

Algoritmer og Datastrukturer 2

Gerth Stølting Brodal



Algoritmer og Datastrukturer 2

Algoritme Design Teknikker
(2 uger)

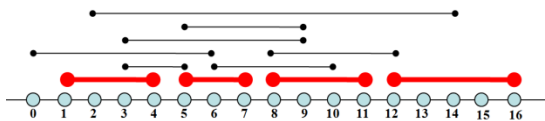
Del-og-kombiner

$$\begin{pmatrix} I & J \\ K & L \end{pmatrix} = \begin{pmatrix} A & B \\ C & D \end{pmatrix} \begin{pmatrix} E & F \\ G & H \end{pmatrix}$$

Dynamisk programmering

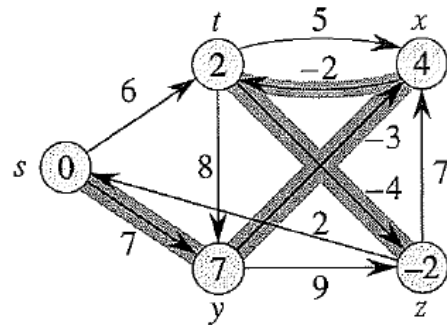
j	0	1	2	3	4	5	6
i	y_j	B	D	C	A	B	A
0	x_i	0	0	0	0	0	0
1	A	0	0	0	1	1	1
2	B	0	1	1	1	2	2
3	C	0	1	1	2	2	2
4	B	0	1	1	2	2	3
5	D	0	1	2	2	2	3
6	A	0	1	2	2	3	4
7	B	0	1	2	2	3	4

Grådige algoritmer

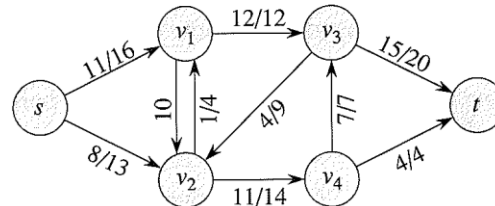


Graf-algoritmer
(3 uge)

Korteste veje

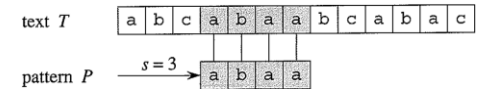


Maksimale strømninger

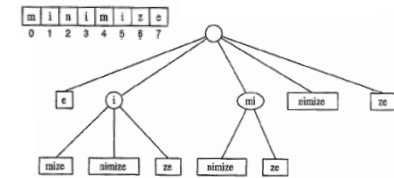


Streng-algoritmer
(1 uge)

Mønstergenkendelse



Suffix-træer



Suffix arrays



Algoritmer og Datastrukturer 2

Gerth Stølting Brodal

Del-og-kombiner

[CLRS, kapitel 2.3, 4.2-4.5, problem 30.1.c]

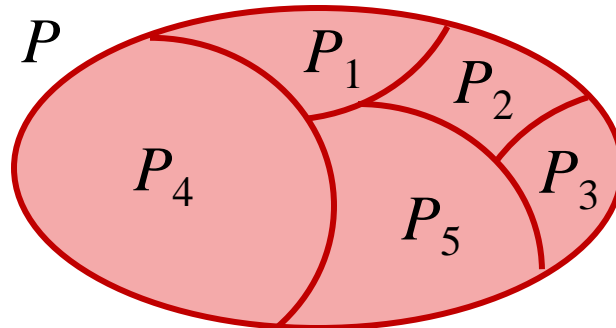


Del-og-Kombiner

Algoritme design teknik

Virker for mange problemer (men langt fra alle)

- **Opdel** et problem P i mindre problemer P_1, \dots, P_k , der kan løses uafhængigt (små problemer løses direkte)
- Løs delproblemerne P_1, \dots, P_k **rekursivt**
- **Kombiner** løsningerne for P_1, \dots, P_k til en løsning for P



Eksempel: Merge-Sort

MERGE-SORT(A, p, r)

① To mindre delproblemer

if $p < r$

$q = \lfloor (p + r) / 2 \rfloor$

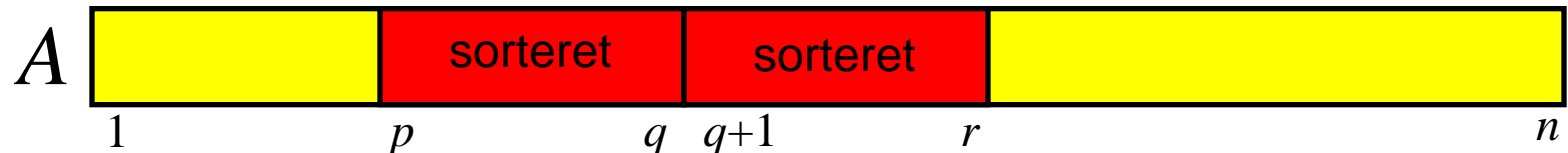
② Løs rekursivt

MERGE-SORT(A, p, q)

MERGE-SORT($A, q + 1, r$)

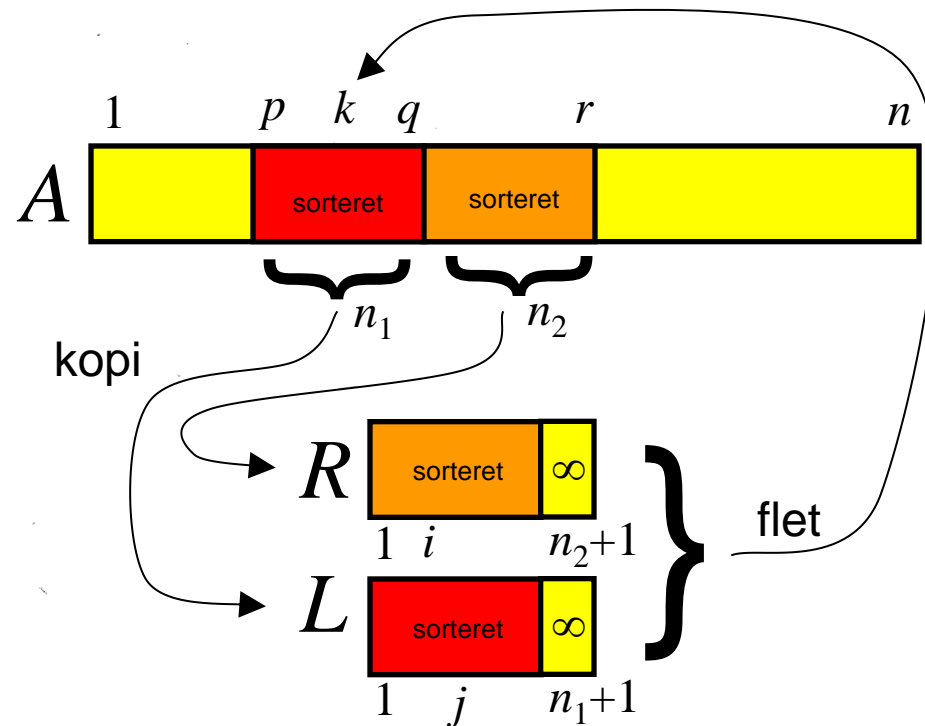
③ Kombiner

MERGE(A, p, q, r)



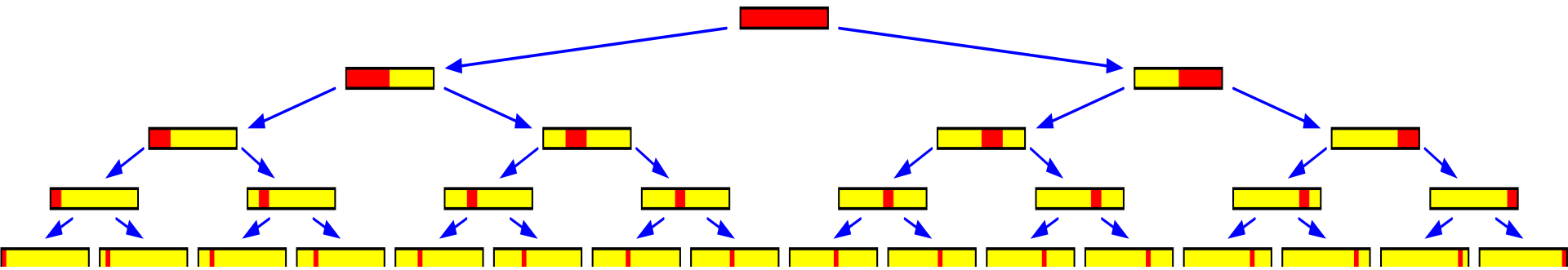
MERGE(A, p, q, r)

```
1   $n_1 = q - p + 1$ 
2   $n_2 = r - q$ 
3  let  $L[1..n_1 + 1]$  and  $R[1..n_2 + 1]$  be new arrays
4  for  $i = 1$  to  $n_1$ 
5       $L[i] = A[p + i - 1]$ 
6  for  $j = 1$  to  $n_2$ 
7       $R[j] = A[q + j]$ 
8   $L[n_1 + 1] = \infty$ 
9   $R[n_2 + 1] = \infty$ 
10  $i = 1$ 
11  $j = 1$ 
12 for  $k = p$  to  $r$ 
13     if  $L[i] \leq R[j]$ 
14          $A[k] = L[i]$ 
15          $i = i + 1$ 
16     else  $A[k] = R[j]$ 
17          $j = j + 1$ 
```



Merge-Sort : Analyse

Rekursionstræet



Observation

Samlet arbejde per lag er $O(n)$

Arbejde

$$O(n \cdot \# \text{ lag}) = O(n \cdot \log_2 n)$$

Del-og-kombiner, dADS 1 eksempler:

- **MergeSort**
 - Del op i to lige store dele
 - Rekursiv sortering
 - Kombiner = fletning
- **QuickSort**
 - Opdel efter tilfældigt pivot (**tilfældig opdeling**)
 - Rekursiv sortering
 - Kombiner = ingen (konkatener venstre og højre)
- **QuickSelect**
 - Opdel efter tilfældigt pivot (**tilfældig opdeling**)
 - Rekursiv select
 - Kombiner = ingen

Analyse af Del-og-Kombiner

= analyse af en rekursiv procedure

Essentielt to forskellige måder:

1. Argumenter direkte om **rekursionstræet** (analyser dybde, #knuder på hvert niveau, arbejde i knuderne/niveauerne/træet)
2. Løs en matematisk **rekursionsligning**, f.eks.

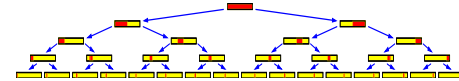
$$T(n) \leq a \quad \text{hvis } n \leq c$$

$$T(n) \leq 2 \cdot T(n/2) + a \cdot n \quad \text{ellers}$$

Bevises f.eks. vha. induktion.

Løsning af rekursionsligninger

- Fold rekursionsligningen ud og argumenter om **rekursionstræet**
- Gæt en løsning og vis den ved induktion efter voksende n



$$T(n) \leq a$$

hvis $n \leq c$

$$T(n) \leq 2 \cdot T(n/2) + a \cdot n$$

ellers

Rekursionsligninger: Faldgrubber

- Ulige opdelinger glemmes (n ulige, så er de rekursive kald typisk $\lfloor n/2 \rfloor$ og $\lceil n/2 \rceil$)

[CLRS, kapitel 4.6.2]

- Analyserer typiske kun for $n = 2^k$
- Brug **aldrig** O-udtryk i rekursionsformlen – brug konstanter (~~$T(n) = O(n) + O(T(n/3))$~~)

$$T(n) \leq c \cdot n + a \cdot T(n/3)$$

Master Theorem

(Simplificering af [CLRS, Theorem 4.1])

Theorem

Hvis a, b, c, d, p er konstanter som opfylder $a, c, p > 0$, $d \geq 1$, og $b > 1$, så har rekursionsligningen

$$T(n) = \begin{cases} c & \text{hvis } n \leq d \\ a \cdot T(n/b) + c \cdot n^p & \text{hvis } n > d \end{cases}$$

følgende løsning

$$\begin{array}{ll} O(n^p) & \text{hvis } a < b^p \\ O(n^p \log n) & \text{hvis } a = b^p \\ O(n^{\log_b a}) & \text{hvis } a > b^p \end{array}$$

Dybde	$i = 0.. \log_b (n/d) - 1$	$\log_b (n/d)$
# delproblemer	a^i	$a^{\log_b (n/d)}$
Størrelse af delproblemer	n/b^i	d
Tid per delproblem	$c \cdot (n/b^i)^p$	c
Tid per lag	$a^i \cdot c \cdot (n/b^i)^p$	$c \cdot a^{\log_b (n/d)}$

$$T(n) \leq c \cdot a^{\log_b (n/d)} + \sum_{i=0}^{\log_b (n/d) - 1} a^i \cdot c \cdot (n/b^i)^p$$

(bunden af rekursionen) (lag $i = 0.. \log_b (n/d) - 1$)

$$\leq c \cdot a^{\log_b n} + c \cdot n^p \cdot \sum_{i=0}^{\log_b n - 1} (a/b^p)^i$$

$$\frac{(a/b^p)^{\log_b n} - 1}{a/b^p - 1} \text{ for } a \neq b^p$$

$$= O \left(n^{\log_b a} + n^p \cdot \begin{cases} 1 & \text{for } a < b^p \\ \log_b n & \text{for } a = b^p \\ (a/b^p)^{\log_b n} & \text{for } a > b^p \end{cases} \right) = O \left(\begin{cases} n^p & \text{for } a < b^p \\ n^p \cdot \log n & \text{for } a = b^p \\ n^{\log_b a} & \text{for } a > b^p \end{cases} \right)$$

$$a^{\log_b n} = n^{\log_b a}$$

$$(b^p)^{\log_b n} = n^p$$

Multiplikation af lange heltal

[CLRS, problem 30.1.c]

Karatsuba 1960

- I og J hver heltal med n bits
- Naive implementation kræver $O(n^2)$ bit operationer
- Lad $I = I_h \cdot 2^{n/2} + I_l$ og $J = J_h \cdot 2^{n/2} + J_l$
- $I \cdot J = I_h \cdot J_h \cdot 2^n + ((I_h - I_l) \cdot (J_l - J_h) + I_l \cdot J_l + I_h \cdot J_h) \cdot 2^{n/2} + I_l \cdot J_l$

$$T(n) \leq 3 \cdot T(n/2) + c \cdot n \quad \text{for } n \geq 2$$

$$T(n) \leq c \quad \text{for } n = 1$$

- $T(n) = O(n^{\log_2 3}) = O(n^{1.58})$

Multiplikation af lange heltal

Del-og-kombiner Karatsuba 1960	$O(n^{\log_2 3})$
Schönhage-Strassen, 1971	$O(n \cdot \log n \cdot \log \log n)$
Fürer, 2007	$O(n \cdot \log n \cdot 2^{O(\log^* n)})$

Matrix Multiplication

$$\begin{pmatrix} c_{11} & c_{12} & \cdots & c_{1p} \\ c_{21} & c_{22} & \cdots & c_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ c_{n1} & c_{n2} & \cdots & c_{np} \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nm} \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} & \cdots & b_{1p} \\ b_{21} & b_{22} & \cdots & b_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ b_{m1} & b_{m2} & \cdots & b_{mp} \end{pmatrix}$$

$$c_{ij} = \sum_{k=1..m} a_{ik} \cdot b_{kj}$$

Matrix Multiplication

$$\begin{pmatrix} c_{11} & c_{12} & \cdots & c_{1p} \\ c_{21} & c_{22} & \cdots & c_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ c_{n1} & c_{n2} & \cdots & c_{np} \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nm} \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} & \cdots & b_{1p} \\ b_{21} & b_{22} & \cdots & b_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ b_{m1} & b_{m2} & \cdots & b_{mp} \end{pmatrix}$$

MATRIX-MULTIPLY(*A*, *B*)

```
1  if A.columns ≠ B.rows
2      error “incompatible dimensions”
3  else let C be a new A.rows × B.columns matrix
4      for i = 1 to A.rows
5          for j = 1 to B.columns
6              cij = 0
7              for k = 1 to A.columns
8                  cij = cij + aik · bkj
9      return C
```

Naive implementation: tid $O(npm)$

(Kvadratisk) Matrix Multiplikation

[CLRS, kapitel 4.2]

$$\begin{pmatrix} I & J \\ K & L \end{pmatrix} = \begin{pmatrix} A & B \\ C & D \end{pmatrix} \begin{pmatrix} E & F \\ G & H \end{pmatrix}$$

$$\begin{aligned} I &= A \cdot E + B \cdot G \\ J &= A \cdot F + B \cdot H \\ K &= C \cdot E + D \cdot G \\ L &= C \cdot F + D \cdot H \end{aligned}$$

- A, B, \dots, K, L er $n/2 \times n/2$ -matricer
- I, J, K, L kan beregnes med **8 rekursive multiplication** og **4 matrix additioner** på $n/2 \times n/2$ -matricer
- $T(n) \leq 8 \cdot T(n/2) + c \cdot n^2$ for $n \geq 2$
 $T(n) \leq c$ for $n = 1$
- $T(n) = O(n^{\log_2 8}) = O(n^3)$

Strassen's Matrix Multiplikation

1969

$$\begin{pmatrix} I & J \\ K & L \end{pmatrix} = \begin{pmatrix} A & B \\ C & D \end{pmatrix} \begin{pmatrix} E & F \\ G & H \end{pmatrix}$$

$$I = S_5 + S_6 + S_4 - S_2$$

$$= (A + D)(E + H) + (B - D)(G + H) + D(G - E) - (A + B)H$$

$$= AE + DE + AH + DH + BG - DG + BH - DH + DG - DE - AH - BH$$

$$= AE + BG.$$

$$J = S_1 + S_2$$

$$= A(F - H) + (A + B)H$$

$$= AF - AH + AH + BH$$

$$= AF + BH.$$

$$K = S_3 + S_4$$

$$= (C + D)E + D(G - E)$$

$$= CE + DE + DG - DE$$

$$= CE + DG.$$

$$L = S_1 - S_7 - S_3 + S_5$$

$$= A(F - H) - (A - C)(E + F) - (C + D)E + (A + D)(E + H)$$

$$= AF - AH - AE + CE - AF + CF - CE - DE + AE + DE + AH + DH$$

$$= CF + DH.$$

$$S_1 = A \cdot (F - H)$$

$$S_2 = (A + B) \cdot H$$

$$S_3 = (C + D) \cdot E$$

$$S_4 = D \cdot (G - E)$$

$$S_5 = (A + D) \cdot (E + H)$$

$$S_6 = (B - D) \cdot (G + H)$$

$$S_7 = (A - C) \cdot (E + F)$$

7 rekursive multiplikationen

Strassen's Matrix Multiplikation

(af to $n \times n$ matricer)

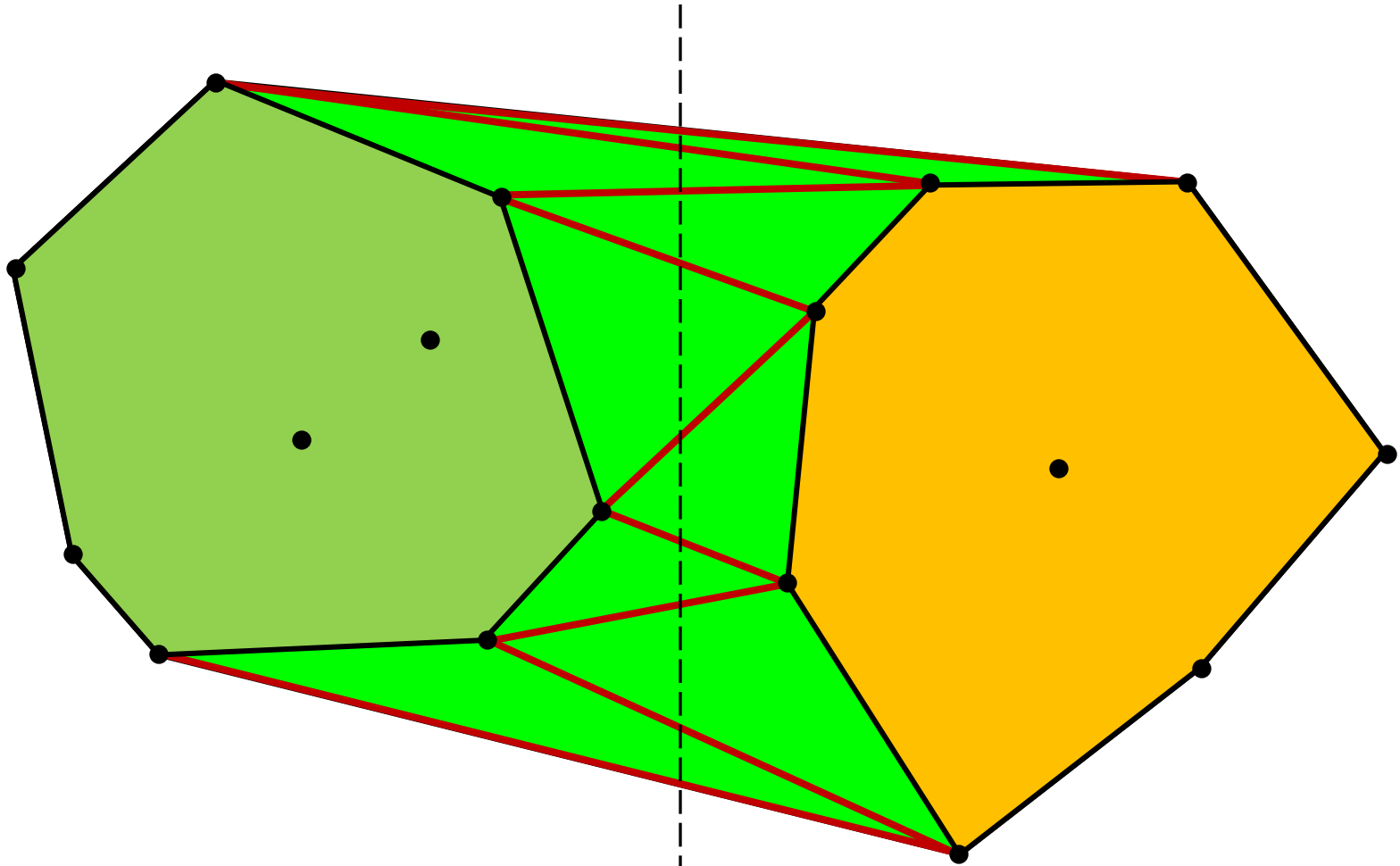
- Bruger **18 matrix additioner** (tid $O(n^2)$) og **7 rekursive matrix multiplikationer**

$$T(n) \leq 7 \cdot T(n/2) + c \cdot n^2 \quad \text{for } n \geq 2$$

$$T(n) \leq c \quad \text{for } n = 1$$

- $T(n) = O(n^{\log_2 7}) = O(n^{2.81})$

Konveks Hylster

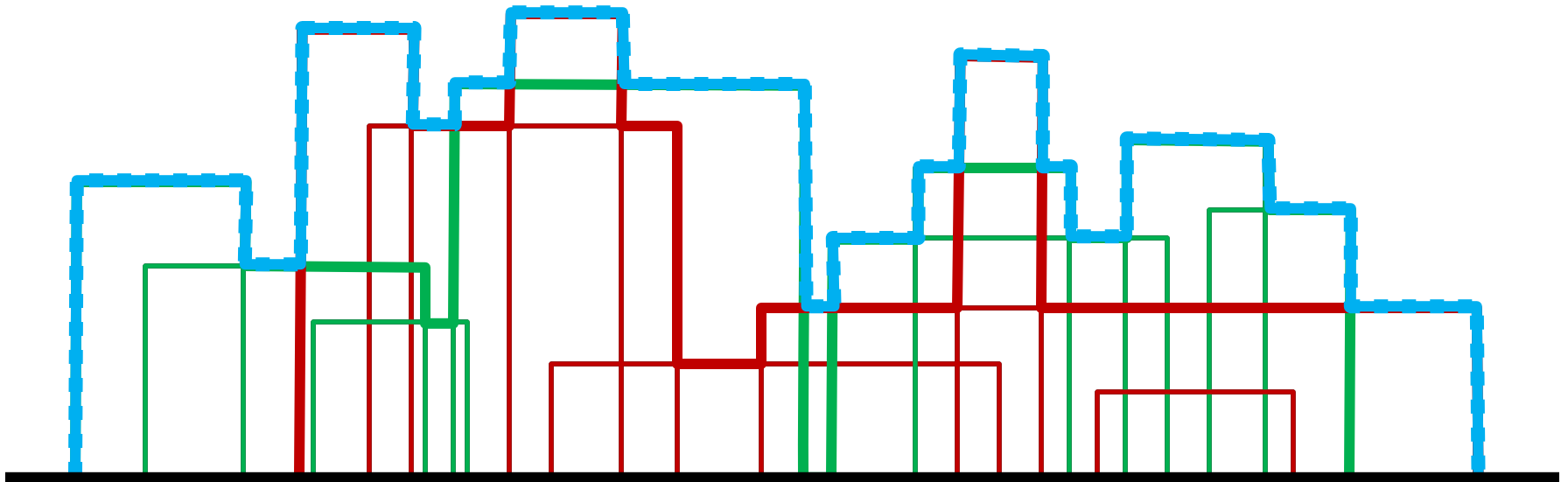


$$T(n) \leq 2 \cdot T(n/2) + c \cdot n \quad \text{for } n \geq 2$$
$$T(n) \leq c \quad \text{for } n = 1$$

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

Silhuet

(afleveringsopgave)



$$T(n) \leq ? \cdot T(n/?) + ? \quad \text{for } n \geq 2$$
$$T(n) \leq c \quad \text{for } n = 1$$