

# Ugeseddel: Søgning og sortering

Eva Rotenberg\*

11. februar 2021

## Om denne uge

**Materialer** Introduction to Algorithms, Cormen, Leiserson, Rivest og Stein (CLRS), kap. 2.

## Opgaver

Opgaver markeret med † er implementationsopgaver, og opgaver markeret med \* er særligt udfordrende.

### 1 Håndkøring og egenskaber

Løs følgende opgaver:

- 1.1 [w] Udfør indsættelsessortering på  $A = [31, 41, 59, 26, 41, 58]$ .
- 1.2 [w] Modifier pseudokoden for indsættelsessortering, så den sorterer i svagt faldende (dvs. ikke-stigende) orden, i stedet for svagt stigende (dvs. ikke-faldende) orden.
- 1.3  $A$  er et sorteret array, som indeholder  $n$  forskellige elementer. Hvad er den gennemsnitlige køretid for lineær søgning efter et element i  $A$ ?
- 1.4 Udfør flettesortering på  $A = [3, 41, 52, 26, 38, 57, 9, 49]$
- 1.5 En rekursiv version af indsættelsessortering.
  - Argumenter kort for, at indsættelsessortering kan udtrykkes rekursivt som følger: for at sortere  $A[0, n-1]$  sorterer vi rekursivt  $A[0, n-2]$  og indsætter dernæst element  $A[n-1]$  på den rette plads.
  - Opskriv rekursionsligningen og find en løsning.
- 1.6 En ven foreslår, at man kan forbedre indsættelsessortering ved at benytte binær søgning til hurtigere at finde, hvor næste element skal indsættes.
  - Virker dette? Hvordan påvirkes køretiden?

### 2 Duplikater og tætte naboer

Lad  $A[0..n-1]$  være et array af heltal. Løs følgende opgaver:

- 2.1 Et duplikat i  $A$  er et par af elementer  $i$  og  $j$  (hvor  $i \neq j$ ) så  $A[i] = A[j]$ . Giv en algoritme der afgør, om der et duplikat i  $A$  i  $\Theta(n^2)$  tid.
- 2.2 Giv en algoritme der afgør om der et duplikat i  $A$  i  $\Theta(n \log n)$  tid. Hint: benyt sortering.
- 2.3 Et tætteste par i  $A$  er et par af indekser  $i$  og  $j$  (hvor  $i \neq j$ ) så  $|A[i] - A[j]|$  er minimal blandt alle par af elementer. Giv en algoritme der finder et tætteste par i  $A$  i  $\Theta(n \log n)$  tid.

---

\*baseret på materiale af Bille & Gørtz

### 3 Korrekthed af flettesortering

Vis at flettesortering altid sorterer korrekt. Antag gerne at selve fletteoperationen er korrekt.  
Ledetråd: benyt induktion.

### 4 2-sum og 3-sum

Lad  $A[0..n-1]$  være et array af heltal (positive og negative). Arrayet  $A$  har en 2-sum hvis der findes to indeks  $i$  og  $j$  så  $A[i] + A[j] = 0$ . Tilsvarende har  $A$  en 3-sum hvis der findes tre indeks  $i$ ,  $j$  og  $k$  så  $A[i] + A[j] + A[k] = 0$ . Løs følgende opgaver:

- 4.1 Giv en algoritme der afgør, om  $A$  har en 2-sum, i  $\Theta(n^2)$  tid.
- 4.2 Giv en algoritme der afgør, om  $A$  har en 2-sum, i  $\Theta(n \log n)$  tid. Hint: benyt binær søgning.
- 4.3 Giv en algoritme der afgør om  $A$  har en 3-sum i  $\Theta(n^3)$  tid.
- 4.4 Giv en algoritme der afgør om  $A$  har en 3-sum i  $\Theta(n^2 \log n)$  tid. Hint: benyt binær søgning.
- 4.5 **[\*\*]** Giv en algoritme der afgør, om  $A$  har en 3-sum, i  $\Theta(n^2)$  tid.

### 5 Udvælgelse, partitionering og kviksortering

Lad  $A[0..n-1]$  være et array af forskellige heltal. Tallet med rang  $k$  i  $A$  er det tal, der fremkommer på position  $k$ , såfremt man sorterer  $A$ . Medianen af  $A$  er tallet i  $A$  med rang  $\lceil (n-1)/2 \rceil$ . Løs følgende opgaver:

- 5.1 Giv en algoritme der givet et  $k$  finder tallet med rang  $k$  i  $A$  i  $\Theta(n \log n)$  tid.

En partitionering af  $A$  er en opdeling af  $A$  i to arrays  $A_{\text{lav}}$  og  $A_{\text{høj}}$  således at  $A_{\text{lav}}$  indeholder alle tal fra  $A$  der er mindre end eller lig med medianen af  $A$  og  $A_{\text{høj}}$  indeholder alle tal fra  $A$  der er større end medianen af  $A$ . Antag i det følgende at du har en lineærtidsalgoritme til at finde medianen af et array.

- 5.2 Giv en algoritme til at beregne en partitionering af  $A$  i  $\Theta(n)$  tid.
- 5.3 **[\*]** Giv en algoritme til at sortere  $A$  i  $\Theta(n \log n)$  tid vha. rekursiv partitionering.
- 5.4 **[\*\*]** Giv en algoritme, der givet et  $k$ , finder tallet med rang  $k$  i  $A$  i  $\Theta(n)$  tid.

## Implementationsopgaver

### 6 Implementation af binær søgning

† Implementer algoritmen til binær søgning.

### 7 Implementation af flettesortering

Løs følgende opgaver:

- 7.1 † Implementér algoritmen til fletning.
- 7.2 † Implementér flettesortering.