



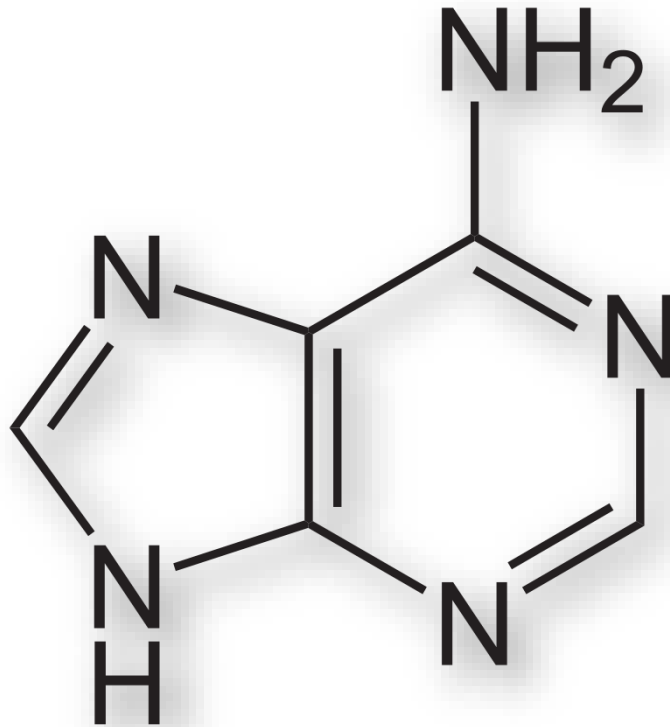
Technische
Universität
Braunschweig

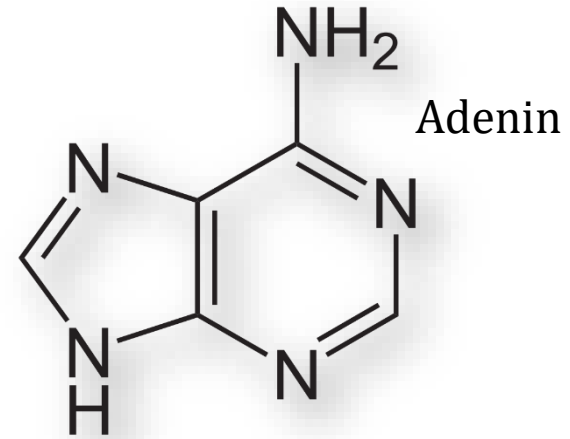
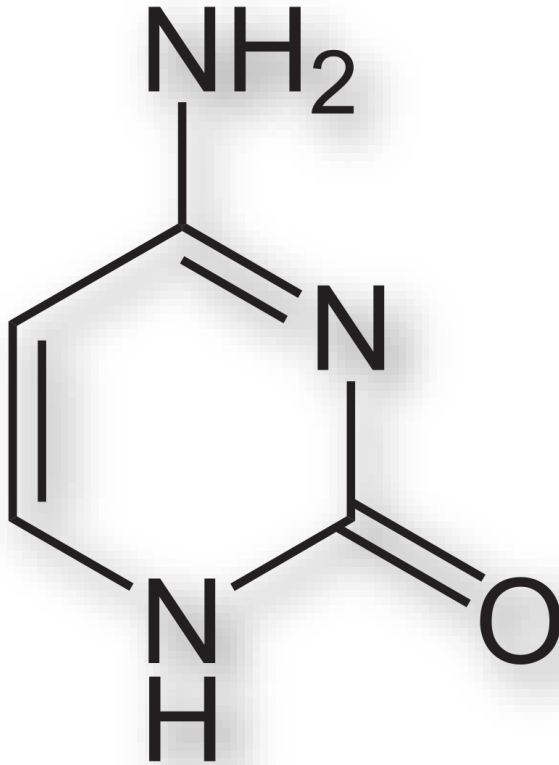


Algorithmen und Datenstrukturen 2 – Übung #2

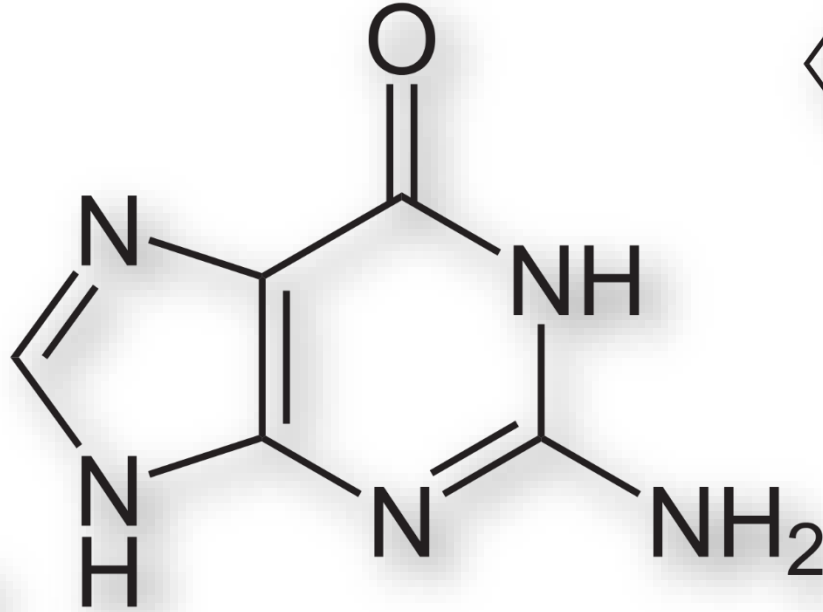
Matthias Konitzny

12.05.2021





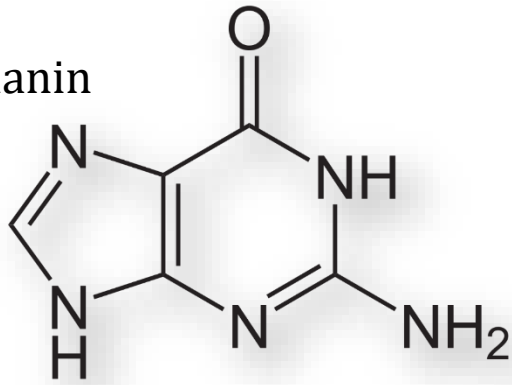
Cytosin



Adenin



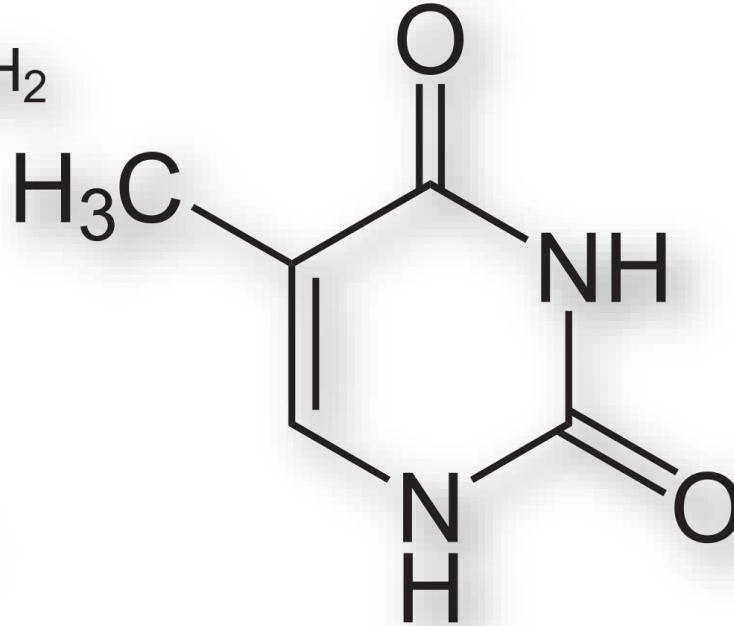
Guanin



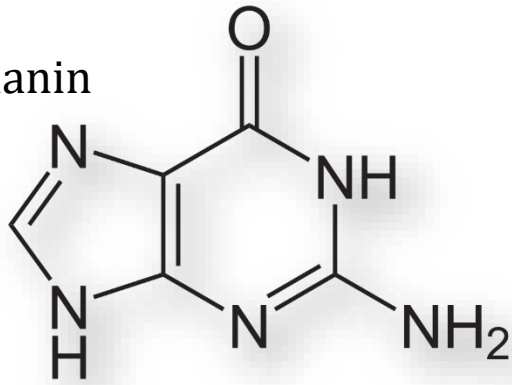
Adenin



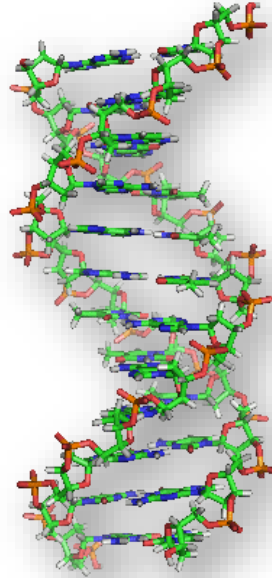
Cytosin



Guanin



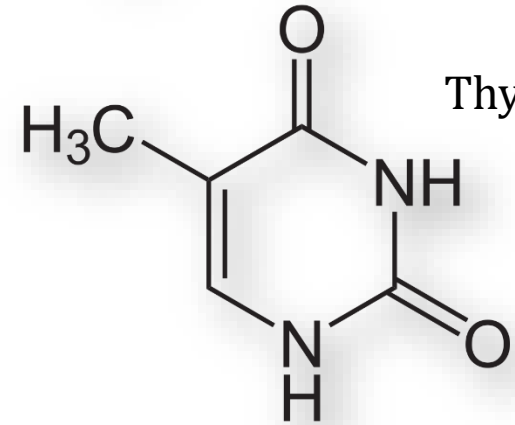
Cytosin



Adenin



Thymin



Sequenzierung von DNA Strängen

Wie ähnlich sind die
beiden Wörter?

```
CAGTAGACCAGATTGACCGATAGATGCCCTTCCCTGGCTGGTTGTGGTCCGATTCC
TG TAGACCGT GCGTCGCTTGCCTACTGTTTACAGCGCCACTGACGCTAGTCATTAT
TGAGCACGCGGCCATAACAACCCTTACTCTTCTAAGCCTGGATCTCTAAGGTTGACT
CCAAGAAGATGTAAAGTACTCCGTTAGCTCCGAAAAATTTAATTGGGACCATACG
AGCACTTTAGCACTTCTAGTTCTCATTTTCGGCTAAGCGACCCCGAGAATACTTTTC
TG TCAACGACTGGACCTAGTGGGTCTATTTACTCATCGCCTGGCGGT CAGAGCCGT
GGATAGTTT GCGCTCGTTGGTAAGTGAACGGAGATAAAGGTGGTACCGAAGTATC
GGAAGGAATATACATCAATAGA ACTGAAGTATATCGGCTTCTGTGCGCCAATACAGT
ACCTTGAGCCGGATCGAAAGCGATTTGTGTGCGAACTAGGATGGATCTATTGTT
```

```
CTTGCTCATTGGAGTTGACAGAGTACTGTTTCACATCTGATCAAGGTTATGCTAGC
ACGTCCAATGCAGGATAACTCAATATGAACTCCTTATAAGGCGATGAATTTGTTT
CTATGGTTGCCACGCAGCTCTTGGTCGGGTCAGAAGGGGTTTCCTAGGTGTGGCG
TCATGTCCTTTCTGCGGCCACAGGCGTTTGTGGTGGATCTGCACCACGTGGGTGT
CTGGCTACGCACCGTTTGTACATCTTCAAAAATCGAGCTTTGCACGGCTCAATTG
GCATAGACTGCCTGCCGTAATCTCGCTGAGTATAAGTTGATGTAATTTTCAAGACG
AGAGAGCTGGTATCCAGACAAGTCCGATGGT GAGGTTACTGAAGCGGATCCCGGA
CACTAGCTAAATATAATCGACGGATGAGACGAGGTGTAACAGGACTTTATCTCCGC
TTACGCCACACGTTCCCGGCCCTGCCGCTAGTTCCAGTTCCAATGTCCAAATGAGT
```

Sequenzierung von DNA Strängen

Wie ähnlich sind die
beiden Wörter?

```
CAGTAGACCAGATTGACCGATAGATGCCCTTCCCTGGCTGGTTGTGGTCCGATTCC
TG TAGACCGTGCGTCGCTTGCCTACTGTTTACAGCGCCACTGACCGCTAGTCATTAT
TGAGCACGCGGCCATACAACCCTTACTCTTCTAAGCCTGGATCTCTAAGGTTGACT
CCAAGAAGATGTAAAGTACTCCGTTAGCTCCGAAAAATTTAATTGGGACCATACG
AGCACTTTAGCACTTCTAGTTCTCATTTTCGGCTAAGCGACCCCGAGAATACTTTTC
TGTC AACGACTGGACCTAGTGGGTCTATTTACTCATCGCCTGGCGGTCAGAGCCGT
GGATAGTTTGGCTCGTTGGTAAGTGAACGGAGATAAAGGTGGTACCGAAGTATC
GGAAGGAATATACATCAATAGAACTGAAGTATATCGGCTTCTGTCGCCAATACAGT
ACCTTGAGCCGGATCGAAAGCGATTTGTGTGCGAACTAGGATGGATCTATTGTT
```

```
CTTGCTCATTGGAGTTGACAGAGTACTGTTTCACATCTGATCAAGGTTATGCTAGC
ACGTCCAATGCAGGATAACTCAATATGAACTCCTTATAAGGCGATGAATTTGTTT
CTATGGTTGCCACGCAGCTCTTGGTCGGGTCAGAAGGGGTTTCTAGGTGTGGCG
TCATGTCCTTTCTGCGGCCACAGGCGTTTGTGGTGGATCTGCACCACGTGGGTGT
CTGGCTACGCACCGTTTGTACATCTTCAAAAATCGAGCTTTGCACGGCTCAATTG
GCATAGACTGCCTGCCGTAATCTCGCTGAGTATAAGTTGATGTAATTTTCAAGACG
AGAGAGCTGGTATCCAGACAAGTCCGATGGTGAGGTTACTGAAGCGGATCCCGGA
CACTAGCTAAATATAATCGACGGATGAGACGAGGTGTAACAGGACTTTATCTCCGC
TTACGCCACACGTTCCCGGCCCTGCCGCTAGTTCCAGTTCCAATGTCCAATGAGT
```


Longest Common Subsequence

Gegeben:

- Alphabet Z
- Sequenzen
 - $X := x_1x_2 \dots x_n$ mit $x_i \in Z$
 - $Y := y_1y_2 \dots y_m$ mit $y_i \in Z$

Gesucht:

- Eine längstmögliche Sequenz T , die eine *Teilsequenz* von X und Y ist.

Eine Teilsequenz eines Wortes entsteht durch Weglassen von Buchstaben.

Beispiel:

ACTG

ist eine Teilsequenz von

ACCTATATGTT

Longest Common Subsequence

Gegeben:

- Alphabet Z
- Sequenzen
 - $X := x_1x_2 \dots x_n$ mit $x_i \in Z$
 - $Y := y_1y_2 \dots y_m$ mit $y_i \in Z$

Gesucht:

- Eine längstmögliche Sequenz T , die eine *Teilsequenz* von X und Y ist.

Eine Teilsequenz eines Wortes entsteht durch Weglassen von Buchstaben.

Beispiel:

ACTG

ist eine Teilsequenz von

ACCTATATGTT

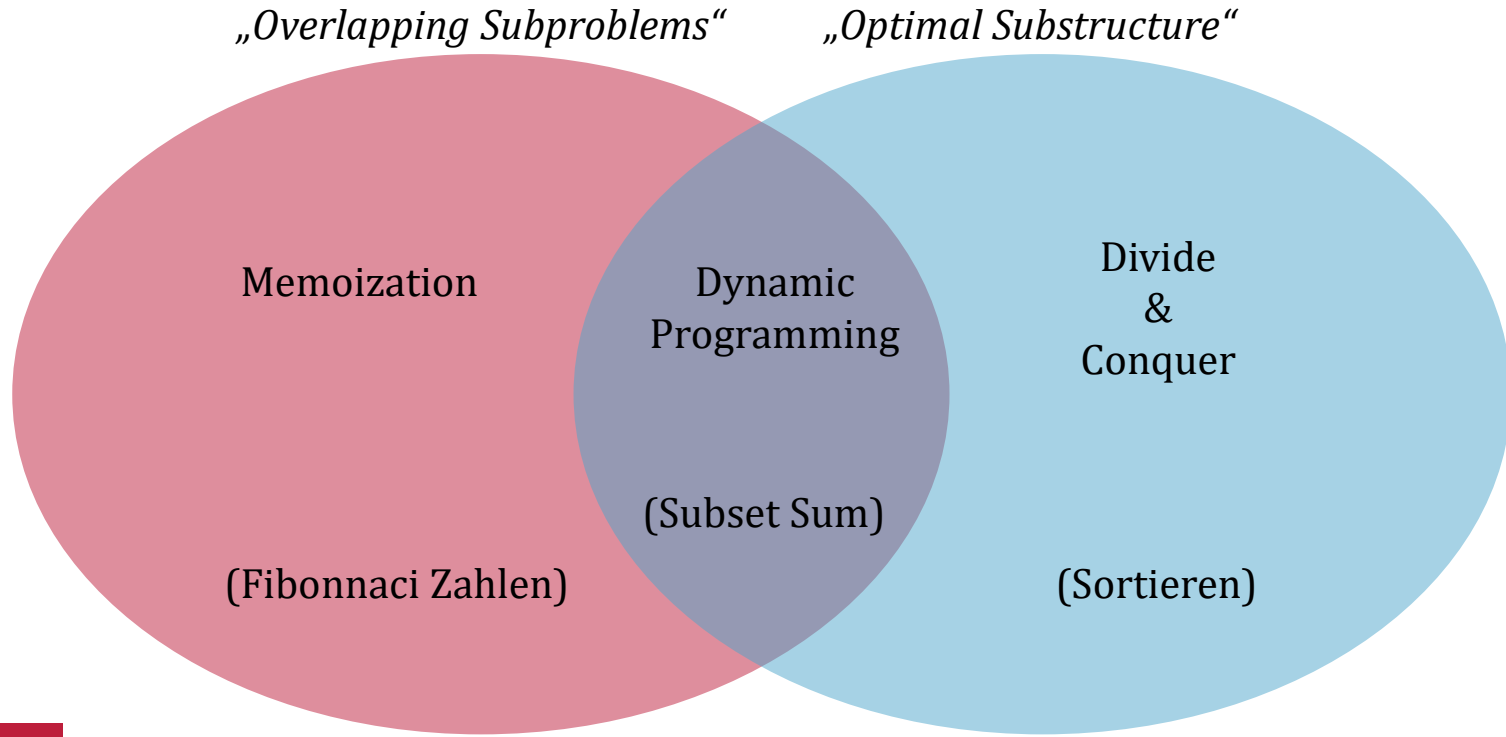
Wie lösen wir das Problem?



Greedy?

Dynamic Programming?

Dynamic Programming



Overlapping Subproblems

Fibonacci-Zahlen:

$$F_n := F_{n-1} + F_{n-2}, \quad F_0 = 0, \quad F_1 = 1$$

Input : $n \geq 0$

Output : n -te Fibonacci Zahl

if $n \leq 2$ **then**

$f \leftarrow 1$

else if $\exists \text{memo}[n]$ **then**

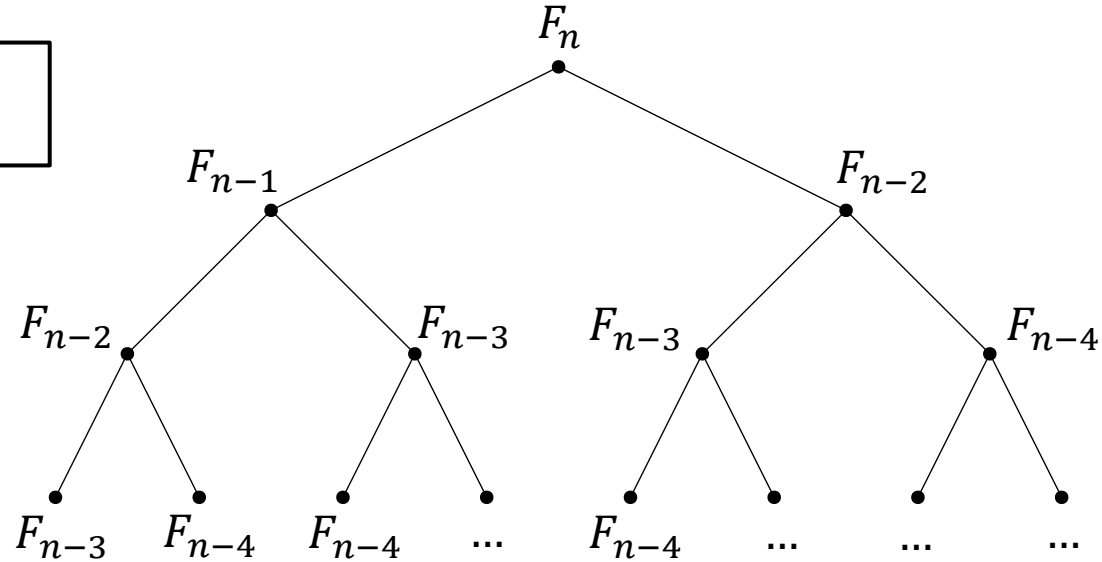
$f \leftarrow \text{memo}[n]$

else

$f \leftarrow \text{FibonacciMemoization}(n-1) + \text{FibonacciMemoization}(n-2)$

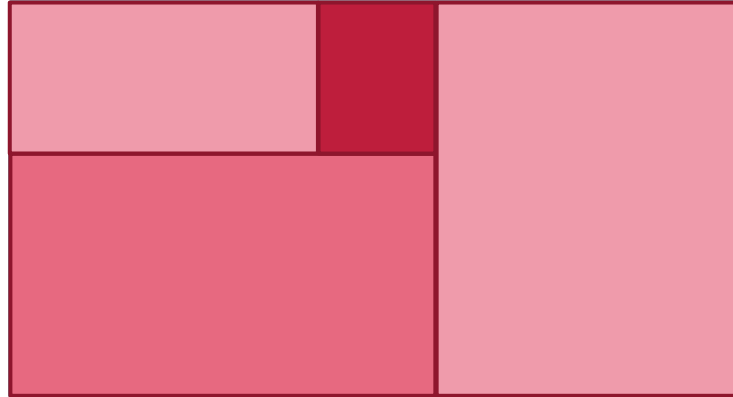
$\text{memo}[n] \leftarrow f$

return f



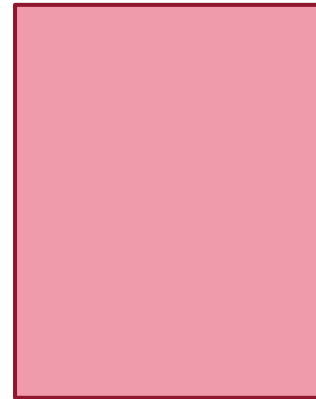
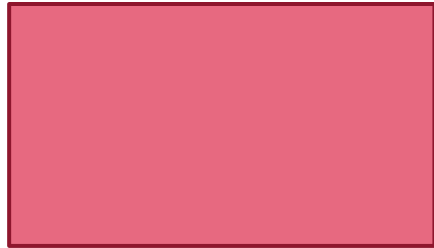
Optimal Substructure

Aufteilen in Teilprobleme



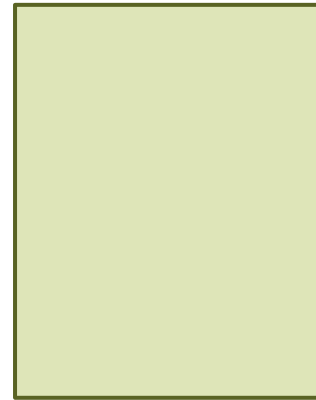
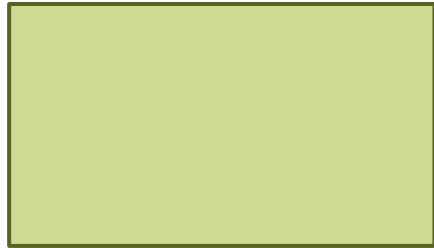
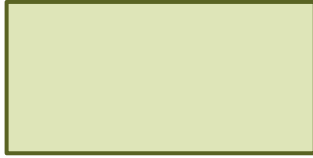
Optimal Substructure

Lösen der Teilprobleme



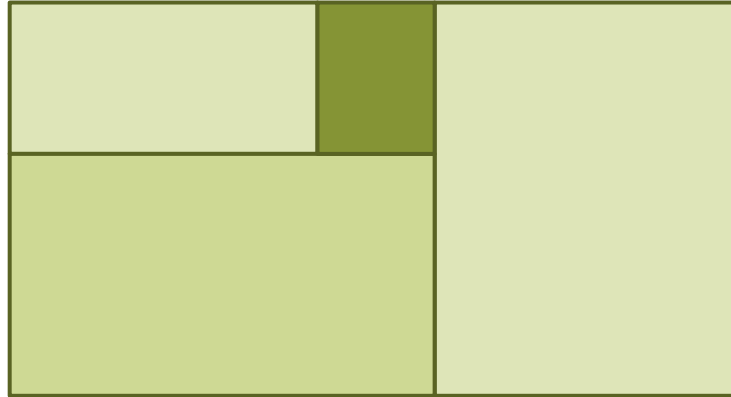
Optimal Substructure

Lösen des Hauptproblems



Optimal Substructure

Lösen des Hauptproblems

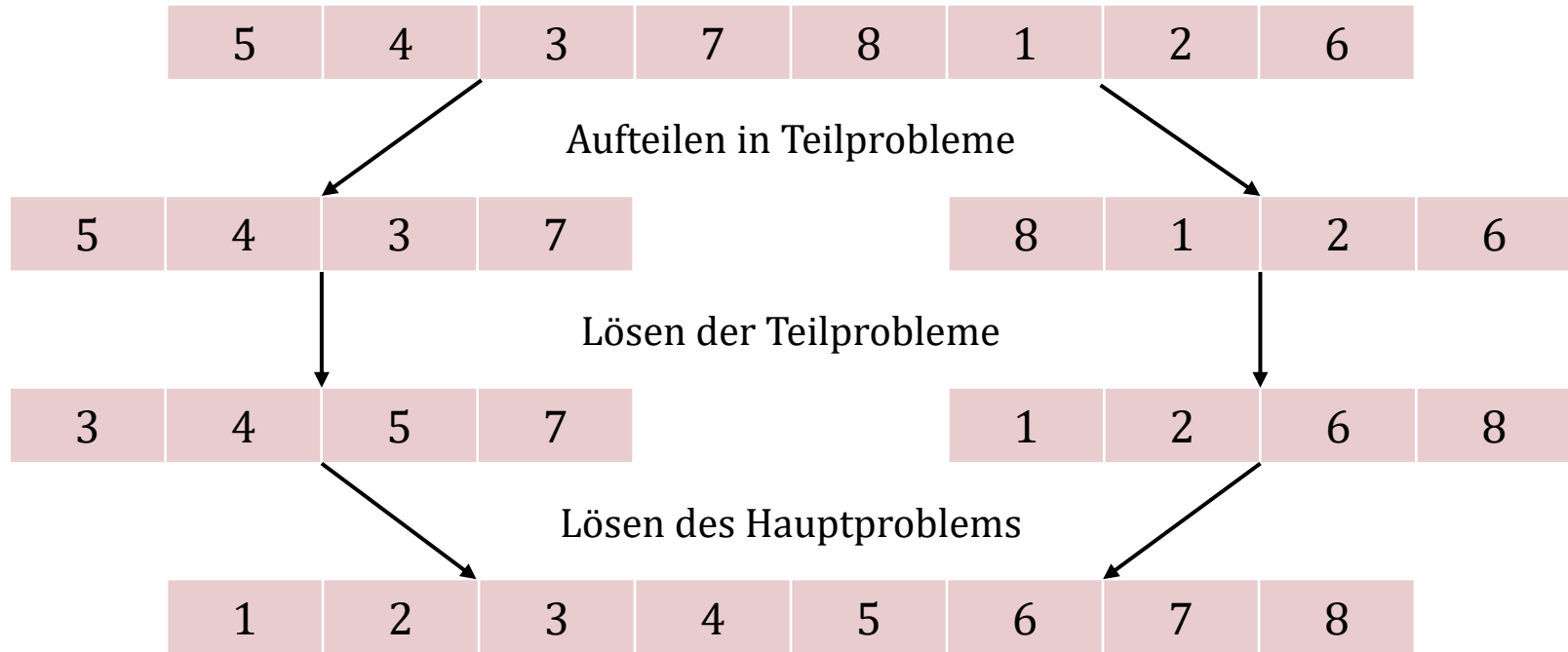


Optimal Substructure

Lösen des Hauptproblems



Optimal Substructure – Beispiel Sortieren



Longest Common Subsequence

Gegeben:

- Alphabet Z
- Sequenzen
 - $X := x_1x_2 \dots x_n$ mit $x_i \in Z$
 - $Y := y_1y_2 \dots y_m$ mit $y_j \in Z$

Gesucht:

- Eine längstmögliche Sequenz T , die eine *Teilsequenz* von X und Y ist.

Eine Teilsequenz eines Wortes entsteht durch Weglassen von Buchstaben.

Beispiel:

ACTG

ist eine Teilsequenz von

ACCTATATGTT

Was sind
Teilprobleme?



Alle
Teilsequenzen?

Die ersten
 i Buchstaben?

Dynamic Programming für LCS

Seien

- $X^i := x_1 \dots x_i$
- $Y^j := y_1 \dots y_j$

Was sind die kleinsten Wörter, die wir lösen können?

Einfach: $i = 0$ und $j = 0$

$$LCS(X^0, Y^0) = 0$$

Geht das allgemeiner?

Ja! Nur ein Wort ist leer!

Für alle $0 \leq i \leq n$ gilt

$$LCS(X^i, Y^0) = 0$$

und für alle $0 \leq j \leq m$ gilt

$$LCS(X^0, Y^j) = 0$$

Dynamic Programming für LCS

Betrachte: $x_1 \dots x_{i-1}x_i$
 $y_1 \dots y_{j-1}y_j$

$x_i \neq y_j$

Entweder x_i oder y_j ist nicht in LCS

⇓

$$LCS(X^i, Y^j) = \max(LCS(X^{i-1}, Y^j), LCS(X^i, Y^{j-1}))$$

$x_i = y_j$

Nutze den Match oder lass es sein.

⇓

$$LCS(X^i, Y^j) = \max\left(\begin{array}{l} LCS(X^{i-1}, Y^{j-1}) + 1, \\ LCS(X^{i-1}, Y^j), \\ LCS(X^i, Y^{j-1}) \end{array}\right)$$

Braucht man die?

Dynamic Programming für LCS

$$LCS(X^i, Y^j) = \begin{cases} 0 & , \text{ falls } i = 0 \text{ oder } j = 0 \\ LCS(X^{i-1}, Y^{j-1}) + 1 & , \text{ falls } x_i = y_j \\ \max(LCS(X^{i-1}, Y^j), LCS(X^i, Y^{j-1})) & , \text{ sonst} \end{cases}$$

Dynamic Programming für LCS

$$LCS(X^i, Y^j) = \begin{cases} 0 & , \text{ falls } i = 0 \text{ oder } j = 0 \\ LCS(X^{i-1}, Y^{j-1}) + 1 & , \text{ falls } x_i = y_j \\ \max(LCS(X^{i-1}, Y^j), LCS(X^i, Y^{j-1})) & , \text{ sonst} \end{cases}$$

Y \ X	∅	A	C	C	T	A	T	A	T	G	T
∅											
C											
A											
T											
G											
A											
C											
A											
T											
T											

Dynamic Programming für LCS

$$LCS(X^i, Y^j) = \begin{cases} 0 & , \text{ falls } i = 0 \text{ oder } j = 0 \\ LCS(X^{i-1}, Y^{j-1}) + 1 & , \text{ falls } x_i = y_j \\ \max(LCS(X^{i-1}, Y^j), LCS(X^i, Y^{j-1})) & , \text{ sonst} \end{cases}$$

Y \ X	∅	A	C	C	T	A	T	A	T	G	T
∅	0	0	0	0	0	0	0	0	0	0	0
C	0										
A	0										
T	0										
G	0										
A	0										
C	0										
A	0										
T	0										
T	0										

Dynamic Programming für LCS

$$LCS(X^i, Y^j) = \begin{cases} 0 & , \text{ falls } i = 0 \text{ oder } j = 0 \\ LCS(X^{i-1}, Y^{j-1}) + 1 & , \text{ falls } x_i = y_j \\ \max(LCS(X^{i-1}, Y^j), LCS(X^i, Y^{j-1})) & , \text{ sonst} \end{cases}$$

Y \ X	∅	A	C	C	T	A	T	A	T	G	T
∅	0	0	0	0	0	0	0	0	0	0	0
C	0	0									
A	0										
T	0										
G	0										
A	0										
C	0										
A	0										
T	0										
T	0										

Dynamic Programming für LCS

$$LCS(X^i, Y^j) = \begin{cases} 0 & , \text{ falls } i = 0 \text{ oder } j = 0 \\ LCS(X^{i-1}, Y^{j-1}) + 1 & , \text{ falls } x_i = y_j \\ \max(LCS(X^{i-1}, Y^j), LCS(X^i, Y^{j-1})) & , \text{ sonst} \end{cases}$$

Y \ X	∅	A	C	C	T	A	T	A	T	G	T
∅	0	0	0	0	0	0	0	0	0	0	0
C	0	0	1								
A	0										
T	0										
G	0										
A	0										
C	0										
A	0										
T	0										
T	0										

Dynamic Programming für LCS

$$LCS(X^i, Y^j) = \begin{cases} 0 & , \text{ falls } i = 0 \text{ oder } j = 0 \\ LCS(X^{i-1}, Y^{j-1}) + 1 & , \text{ falls } x_i = y_j \\ \max(LCS(X^{i-1}, Y^j), LCS(X^i, Y^{j-1})) & , \text{ sonst} \end{cases}$$

Y \ X	∅	A	C	C	T	A	T	A	T	G	T
∅	0	0	0	0	0	0	0	0	0	0	0
C	0	0	1	1							
A	0										
T	0										
G	0										
A	0										
C	0										
A	0										
T	0										
T	0										

Dynamic Programming für LCS

$$LCS(X^i, Y^j) = \begin{cases} 0 & , \text{ falls } i = 0 \text{ oder } j = 0 \\ LCS(X^{i-1}, Y^{j-1}) + 1 & , \text{ falls } x_i = y_j \\ \max(LCS(X^{i-1}, Y^j), LCS(X^i, Y^{j-1})) & , \text{ sonst} \end{cases}$$

Y \ X	∅	A	C	C	T	A	T	A	T	G	T
∅	0	0	0	0	0	0	0	0	0	0	0
C	0	0	1	1	1						
A	0										
T	0										
G	0										
A	0										
C	0										
A	0										
T	0										
T	0										

Dynamic Programming für LCS

$$LCS(X^i, Y^j) = \begin{cases} 0 & , \text{ falls } i = 0 \text{ oder } j = 0 \\ LCS(X^{i-1}, Y^{j-1}) + 1 & , \text{ falls } x_i = y_j \\ \max(LCS(X^{i-1}, Y^j), LCS(X^i, Y^{j-1})) & , \text{ sonst} \end{cases}$$

Y \ X	∅	A	C	C	T	A	T	A	T	G	T
∅	0	0	0	0	0	0	0	0	0	0	0
C	0	0	1	1	1	1	1	1	1	1	1
A	0										
T	0										
G	0										
A	0										
C	0										
A	0										
T	0										
T	0										

Dynamic Programming für LCS

$$LCS(X^i, Y^j) = \begin{cases} 0 & , \text{ falls } i = 0 \text{ oder } j = 0 \\ LCS(X^{i-1}, Y^{j-1}) + 1 & , \text{ falls } x_i = y_j \\ \max(LCS(X^{i-1}, Y^j), LCS(X^i, Y^{j-1})) & , \text{ sonst} \end{cases}$$

Y \ X	∅	A	C	C	T	A	T	A	T	G	T
∅	0	0	0	0	0	0	0	0	0	0	0
C	0	0	1	1	1	1	1	1	1	1	1
A	0	1	1	1	1	2	2	2	2	2	2
T	0										
G	0										
A	0										
C	0										
A	0										
T	0										
T	0										

Dynamic Programming für LCS

$$LCS(X^i, Y^j) = \begin{cases} 0 & , \text{ falls } i = 0 \text{ oder } j = 0 \\ LCS(X^{i-1}, Y^{j-1}) + 1 & , \text{ falls } x_i = y_j \\ \max(LCS(X^{i-1}, Y^j), LCS(X^i, Y^{j-1})) & , \text{ sonst} \end{cases}$$

$Y \backslash X$	\emptyset	A	C	C	T	A	T	A	T	G	T
\emptyset	0	0	0	0	0	0	0	0	0	0	0
C	0	0	1	1	1	1	1	1	1	1	1
A	0	1	1	1	1	2	2	2	2	2	2
T	0	1	1	1	2	2	3	3	3	3	3
G	0										
A	0										
C	0										
A	0										
T	0										
T	0										

Dynamic Programming für LCS

$$LCS(X^i, Y^j) = \begin{cases} 0 & , \text{ falls } i = 0 \text{ oder } j = 0 \\ LCS(X^{i-1}, Y^{j-1}) + 1 & , \text{ falls } x_i = y_j \\ \max(LCS(X^{i-1}, Y^j), LCS(X^i, Y^{j-1})) & , \text{ sonst} \end{cases}$$

$Y \backslash X$	\emptyset	A	C	C	T	A	T	A	T	G	T
\emptyset	0	0	0	0	0	0	0	0	0	0	0
C	0	0	1	1	1	1	1	1	1	1	1
A	0	1	1	1	1	2	2	2	2	2	2
T	0	1	1	1	2	2	3	3	3	3	3
G	0	1	1	1	2	2	3	3	3	4	4
A	0										
C	0										
A	0										
T	0										
T	0										

Dynamic Programming für LCS

$$LCS(X^i, Y^j) = \begin{cases} 0 & , \text{ falls } i = 0 \text{ oder } j = 0 \\ LCS(X^{i-1}, Y^{j-1}) + 1 & , \text{ falls } x_i = y_j \\ \max(LCS(X^{i-1}, Y^j), LCS(X^i, Y^{j-1})) & , \text{ sonst} \end{cases}$$

$Y \backslash X$	\emptyset	A	C	C	T	A	T	A	T	G	T
\emptyset	0	0	0	0	0	0	0	0	0	0	0
C	0	0	1	1	1	1	1	1	1	1	1
A	0	1	1	1	1	2	2	2	2	2	2
T	0	1	1	1	2	2	3	3	3	3	3
G	0	1	1	1	2	2	3	3	3	4	4
A	0	1	1	1	2	3	3	4	4	4	4
C	0										
A	0										
T	0										
T	0										

Dynamic Programming für LCS

$$LCS(X^i, Y^j) = \begin{cases} 0 & , \text{ falls } i = 0 \text{ oder } j = 0 \\ LCS(X^{i-1}, Y^{j-1}) + 1 & , \text{ falls } x_i = y_j \\ \max(LCS(X^{i-1}, Y^j), LCS(X^i, Y^{j-1})) & , \text{ sonst} \end{cases}$$

$Y \backslash X$	\emptyset	A	C	C	T	A	T	A	T	G	T
\emptyset	0	0	0	0	0	0	0	0	0	0	0
C	0	0	1	1	1	1	1	1	1	1	1
A	0	1	1	1	1	2	2	2	2	2	2
T	0	1	1	1	2	2	3	3	3	3	3
G	0	1	1	1	2	2	3	3	3	4	4
A	0	1	1	1	2	3	3	4	4	4	4
C	0	1	2	2	2	3	3	4	4	4	4
A	0										
T	0										
T	0										

Dynamic Programming für LCS

$$LCS(X^i, Y^j) = \begin{cases} 0 & , \text{ falls } i = 0 \text{ oder } j = 0 \\ LCS(X^{i-1}, Y^{j-1}) + 1 & , \text{ falls } x_i = y_j \\ \max(LCS(X^{i-1}, Y^j), LCS(X^i, Y^{j-1})) & , \text{ sonst} \end{cases}$$

$Y \backslash X$	\emptyset	A	C	C	T	A	T	A	T	G	T
\emptyset	0	0	0	0	0	0	0	0	0	0	0
C	0	0	1	1	1	1	1	1	1	1	1
A	0	1	1	1	1	2	2	2	2	2	2
T	0	1	1	1	2	2	3	3	3	3	3
G	0	1	1	1	2	2	3	3	3	4	4
A	0	1	1	1	2	3	3	4	4	4	4
C	0	1	2	2	2	3	3	4	4	4	4
A	0	1	2	2	2	3	3	4	4	4	4
T	0										
T	0										

Dynamic Programming für LCS

$$LCS(X^i, Y^j) = \begin{cases} 0 & , \text{ falls } i = 0 \text{ oder } j = 0 \\ LCS(X^{i-1}, Y^{j-1}) + 1 & , \text{ falls } x_i = y_j \\ \max(LCS(X^{i-1}, Y^j), LCS(X^i, Y^{j-1})) & , \text{ sonst} \end{cases}$$

$Y \backslash X$	\emptyset	A	C	C	T	A	T	A	T	G	T
\emptyset	0	0	0	0	0	0	0	0	0	0	0
C	0	0	1	1	1	1	1	1	1	1	1
A	0	1	1	1	1	2	2	2	2	2	2
T	0	1	1	1	2	2	3	3	3	3	3
G	0	1	1	1	2	2	3	3	3	4	4
A	0	1	1	1	2	3	3	4	4	4	4
C	0	1	2	2	2	3	3	4	4	4	4
A	0	1	2	2	2	3	3	4	4	4	4
T	0	1	2	2	3	3	4	4	5	5	5
T	0										

Dynamic Programming für LCS

$$LCS(X^i, Y^j) = \begin{cases} 0 & , \text{ falls } i = 0 \text{ oder } j = 0 \\ LCS(X^{i-1}, Y^{j-1}) + 1 & , \text{ falls } x_i = y_j \\ \max(LCS(X^{i-1}, Y^j), LCS(X^i, Y^{j-1})) & , \text{ sonst} \end{cases}$$

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

Dynamic Programming für LCS

$$LCS(X^i, Y^j) = \begin{cases} 0 & , \text{ falls } i = 0 \text{ oder } j = 0 \\ LCS(X^{i-1}, Y^{j-1}) + 1 & , \text{ falls } x_i = y_j \\ \max(LCS(X^{i-1}, Y^j), LCS(X^i, Y^{j-1})) & , \text{ sonst} \end{cases}$$

T

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

Dynamic Programming für LCS

$$LCS(X^i, Y^j) = \begin{cases} 0 & , \text{ falls } i = 0 \text{ oder } j = 0 \\ LCS(X^{i-1}, Y^{j-1}) + 1 & , \text{ falls } x_i = y_j \\ \max(LCS(X^{i-1}, Y^j), LCS(X^i, Y^{j-1})) & , \text{ sonst} \end{cases}$$

T

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

Dynamic Programming für LCS

$$LCS(X^i, Y^j) = \begin{cases} 0 & , \text{ falls } i = 0 \text{ oder } j = 0 \\ LCS(X^{i-1}, Y^{j-1}) + 1 & , \text{ falls } x_i = y_j \\ \max(LCS(X^{i-1}, Y^j), LCS(X^i, Y^{j-1})) & , \text{ sonst} \end{cases}$$

TT

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

Dynamic Programming für LCS

$$LCS(X^i, Y^j) = \begin{cases} 0 & , \text{ falls } i = 0 \text{ oder } j = 0 \\ LCS(X^{i-1}, Y^{j-1}) + 1 & , \text{ falls } x_i = y_j \\ \max(LCS(X^{i-1}, Y^j), LCS(X^i, Y^{j-1})) & , \text{ sonst} \end{cases}$$

TTA

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

Dynamic Programming für LCS

$$LCS(X^i, Y^j) = \begin{cases} 0 & , \text{ falls } i = 0 \text{ oder } j = 0 \\ LCS(X^{i-1}, Y^{j-1}) + 1 & , \text{ falls } x_i = y_j \\ \max(LCS(X^{i-1}, Y^j), LCS(X^i, Y^{j-1})) & , \text{ sonst} \end{cases}$$

TTA

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

Dynamic Programming für LCS

$$LCS(X^i, Y^j) = \begin{cases} 0 & , \text{ falls } i = 0 \text{ oder } j = 0 \\ LCS(X^{i-1}, Y^{j-1}) + 1 & , \text{ falls } x_i = y_j \\ \max(LCS(X^{i-1}, Y^j), LCS(X^i, Y^{j-1})) & , \text{ sonst} \end{cases}$$

TTA

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

Dynamic Programming für LCS

$$LCS(X^i, Y^j) = \begin{cases} 0 & , \text{ falls } i = 0 \text{ oder } j = 0 \\ LCS(X^{i-1}, Y^{j-1}) + 1 & , \text{ falls } x_i = y_j \\ \max(LCS(X^{i-1}, Y^j), LCS(X^i, Y^{j-1})) & , \text{ sonst} \end{cases}$$

TTA

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

Dynamic Programming für LCS

$$LCS(X^i, Y^j) = \begin{cases} 0 & , \text{ falls } i = 0 \text{ oder } j = 0 \\ LCS(X^{i-1}, Y^{j-1}) + 1 & , \text{ falls } x_i = y_j \\ \max(LCS(X^{i-1}, Y^j), LCS(X^i, Y^{j-1})) & , \text{ sonst} \end{cases}$$

TTAT

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

Dynamic Programming für LCS

$$LCS(X^i, Y^j) = \begin{cases} 0 & , \text{ falls } i = 0 \text{ oder } j = 0 \\ LCS(X^{i-1}, Y^{j-1}) + 1 & , \text{ falls } x_i = y_j \\ \max(LCS(X^{i-1}, Y^j), LCS(X^i, Y^{j-1})) & , \text{ sonst} \end{cases}$$

TTATA

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

Dynamic Programming für LCS

$$LCS(X^i, Y^j) = \begin{cases} 0 & , \text{ falls } i = 0 \text{ oder } j = 0 \\ LCS(X^{i-1}, Y^{j-1}) + 1 & , \text{ falls } x_i = y_j \\ \max(LCS(X^{i-1}, Y^j), LCS(X^i, Y^{j-1})) & , \text{ sonst} \end{cases}$$

TTATA

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

Dynamic Programming für LCS

$$LCS(X^i, Y^j) = \begin{cases} 0 & , \text{ falls } i = 0 \text{ oder } j = 0 \\ LCS(X^{i-1}, Y^{j-1}) + 1 & , \text{ falls } x_i = y_j \\ \max(LCS(X^{i-1}, Y^j), LCS(X^i, Y^{j-1})) & , \text{ sonst} \end{cases}$$

TTATAC

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

Dynamic Programming für LCS

$$LCS(X^i, Y^j) = \begin{cases} 0 & , \text{ falls } i = 0 \text{ oder } j = 0 \\ LCS(X^{i-1}, Y^{j-1}) + 1 & , \text{ falls } x_i = y_j \\ \max(LCS(X^{i-1}, Y^j), LCS(X^i, Y^{j-1})) & , \text{ sonst} \end{cases}$$

TTATAC

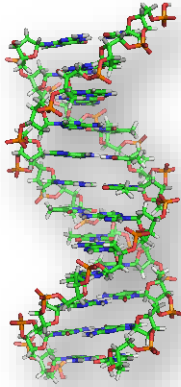
Y \ X	∅	A	C	C	T	A	T	A	T	G	T
∅	0	0	0	0	0	0	0	0	0	0	0
C	0	0	1	1	1	1	1	1	1	1	1
A	0	1	1	1	1	2	2	2	2	2	2
T	0	1	1	1	2	2	3	3	3	3	3
G	0	1	1	1	2	2	3	3	3	4	4
A	0	1	1	1	2	3	3	4	4	4	4
C	0	1	2	2	2	3	3	4	4	4	4
A	0	1	2	2	2	3	3	4	4	4	4
T	0	1	2	2	3	3	4	4	5	5	5
T	0	1	2	2	3	3	4	4	5	5	6

Dynamic Programming für LCS

$$LCS(X^i, Y^j) = \begin{cases} 0 & , \text{ falls } i = 0 \text{ oder } j = 0 \\ LCS(X^{i-1}, Y^{j-1}) + 1 & , \text{ falls } x_i = y_j \\ \max(LCS(X^{i-1}, Y^j), LCS(X^i, Y^{j-1})) & , \text{ sonst} \end{cases}$$

CATATT

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



```
CAGTAGACCAGATTGACCGATAGATGCCCTTCCCTGGCTGGTTGTGGTCCGATTCC
TG TAGACCGTGCGTCGCTTGCCTACTGTTTACAGCGCCACTGACGCTAGTCATTAT
TGAGCACGCGGCCATAACAACCCTTACTCTTCTAAGCCTGGATCTCTAAGGTTGACT
CCAAGAAGATGTAAAGTACTCCGTTAGCTCCGAAAAATTTAATTGGGACCATACG
AGCACTTTAGCACTTCTAGTTCTCATTTTCGGCTAAGCGACCCCGAGAATACTTTTC
TG TCAACGACTGGACCTAGTGGGTCTATTTACTCATCGCCTGGCGGTGAGAGCCGT
GGATAGTTTGGCTCGTTGGTAAGTGAACGGAGATAAAGGTGGTACCGAAGTATC
GGAAGGAATATACATCAATAGAACTGAAGTATATCGGCTTCTGTGCGCCAATACAGT
ACCTTGAGCCGGATCGAAAGCGATTTGTGTGCGAACTAGGATGGATCTATTGTT
```

```
CTTGCTCATTGGAGTTGACAGAGTACTGTTTCACATCTGATCAAGGTTATGCTAGC
ACGTCCAATGCAGGATAACTCAATATGAACTCCTTATAAGGCGATGAATTTGTTT
CTATGGTTGCCACGCAGCTCTTGGTCGGGTCAGAAGGGGTTTCCTAGGTGTGGCG
TCATGTCCTTTCTGCGGCCACAGGCGTTTGTGGTGGATCTGCACCACGTGGGTGT
CTGGCTACGCACCGTTTGTACATCTTCAAAAATCGAGCTTTGCACGGCTCAATTG
GCATAGACTGCCTGCCGTAATCTCGCTGAGTATAAGTTGATGTAATTTTCAAGACG
AGAGAGCTGGTATCCAGACAAGTCCGATGGTGAAGTTACTGAAGCGGATCCCGGA
CACTAGCTAAATATAATCGACGGATGAGACGAGGTGTAACAGGACTTTATCTCCGC
TTACGCCACACGTTCCCGGCCCTGCCGCTAGTTCCAGTTCCAATGTCCAAATGAGT
```

LCS: 323

Matrix Chain Multiplication

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \times \begin{pmatrix} e & f & g \\ h & i & j \end{pmatrix}$$

Wie viele Multiplikationen werden benötigt?

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \times \begin{pmatrix} e & f & g \\ h & i & j \end{pmatrix} = \begin{pmatrix} ae + bh & af + bi & ag + bj \\ ce + dh & cf + di & cg + dj \end{pmatrix}$$



$$2 \times 2 \quad \text{---} \quad 2 \times 3$$

$$2 \cdot 2 \cdot 3 = 12 \text{ Multiplikationen}$$

Matrix Chain Multiplication

$$\begin{matrix} A & & B & & C \\ (a & b) & \times & (e & f & g) & \times & \begin{pmatrix} k & l \\ m & n \\ o & p \end{pmatrix} \\ (c & d) & & (h & i & j) & & \end{matrix}$$

Wie viele Multiplikationen werden benötigt?

Operation	Dimensionen	Multiplikationen
$A \times B$	$(2 \times 2) \times (2 \times 3)$	12
$(A \times B) \times C$	$(2 \times 3) \times (3 \times 2)$	12
		24

Alternative

Operation	Dimensionen	Multiplikationen
$B \times C$	$(2 \times 3) \times (3 \times 2)$	12
$A \times (B \times C)$	$(2 \times 2) \times (2 \times 2)$	10
		22

Matrix Chain Multiplication

Gegeben:

- Matrizen $M_1 \times M_2 \times \dots \times M_n$ wobei M_i eine Matrix der Form $d_i \times d_{i+1}$ ist.

Gesucht:

- Eine Klammerung für die Multiplikation der Matrizen, sodass die Summe der benötigten Multiplikationen minimal ist.

Die Anzahl der möglichen Klammerungen von n Matrizen kann durch die Folge der Catalan-Zahlen C_n beschrieben werden.

Die Folge beginnt mit

1, 1, 2, 5, 14, 42, 132, 429, 1430, 4862, 16796,
58786, 208012,...



Dynamic Programming für Matrix Chain Multiplication

$OPT(i, j)$ ist die Anzahl Multiplikationen, die für Matrizen M_i bis M_j benötigt werden.

Was ist das kleinste Teilproblem, welches wir Lösen können?

Für $i = j$ müssen keine Multiplikationen durchgeführt werden.

Für alle $i \in \{1, \dots, n\}$ gilt
 $OPT(i, i) = 0$

Dynamic Programming für Matrix Chain Multiplication

Betrachte

$$\dots \left(\underline{(M_i \times M_{i+1} \times M_k)} \times \underline{(M_{j-1} \times M_j)} \right) \dots$$

$$M: d_i \times d_{k+1}$$

$$M': d_{k+1} \times d_{j+1}$$



Wir brauchen $d_i \cdot d_{k+1} \cdot d_{j+1}$ Multiplikationen für die Multiplikation von M und M' .

Wir brauchen insgesamt $d_i \cdot d_{k+1} \cdot d_{j+1} + OPT(i, k) + OPT(k + 1, j)$ Multiplikationen.

Dynamic Programming für Matrix Chain Multiplication

$$OPT(i, j) = \begin{cases} 0 & , \text{ falls } i = j \\ \min_{i \leq k < j} d_i \cdot d_{k+1} \cdot d_{j+1} + OPT(i, k) + OPT(k + 1, j) & , \text{ sonst} \end{cases}$$

Dynamic Programming für Matrix Chain Multiplication

$$OPT(i, j) = \begin{cases} 0 & , \text{ falls } i = j \\ \min_{i \leq k < j} d_i \cdot d_{k+1} \cdot d_{j+1} + OPT(i, k) + OPT(k + 1, j) & , \text{ sonst} \end{cases}$$

<i>i</i>	<i>d_i</i>
1	3
2	5
3	10
4	2
5	4
6	3

<i>ij</i>	<i>M₁</i>	<i>M₂</i>	<i>M₃</i>	<i>M₄</i>	<i>M₅</i>
<i>M₁</i>					
<i>M₂</i>					
<i>M₃</i>					
<i>M₄</i>					
<i>M₅</i>					

Dynamic Programming für Matrix Chain Multiplication

$$OPT(i, j) = \begin{cases} 0 & , \text{ falls } i = j \\ \min_{i \leq k < j} d_i \cdot d_{k+1} \cdot d_{j+1} + OPT(i, k) + OPT(k + 1, j) & , \text{ sonst} \end{cases}$$

<i>i</i>	<i>d_i</i>
1	3
2	5
3	10
4	2
5	4
6	3

<i>ij</i>	<i>M₁</i>	<i>M₂</i>	<i>M₃</i>	<i>M₄</i>	<i>M₅</i>
<i>M₁</i>	0				
<i>M₂</i>		0			
<i>M₃</i>			0		
<i>M₄</i>				0	
<i>M₅</i>					0

Dynamic Programming für Matrix Chain Multiplication

$$OPT(i, j) = \begin{cases} 0 & , \text{ falls } i = j \\ \min_{i \leq k < j} d_i \cdot d_{k+1} \cdot d_{j+1} + OPT(i, k) + OPT(k + 1, j) & , \text{ sonst} \end{cases}$$

<i>i</i>	<i>d_i</i>
1	3
2	5
3	10
4	2
5	4
6	3

<i>ij</i>	<i>M₁</i>	<i>M₂</i>	<i>M₃</i>	<i>M₄</i>	<i>M₅</i>
<i>M₁</i>	0				
<i>M₂</i>		0			
<i>M₃</i>			0		
<i>M₄</i>				0	
<i>M₅</i>					0

Dynamic Programming für Matrix Chain Multiplication

$$OPT(i, j) = \begin{cases} 0 & , \text{ falls } i = j \\ \min_{i \leq k < j} d_i \cdot d_{k+1} \cdot d_{j+1} + OPT(i, k) + OPT(k + 1, j) & , \text{ sonst} \end{cases}$$

$i = 1; j = 2$

- $k = 1: 3 \cdot 5 \cdot 10 + 0 + 0 = 150$

ij	M_1	M_2	M_3	M_4	M_5
M_1	0 → 150				
M_2		↑ 0			
M_3			0		
M_4				0	
M_5					0

i	d_i
1	3
2	5
3	10
4	2
5	4
6	3

Dynamic Programming für Matrix Chain Multiplication

$$OPT(i, j) = \begin{cases} 0 & , \text{ falls } i = j \\ \min_{i \leq k < j} d_i \cdot d_{k+1} \cdot d_{j+1} + OPT(i, k) + OPT(k + 1, j) & , \text{ sonst} \end{cases}$$

$i = 1; j = 2$

- $k = 1: 3 \cdot 5 \cdot 10 + 0 + 0 = 150$

$i = 2; j = 3$

- $k = 2: 5 \cdot 10 \cdot 2 + 0 + 0 = 100$

ij	M_1	M_2	M_3	M_4	M_5	
M_1	0	150				
M_2			0 → 100			
M_3				0		
M_4					0	
M_5						0

i	d_i
1	3
2	5
3	10
4	2
5	4
6	3

Dynamic Programming für Matrix Chain Multiplication

$$OPT(i, j) = \begin{cases} 0 & , \text{ falls } i = j \\ \min_{i \leq k < j} d_i \cdot d_{k+1} \cdot d_{j+1} + OPT(i, k) + OPT(k + 1, j) & , \text{ sonst} \end{cases}$$

$i = 1; j = 2$

- $k = 1: 3 \cdot 5 \cdot 10 + 0 + 0 = 150$

$i = 2; j = 3$

- $k = 2: 5 \cdot 10 \cdot 2 + 0 + 0 = 100$

$i = 3; j = 4$

- $k = 3: 10 \cdot 2 \cdot 4 + 0 + 0 = 80$

ij	M_1	M_2	M_3	M_4	M_5
M_1	0	150			
M_2		0	100		
M_3			0 → 80		
M_4				↑ 0	
M_5					0

i	d_i
1	3
2	5
3	10
4	2
5	4
6	3

Dynamic Programming für Matrix Chain Multiplication

$$OPT(i, j) = \begin{cases} 0 & , \text{ falls } i = j \\ \min_{i \leq k < j} d_i \cdot d_{k+1} \cdot d_{j+1} + OPT(i, k) + OPT(k + 1, j) & , \text{ sonst} \end{cases}$$

$i = 1; j = 2$

- $k = 1: 3 \cdot 5 \cdot 10 + 0 + 0 = 150$

$i = 2; j = 3$

- $k = 2: 5 \cdot 10 \cdot 2 + 0 + 0 = 100$

$i = 3; j = 4$

- $k = 3: 10 \cdot 2 \cdot 4 + 0 + 0 = 80$

$i = 4; j = 5$

- $k = 4: 2 \cdot 4 \cdot 3 + 0 + 0 = 24$

ij	M_1	M_2	M_3	M_4	M_5
M_1	0	150			
M_2		0	100		
M_3			0	80	
M_4				0	24
M_5					0

i	d_i
1	3
2	5
3	10
4	2
5	4
6	3

Dynamic Programming für Matrix Chain Multiplication

$$OPT(i, j) = \begin{cases} 0 & , \text{ falls } i = j \\ \min_{i \leq k < j} d_i \cdot d_{k+1} \cdot d_{j+1} + OPT(i, k) + OPT(k + 1, j) & , \text{ sonst} \end{cases}$$

$i = 1; j = 3$

- $k = 1: 3 \cdot 5 \cdot 2 + 0 + 100 = 130$
- $k = 2: 3 \cdot 10 \cdot 2 + 150 + 0 = 210$

ij	M_1	M_2	M_3	M_4	M_5
M_1	0	150	130		
M_2		0	100	140	
M_3			0	80	
M_4				0	24
M_5					0

i	d_i
1	3
2	5
3	10
4	2
5	4
6	3

Dynamic Programming für Matrix Chain Multiplication

$$OPT(i, j) = \begin{cases} 0 & , \text{ falls } i = j \\ \min_{i \leq k < j} d_i \cdot d_{k+1} \cdot d_{j+1} + OPT(i, k) + OPT(k + 1, j) & , \text{ sonst} \end{cases}$$

$i = 1; j = 3$

- $k = 1: 3 \cdot 5 \cdot 2 + 0 + 100 = 130$
- $k = 2: 3 \cdot 10 \cdot 2 + 150 + 0 = 210$

$i = 2; j = 4$

- $k = 2: 5 \cdot 10 \cdot 4 + 0 + 80 = 280$
- $k = 3: 5 \cdot 2 \cdot 4 + 100 + 0 = 140$

ij	M_1	M_2	M_3	M_4	M_5
M_1	0	150	130		
M_2		0	100	140	
M_3			0	80	
M_4				0	24
M_5					0

Diagram illustrating the calculation of the minimum cost for matrix chain multiplication. The table shows the cost for subproblems M_1 to M_5 . Arrows indicate the optimal split point k for each subproblem: a horizontal arrow from M_2 to M_3 (at $k=2$), a vertical arrow from M_3 to M_4 (at $k=3$), and a diagonal arrow from M_4 to M_5 (at $k=4$).

i	d_i
1	3
2	5
3	10
4	2
5	4
6	3

Dynamic Programming für Matrix Chain Multiplication

$$OPT(i, j) = \begin{cases} 0 & , \text{ falls } i = j \\ \min_{i \leq k < j} d_i \cdot d_{k+1} \cdot d_{j+1} + OPT(i, k) + OPT(k + 1, j) & , \text{ sonst} \end{cases}$$

$i = 1; j = 3$

- $k = 1: 3 \cdot 5 \cdot 2 + 0 + 100 = 130$
- $k = 2: 3 \cdot 10 \cdot 2 + 150 + 0 = 210$

$i = 2; j = 4$

- $k = 2: 5 \cdot 10 \cdot 4 + 0 + 80 = 280$
- $k = 3: 5 \cdot 2 \cdot 4 + 100 + 0 = 140$

$i = 3; j = 5$

- $k = 3: 10 \cdot 2 \cdot 3 + 0 + 24 = 84$
- $k = 4: 10 \cdot 4 \cdot 3 + 80 + 0 = 200$

ij	M_1	M_2	M_3	M_4	M_5
M_1	0	150	130		
M_2		0	100	140	
M_3			0	80	84
M_4				0	24
M_5					0

i	d_i
1	3
2	5
3	10
4	2
5	4
6	3

Dynamic Programming für Matrix Chain Multiplication

$$OPT(i, j) = \begin{cases} 0 & , \text{ falls } i = j \\ \min_{i \leq k < j} d_i \cdot d_{k+1} \cdot d_{j+1} + OPT(i, k) + OPT(k + 1, j) & , \text{ sonst} \end{cases}$$

$i = 1; j = 4$

- $k = 1: 3 \cdot 5 \cdot 4 + 0 + 140 = 200$
- $k = 2: 3 \cdot 10 \cdot 4 + 140 + 80 = 340$
- $k = 3: 3 \cdot 2 \cdot 4 + 130 + 0 = 154$

ij	M_1	M_2	M_3	M_4	M_5
M_1	0	150	130	154	
M_2		0	100	140	
M_3			0	80	84
M_4				0	24
M_5					0

i	d_i
1	3
2	5
3	10
4	2
5	4
6	3

Dynamic Programming für Matrix Chain Multiplication

$$OPT(i, j) = \begin{cases} 0 & , \text{ falls } i = j \\ \min_{i \leq k < j} d_i \cdot d_{k+1} \cdot d_{j+1} + OPT(i, k) + OPT(k + 1, j) & , \text{ sonst} \end{cases}$$

$i = 1; j = 4$

- $k = 1: 3 \cdot 5 \cdot 4 + 0 + 140 = 200$
- $k = 2: 3 \cdot 10 \cdot 4 + 140 + 80 = 340$
- $k = 3: 3 \cdot 2 \cdot 4 + 130 + 0 = 154$

$i = 2; j = 5$

- $k = 2: 5 \cdot 10 \cdot 3 + 0 + 84 = 234$
- $k = 3: 5 \cdot 2 \cdot 3 + 100 + 24 = 154$
- $k = 4: 5 \cdot 4 \cdot 3 + 140 + 0 = 200$

ij	M_1	M_2	M_3	M_4	M_5
M_1	0	150	130	154	
M_2		0	100	140	154
M_3			0	80	84
M_4				0	24
M_5					0

i	d_i
1	3
2	5
3	10
4	2
5	4
6	3

Dynamic Programming für Matrix Chain Multiplication

$$OPT(i, j) = \begin{cases} 0 & , \text{ falls } i = j \\ \min_{i \leq k < j} d_i \cdot d_{k+1} \cdot d_{j+1} + OPT(i, k) + OPT(k + 1, j) & , \text{ sonst} \end{cases}$$

$i = 1; j = 5$

- $k = 1: 3 \cdot 5 \cdot 3 + 0 + 154 = 199$
- $k = 2: 3 \cdot 10 \cdot 3 + 150 + 84 = 324$
- $k = 3: 3 \cdot 2 \cdot 3 + 130 + 24 = 172$
- $k = 4: 3 \cdot 4 \cdot 3 + 154 + 0 = 190$

ij	M_1	M_2	M_3	M_4	M_5
M_1	0	150	130	154	172
M_2		0	100	140	154
M_3			0	80	84
M_4				0	24
M_5					0

i	d_i
1	3
2	5
3	10
4	2
5	4
6	3

Dynamic Programming für Matrix Chain Multiplication

$$OPT(i, j) = \begin{cases} 0 & , \text{ falls } i = j \\ \min_{i \leq k < j} d_i \cdot d_{k+1} \cdot d_{j+1} + OPT(i, k) + OPT(k + 1, j) & , \text{ sonst} \end{cases}$$

$i = 1; j = 5$

- $k = 1: 3 \cdot 5 \cdot 3 + 0 + 154 = 199$
- $k = 2: 3 \cdot 10 \cdot 3 + 150 + 84 = 324$
- $k = 3: 3 \cdot 2 \cdot 3 + 130 + 24 = 172$
- $k = 4: 3 \cdot 4 \cdot 3 + 154 + 0 = 190$

Ergebnis: 172 Multiplikationen

ij	M_1	M_2	M_3	M_4	M_5
M_1	0	150	130	154	172
M_2		0	100	140	154
M_3			0	80	84
M_4				0	24
M_5					0

i	d_i
1	3
2	5
3	10
4	2
5	4
6	3

Online-Demo: <https://rosulek.github.io/vamonos/demos/matrix-chain.html>

Dynamic Programming für Matrix Chain Multiplication

$$OPT(i, j) = \begin{cases} 0 & , \text{ falls } i = j \\ \min_{i \leq k < j} d_i \cdot d_{k+1} \cdot d_{j+1} + OPT(i, k) + OPT(k + 1, j) & , \text{ sonst} \end{cases}$$

$i = 1; j = 5$

- $k = 1: 3 \cdot 5 \cdot 3 + 0 + 154 = 199$
- $k = 2: 3 \cdot 10 \cdot 3 + 150 + 84 = 324$
- $k = 3: 3 \cdot 2 \cdot 3 + 130 + 24 = 172$
- $k = 4: 3 \cdot 4 \cdot 3 + 154 + 0 = 190$

Ergebnis: 172 Multiplikationen

Reihenfolge: $(M_1 \times (M_2 \times M_3)) \times (M_4 \times M_5)$

ij	M_1	M_2	M_3	M_4	M_5
M_1	0	150	130	154	172
M_2		0	100	140	154
M_3			0	80	84
M_4				0	24
M_5					0

i	d_i
1	3
2	5
3	10
4	2
5	4
6	3

Online-Demo: <https://rosulek.github.io/vamonos/demos/matrix-chain.html>

Dynamic Programming

1. Geeignete Teilprobleme definieren
2. Lösung für kleinste Teilprobleme definieren (Initialisierung)
3. Lösung für Problem aus gelösten Teilprobleme bestimmen (Rekursionsgleichung)
4. Algorithmus schreiben (wie läuft man durch die Tabelle?)

Weitere Fragen:

- Wie bekommt man die Lösung und nicht nur den Wert?
- Gibt es mehrere mögliche dynamische Programme?
- Was ist mit der Effizienz?

