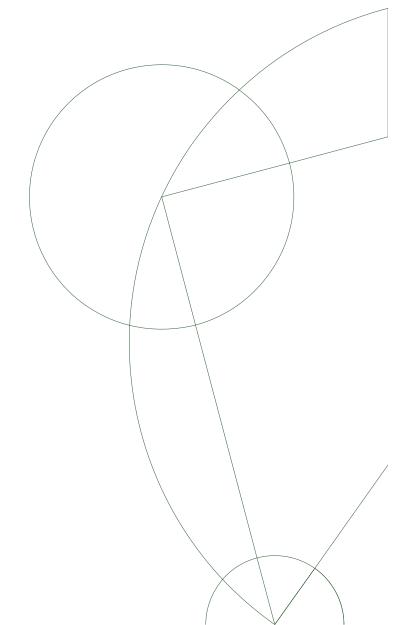


Diskret Matematik og Algoritmer Aflevering 2i

Adam Frederik Ingwersen Linnemann, GQR701 Hold $4\,$

Datalogisk Institut Københavns Universitet

September 19, 2016



Del 1

Antallet af inversioner kan bestemmes ved at tælle det samlede antal 'ikkesorterede' placeringer af elementerne i A. For arrray bestående af følgende n=6 elementer; [2,1,8,4,3,6] er der sammenlagt $\underline{5}$ inversioner.

Del 2

En generel formel for at finde det maksimale antal inversioner, k, i et array bestående af n vilkårlige elementer kan bestemmes ved:

$$k = \frac{n(n-1)}{2}$$

Dette kan bl.a. skrives på følgende måde ved iterativ pseudokode:

Algorithm 1 Maksimale antal inversioner givet n

```
1: function MAXINV(n)
```

2: v = 0

3: for i = 1 to n

4: v = (i-1) + v

5: return v

Del 3

Denne delopgave kan løses på adskillige måder - både rekursivt og iterativt. Her er valgt en iterativ tilgang, der bygger på lineær søgning. Algoritmen opererer på følgende måde:

- 1. Positionér starten af den første løkke (i=0) ved A's 0'te element
- 2. Positionér herefter starten af den anden løkke foran i (j=i+1)
- 3. Lad j
 iterere fra j=1 til n over nedenstående betingelse:
 - (a) Hvis værdien af det i'te element er strengt større end det j'te, da:
 - (b) Denotér variablen 'inv' med sin hidtige værdi plus 1
- 4. Afslut ved at returnere den akkumulerede værdi for 'inv'

Algoritmen er beskrevet i et mere koncist format i pseudokoden nedenfor.

Algorithm 2 Tæl antal inversioner i Afunction CountInv(A, n)2: inv = 0
for i = 0 to n - 2 c_1
 $c_2 \cdot n - 1$ 4: for j = i+1 to n - 1
if A[i] > A[j]
6: inv = inv + 1
return inv $c_3 \cdot \sum_{j=1}^n (t_j - 1)$
 $c_5 \cdot \sum_{j=1}^n (t_j - 1)$

En alternativ løsning til den 'fladpandede' lineære søgning ville være, at anvende merge-sort med et if-statement og en tælle-variabel i den afsluttende 'merge'-del.

Del 4

For at analysere algoritmens køretid, anvendes tilgangen beskrevet i CLRS 2.2.

$$k = (n-1) \cdot (n-1) \cdot n - 2/2$$