0 \text{ or } j = 0 \\ \mathsf{lcs(i-1,j-1)} + 1 & \text{if } i,j > 0 \text{ and } x_i = y_j \\ \max(\mathsf{lcs(i-1,j)}, \mathsf{lcs(i,j-1)}) & \text{if } i,j > 0 \text{ and } x_i \neq y_j \end{cases}$$

Dynamisk programmering

Dynamisk programmering: udfyld tabel over lcs(i, j) bottom-up på struktureret måde.



$$\mathrm{lcs(i,j)} = \begin{cases} 0 & \text{if } i = 0 \text{ or } j = 0 \\ \mathrm{lcs(i-1,j-1)} + 1 & \text{if } i, j > 0 \text{ and } x_i = y_j \\ \max(\mathrm{lcs(i-1,j)}, \mathrm{lcs(i,j-1)}) & \text{if } i, j > 0 \text{ and } x_i \neq y_j \end{cases}$$

Dynamisk programmering

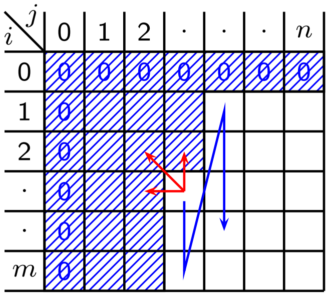
Dynamisk programmering: udfyld tabel over lcs(i, j) bottom-up på struktureret måde.

$i \backslash j$	0	1	2	·	·	·	n
0	0	0	0	0	0	0	0
1	0						
2	0						
·	0						
·	0						
m	0						

$$\mathrm{lcs}(i,j) = \begin{cases} 0 & \text{if } i = 0 \text{ or } j = 0 \\ \mathrm{lcs}(i-1, j-1) + 1 & \text{if } i,j > 0 \text{ and } x_i = y_j \\ \max(\mathrm{lcs}(i-1,j), \mathrm{lcs}(i,j-1)) & \text{if } i,j > 0 \text{ and } x_i \neq y_j \end{cases}$$

Dynamisk programmering

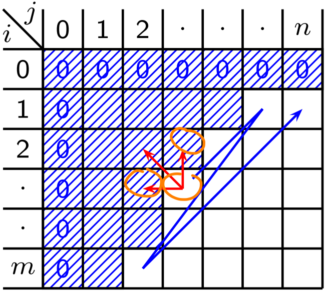
Dynamisk programmering: udfyld tabel over lcs(i, j) bottom-up på struktureret måde.



$$\mathrm{lcs}(i,j) = \begin{cases} 0 & \text{if } i = 0 \text{ or } j = 0 \\ \mathrm{lcs}(i-1, j-1) + 1 & \text{if } i,j > 0 \text{ and } x_i = y_j \\ \max(\mathrm{lcs}(i-1,j), \mathrm{lcs}(i,j-1)) & \text{if } i,j > 0 \text{ and } x_i \neq y_j \end{cases}$$

Dynamisk programmering

Dynamisk programmering: udfyld tabel over lcs(i, j) bottom-up på struktureret måde.



$$\mathrm{lcs}(\mathrm{i},\mathrm{j}) = \begin{cases} 0 & \text{if } i = 0 \text{ or } j = 0 \\ \mathrm{lcs}(\mathrm{i}-1,\mathrm{j}-1)+1 & \text{if } i,j > 0 \text{ and } x_i = y_j \\ \max(\mathrm{lcs}(\mathrm{i}-1,\mathrm{j}),\mathrm{lcs}(\mathrm{i},\mathrm{j}-1)) & \text{if } i,j > 0 \text{ and } x_i \neq y_j \end{cases}$$

Køretid

Dynamisk programmering: udfyld tabel over lcs(i, j) bottom-up på struktureret måde.

$i \backslash j$	0	1	2	$\cdot$	$\cdot$	$\cdot$	n
0	0	0	0	0	0	0	0
1	0						
2	0						
$\cdot$	0						
$\cdot$	0						
m	0						

Tabelstørrelse: mn

Udfyld tabelindgang: O(max størrelse af røde graf) $=$ $O(1)$.

Tid i alt: Θ(produktet af de to) $= O(mn)$.

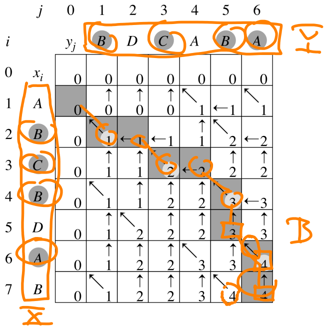
Find en konkret løsning

lcs(m, n) er længden af en længste fælles delsekvens for $X = X_m$ og $Y = Y_n$.

Hvis vi gerne vil finde en konkret fælles delsekvens af denne længde: Gem for hvert felt i tabellen hvilken af de tre røde pile som gav lcs(i, j)-værdien i dette felt.

$O(m+n)$

i j



+1

B[C]B[A]

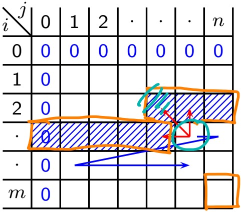
BCBA

Følg gemte pile baglæns fra lcs(m, n). Når en skrå pil følges er det en Case I, og x_i $(=y_j)$ udskrives. Ellers er den en Case II, og intet udskrives.

I alt udskrives en længste fælles delsekvens for X og Y i baglæns orden i tid $O(m+n)$.

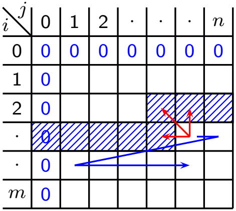
Pladsforbrug for LCS

Hvis vi kun skal bruge længden af længste fælles delsekvens, kan vi nøjes med $\min\{m, n\}$ plads:

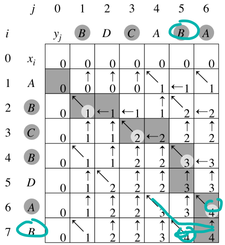


Pladsforbrug for LCS

Hvis vi kun skal bruge længden af længste fælles delsekvens, kan vi nøjes med $\min\{m, n\}$ plads:



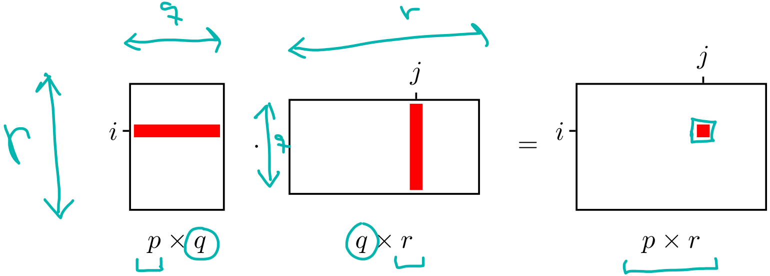
Hvis vi skal bruge en længste fælles delsekvens, må vi gemme hele tabellen, dvs. bruge $\Theta(mn)$ plads (da vi ikke kender stien tilbage, må vi gemme hele tabellen):



[Hirschberg gav i 1975 en metode til også at opnå dette med $\min\{m, n\}$ plads, men det er ikke pensum i DM507.]

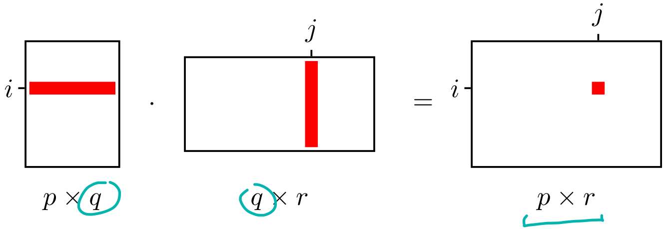
Eksempel 2: Multi-Matrix-multiplikation

En $p \times q$ matrix A_1 og en $q \times r$ matrix A_2 kan multipliceres i tid $O(pqr)$. Resultatet er en $p \times r$ matrix.



Eksempel 2: Multi-Matrix-multiplikation

En $p \times q$ matrix A_1 og en $q \times r$ matrix A_2 kan multipliceres i tid $O(pqr)$. Resultatet er en $p \times r$ matrix.



Matrix-multiplikation er associativ:

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

Multi-Matrix-multiplikation

Matrix-multiplikation er associativ:

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

Men køretiden er IKKE ens.

Multi-Matrix-multiplikation

Matrix-multiplikation er associativ:

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

Men køretiden er IKKE ens. Eksempel:

A_1	A_2	A_3
10×100	100×5	5×50

100×50

10×5

Multi-Matrix-multiplikation

Matrix-multiplikation er associativ:

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

Men køretiden er IKKE ens. Eksempel:

$$\begin{array}{ccc} A_1 & A_2 & A_3 \\ 10 \times 100 & 100 \times 5 & 5 \times 50 \end{array}$$

$$100 \times 50$$

$$10 \times 5$$

Tid for $A_1 \cdot (A_2 \cdot A_3)$: er $10 \cdot 100 \cdot 50 + 100 \cdot 5 \cdot 50 = 75.000$

Multi-Matrix-multiplikation

Matrix-multiplikation er associativ:

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

Men køretiden er IKKE ens. Eksempel:

$$\begin{array}{ccc} A_1 & A_2 & A_3 \\ 10 \times 100 & 100 \times 5 & \underline{5 \times 50} \end{array}$$

$$100 \times 50$$

$$\underline{10 \times 5}$$

Tid for $A_1 \cdot (A_2 \cdot A_3)$: er $10 \cdot 100 \cdot 50 + 100 \cdot 5 \cdot 50 = 75.000$

Tid for $(A_1 \cdot A_2) \cdot A_3$: er $\underline{10 \cdot 100 \cdot 5} + \underline{10 \cdot 5 \cdot 50} = 7.500$

Multi-Matrix-multiplikation

Catalan

Spørgsmålet:

$\sim 4^n$

Givet et produkt af n matricer

$$A_1, \quad A_2, \quad A_3, \quad \ldots, \; A_n$$

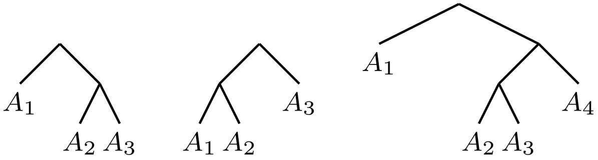
med kombatible dimensioner

$$p_0 \times p_1, \quad p_1 \times p_2, \quad p_2 \times p_3, \quad \ldots, \quad p_{n-1} \times p_n$$

hvad er den billigste rækkefølge at gange dem sammen i?

Beregningstræer

Rækkefølge = parentes-sætning = binært beregningstræ:



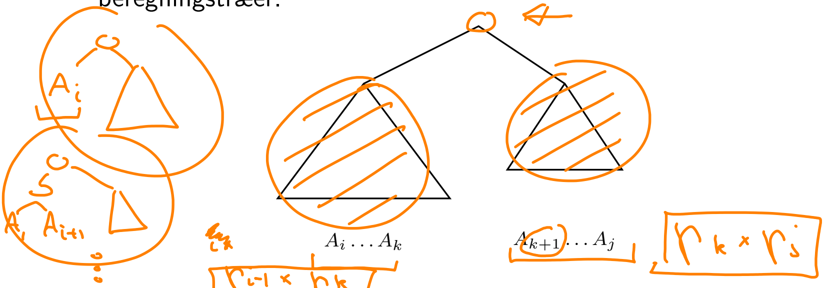
$A_1(A_2A_3)$ $(A_1A_2)A_3$ $A_1(\boxed{(A_2A_3)}A_4)$

Optimale delproblemer og rekursiv ligning

Lad $m(i,j)$ være prisen for bedste måde at gange $A_i, \ldots, A_j$ sammen på.

Observation af den essentielle egenskab:

> Undertræerne for roden af et optimalt træ må selv være optimale beregningstræer.



Prøv alle placeringer af rod, dvs. alle split $A_i, \ldots, A_k$ og $A_{k+1}, \ldots, A_j$:

$$m(i,j) = \begin{cases} 0 & \text{if } i = j \\ \min_{\boxed{i \le k < j}} \{ m(i,k) + m(k+1,j) + \boxed{p_{i-1} p_k p_j} \} & \text{if } i < j \end{cases}$$

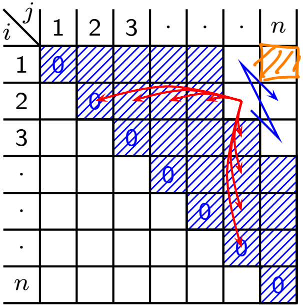
Tabel

Gentagelser blandt delproblemers delproblemer. Lav tabel og udfyld systematisk. Målet er at kende $m(1, n)$.

Tabel

Gentagelser blandt delproblemers delproblemer. Lav tabel og udfyld systematisk. Målet er at kende $m(1, n)$.

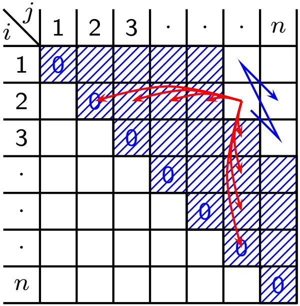
$$m(i,j) = \begin{cases} 0 & \text{if } i = j \\ \min_{i \le k < j}\{m(i,k) + m(k+1,j) + p_{i-1}p_kp_j\} & \text{if } i < j \end{cases}$$



Tabel

Gentagelser blandt delproblemers delproblemer. Lav tabel og udfyld systematisk. Målet er at kende $m(1, n)$.

$$m(i,j) = \begin{cases} 0 & \text{if } i = j \\ \min_{i \le k < j}\{m(i,k) + m(k+1,j) + p_{i-1}p_k p_j\} & \text{if } i < j \end{cases}$$



Tabelstørrelse: $n^2/2$

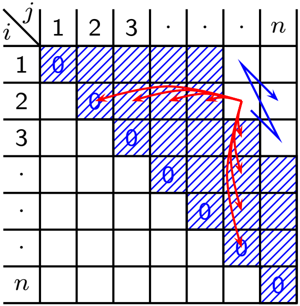
Udfyld tabelindgang: $O(\text{max størrelse af røde graf}) = O(n)$.

Tid i alt: $O(\text{produktet af de to}) = O(n^3)$.

Tabel

Gentagelser blandt delproblemers delproblemer. Lav tabel og udfyld systematisk. Målet er at kende $m(1, n)$.

$$m(i,j) = \begin{cases} 0 & \text{if } i = j \\ \min_{i \le k < j}\{m(i,k) + m(k+1,j) + p_{i-1}p_k p_j\} & \text{if } i < j \end{cases}$$



Tabelstørrelse: $n^2/2$

Udfyld tabelindgang: $O(\text{max størrelse af røde graf}) = O(n)$.

Tid i alt: $O(\text{produktet af de to}) = O(n^3)$.

Find konkret løsning: følg de optimale valg baglæns.