Søgning og Sortering

- Søgning
  - Linæer søgning
  - Binær søgning
- Sortering
  - Indsættelsessortering
  - Flettesortering

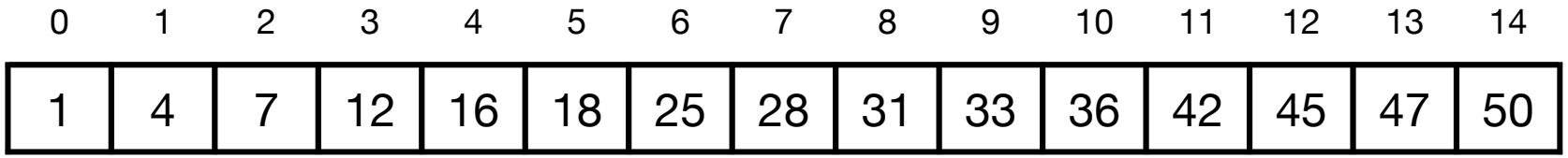
Philip Bille

Søgning og Sortering

- Søgning
  - Linæer søgning
  - Binær søgning
- Sortering
  - Indsættelsessortering
  - Flettesortering

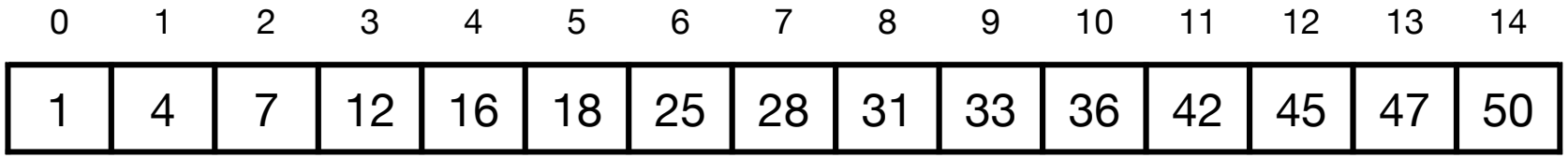
Søgning

- Søgning. Givet en sorteret tabel A og et tal x, afgør om der findes indgang i, så A[i] = x.
- Sorteret tabel. En tabel A[0..n-1] er sorteret hvis A[0] ≤ A[1] ≤ ⋯ ≤ A[n-1] (ikke-faldende rækkefølge).



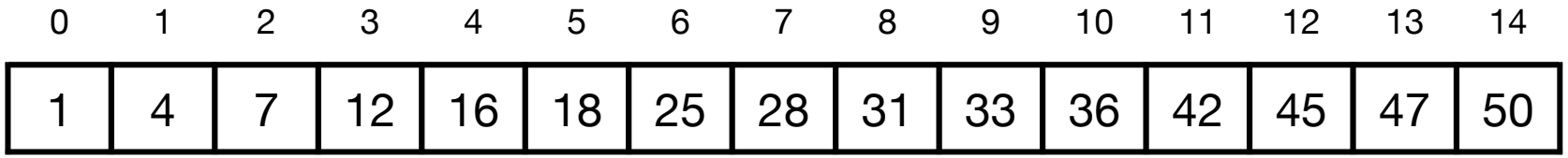
Linæer søgning

- Lineær søgning. Undersøg for alle indgange i om A[i] = x.
- Tid. Θ(n)
- Udfordring. Kan vi udnytte at tabellen er sorteret til at gøre det bedre?



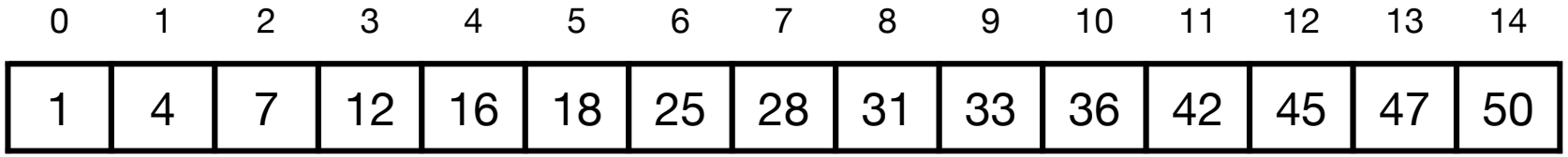
Binær søgning

- Binær søgning (*binary search*). Kig på midterste indgang m i A.
  - hvis A[m] = x returner sand og stop.
  - hvis A[m] < x fortsæt rekursivt på højre halvdel.
  - hvis A[m] > x fortsæt rekursivt på venstre halvdel.
- Stop hvis tabellen har størrelse ≤ 0 og returner falsk.



Binær søgning

```
BinærSøgning(A,i,j,x)
   if j < i return false
   m = ⌊i+j/2⌋
   if A[m] = x return true
   elseif A[m] < x return BinærSøgning(A,m+1,j,x)
   else return BinærSøgning(A,i,m-1,x)     // A[m] > x
```



- Tid. Hvor hurtigt kører den?
- Analyse 1. Analog til analyse af rekursiv toppunktsalgoritme.
  - Et rekursivt kald tager konstant tid.
  - Hvert rekursivt kald halverer tabellen vi kigger på. Vi stopper når tabellen har størrelse ≤ 0.
  - ⇒ Køretiden er Θ(log n)

Binær søgning

- Analyse 2. Lad T(n) være køretiden for binær søgning.
  - Opskriv og udregn rekursionsligningen for T(n).

$$T(n) = \begin{cases} T(n/2) + c & \text{hvis } n > 1 \\ d & \text{hvis } n = 1 \end{cases}$$

$$\begin{aligned}
T(n) &= T\left(\frac{n}{2}\right) + c \\
&= T\left(\frac{n}{4}\right) + c + c \\
&= T\left(\frac{n}{8}\right) + c + c + c \\
&\vdots \\
&= T\left(\frac{n}{2^k}\right) + ck \\
&\vdots \\
&= T\left(\frac{n}{2^{\log_2 n}}\right) + c \log_2 n \\
&= T(1) + c \log_2 n \\
&= d + c \log_2 n \\
&= \Theta(\log n)
\end{aligned}$$

Søgning

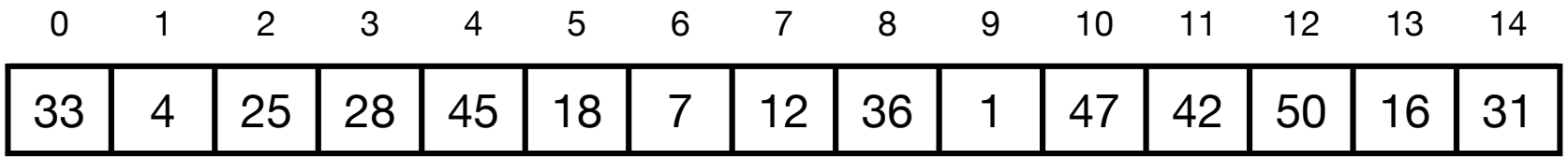
- Vi kan søge i en sorteret tabel i
  - $\Theta(n)$ tid med lineær søgning.
  - $\Theta(\log n)$ tid med binær søgning.

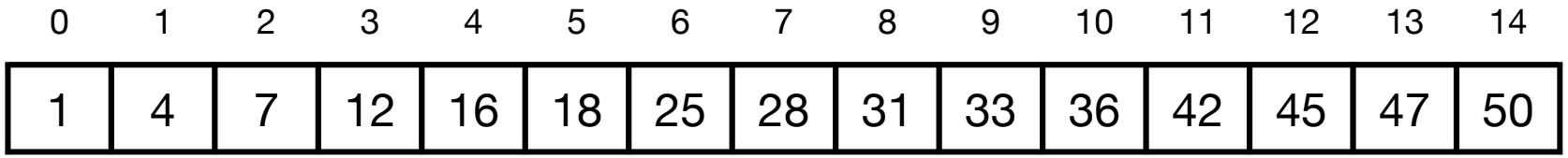
Søgning og Sortering

- Søgning
  - Linæer søgning
  - Binær søgning
- Sortering
  - Indsættelsessortering
  - Flettesortering

Sortering

- Sortering. Givet en tabel A[1..n] returner en tabel B[1..n] med samme værdier som A men i sorteret orden.



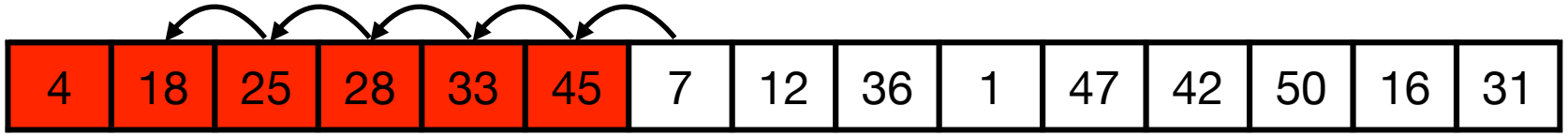


Anvendelser

- Oplagte.
  - Sortere en liste af navne, organisere et MP3 bibliotek, vise Google PageRank resultater, vise Facebook feed i kronologisk rækkefølge.
- Ikke oplagte.
  - Datakompression, computergrafik, bioinformatik, anbefalingssystemer (film på Netflix, bøger på Amazon, reklamer på Google,..).
- Nemme problemer for sorteret data.
  - Binær søgning, find median, identificer duplikater, find tætteste par, find statistiske perifere observationer *(outliers).*

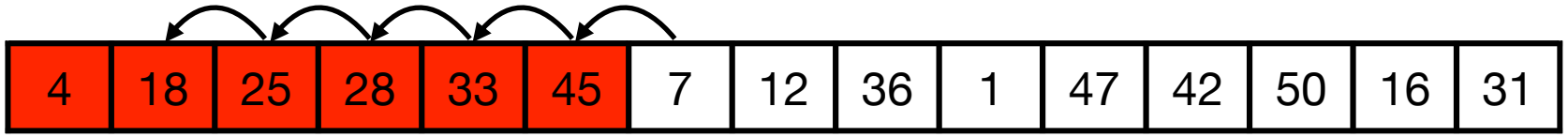
Indsættelsessortering (*insertion-sort*)

- Indsættelsessortering. Start med en usorteret tabel A.
- Kig på indgangene fra venstre til højre i n runder.
- Ved runde i:
  - Deltabel A[0..i-1] er sorteret.
  - Indsæt A[i] i A[0..i-1] så A[0..i] er sorteret.
  - For at finde rette sted til A[i] sammenligner vi med indgangene fra højre til venstre.



Indsættelsessortering

```
INDSÆTTELSESSORTERING(A, n)
   for i = 1 to n-1
      j = i
      while j > 0 and A[j-1] > A[j]
         ombyt A[j] og A[j-1]
         j = j - 1
```



- Tid. Hvad er køretiden T(n)?
  - Hvad er tiden for at indsætte A[i] i sorteret rækkefølge blandt A[0..i-1]?
    - c·i tid for en konstant c.
  - $\Longrightarrow$ samlet tid:

$$T(n) = \sum_{i=1}^{n} ci = c \sum_{i=1}^{n} i = \frac{cn(n+1)}{2} = \Theta(n^2)$$

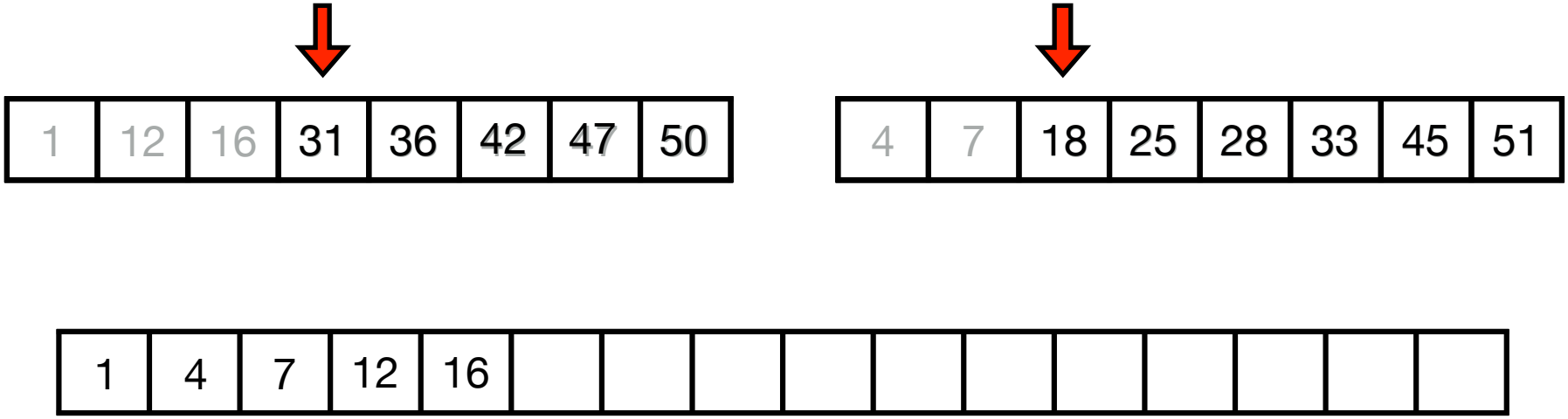
- Udfordring. Kan vi sortere hurtigere?

Flettesortering *(mergesort)*

- Flettesortering. Hurtig sorteringsalgoritme.
- Ide. Rekursiv sortering vha. fletning af sorterede deltabeller.

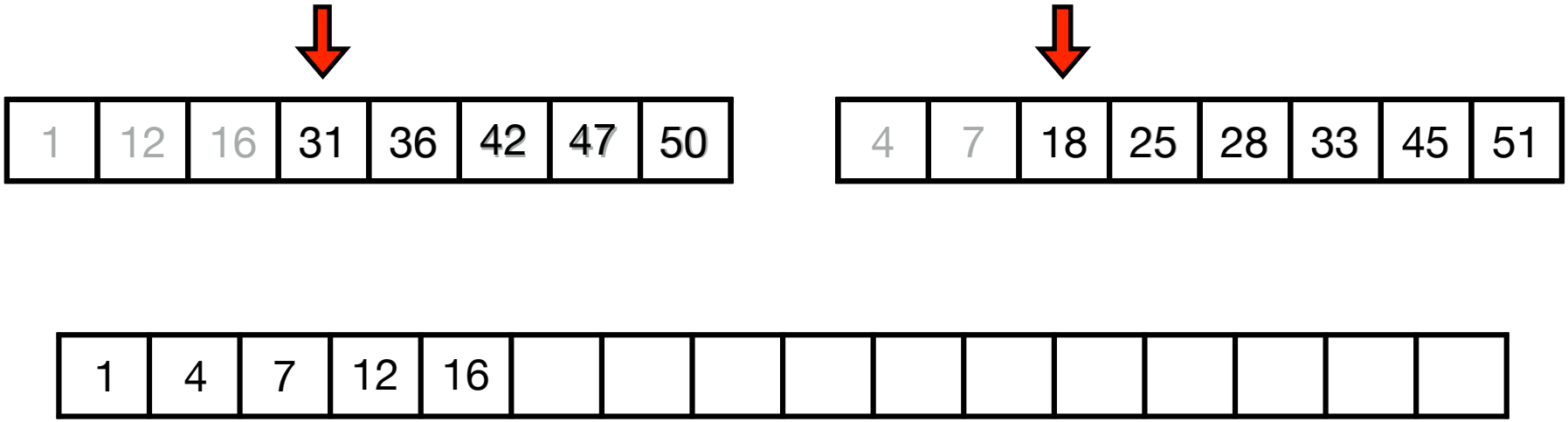
Fletning *(merge)*

- Mål. Kombiner to sorterede tabeller til én sorteret tabel.
- Ide.
  - Gennemløb begge sorterede tabeller i stigende rækkefølge:
    - I hvert skridt indsæt det mindste af de to nuværende elementer fra tabellerne i en ny tabel.
    - Ryk frem i tabellen med det mindste element.
    - Gentag indtil alle elementer er indsat i ny tabel.



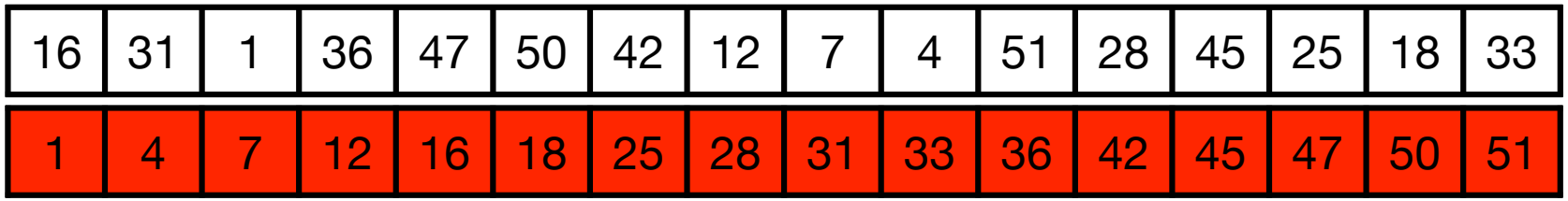
Fletning

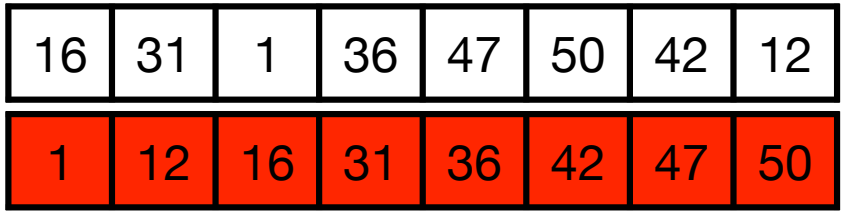
- Tid. Hvor hurtigt kører fletning på to tabeller A_1 og A_2?
  - Hvert skridt i algoritmen tager $\Theta(1)$ tid.
  - I hvert skridt flytter vi en indgang frem i en af tabellerne.
  - $\Rightarrow \Theta(|A_1| + |A_2|)$ tid.

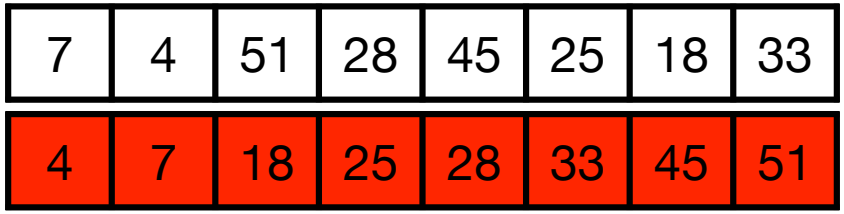


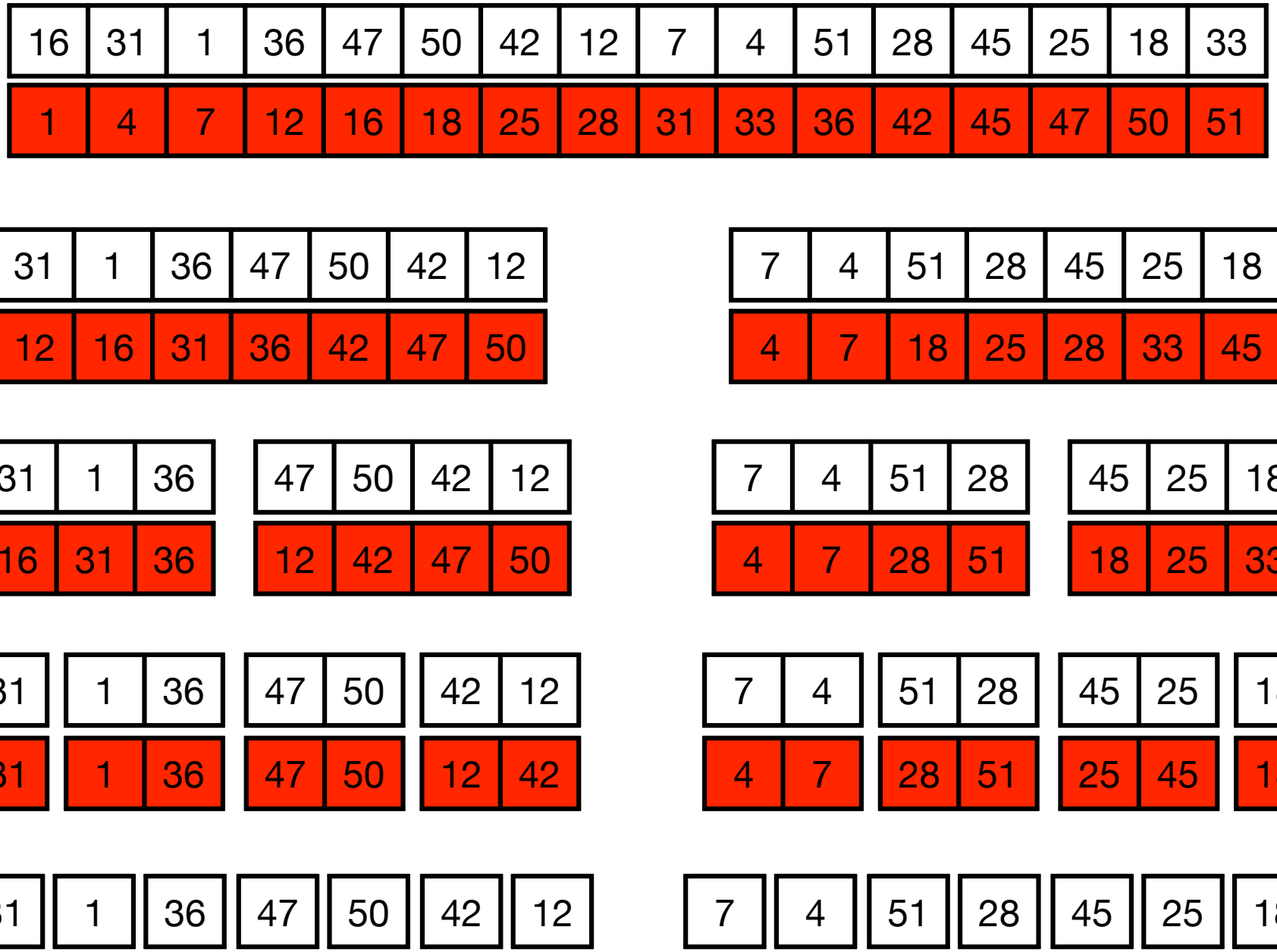
Flettesortering (*mergesort*)

- Flettesortering.
- Hvis |A| ≤ 1, returner A.
- Ellers:
  - Del A i to halvdele.
  - Sorter hver halvdel rekursivt.
  - Flet de to halvdele sammen.



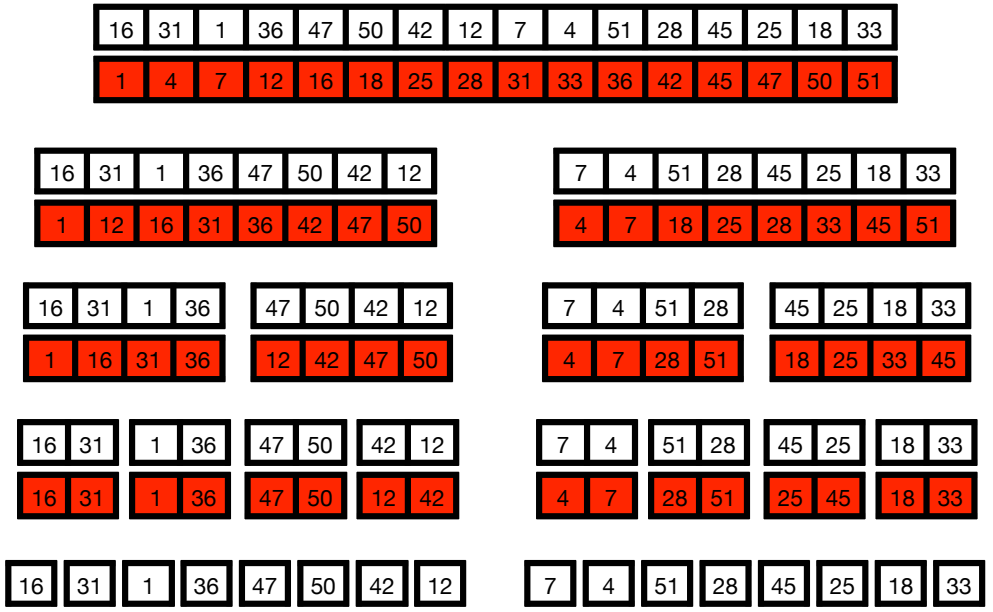





16 31 1 36 47 50 42 12 7 4 51 28 45 25 18 33
1 4 7 12 16 18 25 28 31 33 36 42 45 47 50 51
16 31 1 36 47 50 42 12
1 12 16 31 36 42 47 50
7 4 51 28 45 25 18 33
4 7 18 25 28 33 45 51
16 31 1 36
1 16 31 36
47 50 42 12
12 42 47 50
7 4 51 28
4 7 28 51
45 25 18 33
18 25 33 45
16 31
16 31
1 36
1 36
47 50
47 50
42 12
12 42
7 4
4 7
51 28
28 51
45 25
25 45
18 33
18 33
16 31 1 36 47 50 42 12
7 4 51 28 45 25 18 33

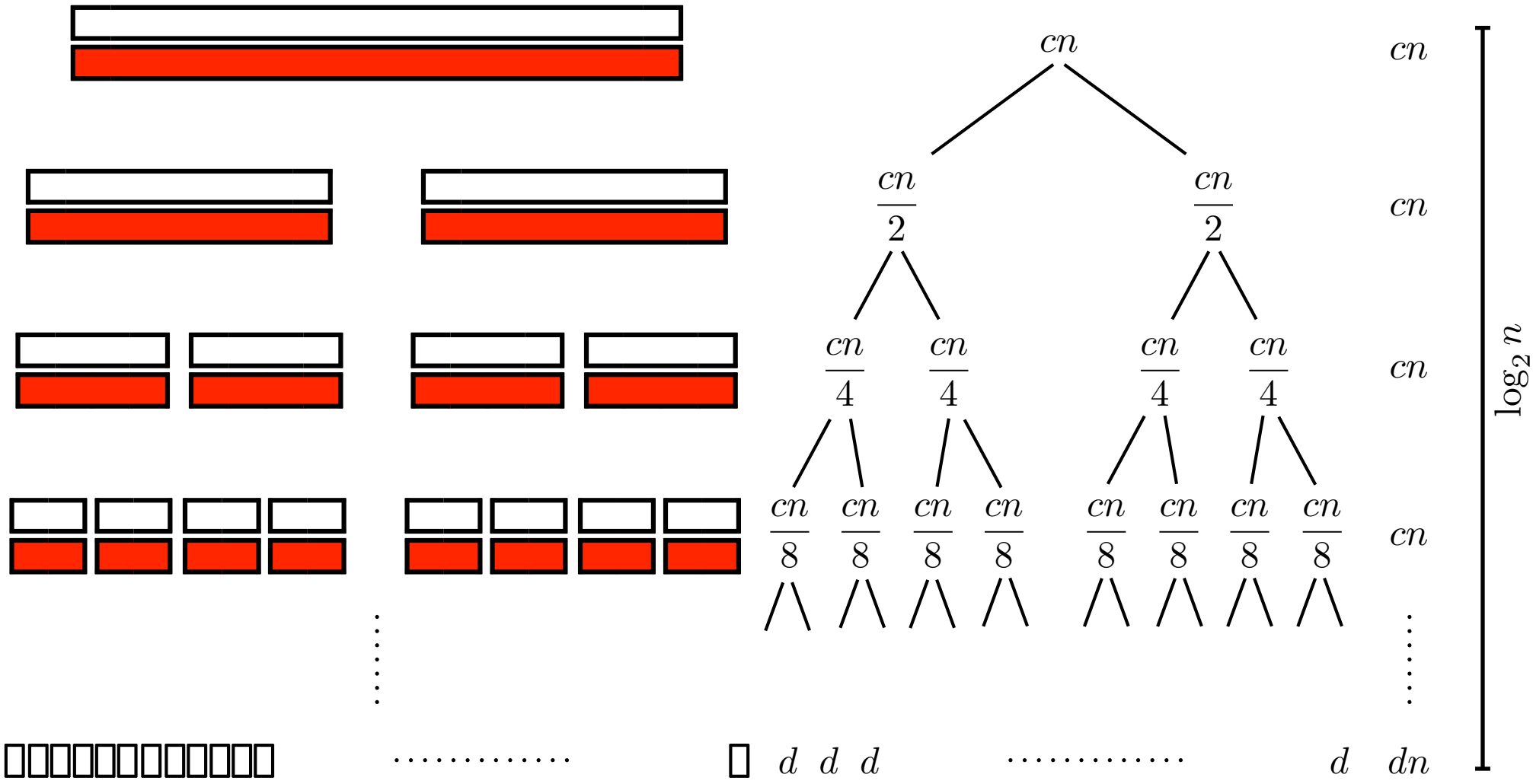
Flettesortering (*mergesort*)

```
FLETTESORTERING(A,i,j)
    if i < j
        m = ⌊i+j/2⌋
        FLETTESORTERING(A,i,m)
        FLETTESORTERING(A,m+1,j)
        FLET(A, i, m, j)
```



- Tid. Lad T(n) være køretiden af flettesortering.
- Hvordan kan vi udregne T(n)?
- Ide. Opstil rekursionstræ og udregn samlet tid for alle rekursive kald.

Flettesortering



$$T(n) = cn \log_2 n + dn = \Theta(n \log_2 n)$$

Sortering

- Vi kan sortere en tabel i
  - $\Theta(n^2)$ tid med indsættelsessortering.
  - $\Theta(n \log n)$ tid med flettesortering.

Del og hersk (*divide-and-conquer*)

- Flettesortering er eksempel på en del-og-hersk algoritme.
- Del-og-hersk. algoritmisk designparadigme.
  - Del. Opdel problemet i et eller flere delproblemer
  - Hersk. Løs delproblemerne rekursivt
  - Kombiner. Sæt løsningerne til delproblemerne sammen til en samlet løsning for problemet.
- Flettesortering.
  - Del. Del A i to halvdele.
  - Hersk. Sorter hver halvdel rekursivt.
  - Kombiner. Flet de to halvdele sammen.

Søgning og Sortering

- Søgning
  - Linæer søgning
  - Binær søgning
- Sortering
  - Indsættelsessortering
  - Flettesortering