

Skriftlig Eksamen
Algoritmer og Datastrukturer (DM02)

Institut for Matematik og Datalogi
Syddansk Universitet, Odense

Tirsdag den 24. januar 2006, kl. 9–13

Løsningsforslag

Opgave 1

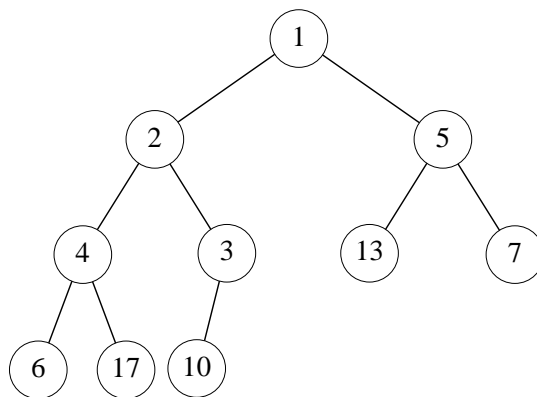
Spørgsmål a:

$$h(18, 0) = h_1(18) \bmod 11 = 7$$

$$h(18, 1) = (h_1(18) + h_2(18)) \bmod 11 = (7 + (1 + 8)) \bmod 11 = 5$$

Plads 7 er optaget, så det nye element havner på plads 5. □

Spørgsmål b:



□

Spørgsmål c: Vi kan bruge Master-sætningen, tilfælde 3:

$$\log_3 3 = 1$$

$$n^2 \in \Omega(n^{1+\frac{1}{2}}) \text{ og } 3 \left(\frac{n}{3}\right)^2 = \frac{1}{3}n^2.$$

Dvs. $T(n) \in \Theta(n^2)$. □

Opgave 2

For enhver følge S af tal definerer vi

$$f(i, j) = \begin{cases} 1, & \text{hvis } S[i] > S[j] \\ 0, & \text{ellers.} \end{cases}$$

Antallet af inversioner i en følge S af n tal er

$$\sum_{i=1}^{n-1} \sum_{j=i+1}^n f(i, j).$$

Spørgsmål a: En følge, som er sorteret i faldende orden, har

$$\sum_{i=1}^{n-1} i = \frac{n(n-1)}{2} = \frac{1}{2}(n^2 - n)$$

inversioner. Dette er selvfølgelig det maksimale antal. □

Spørgsmål b: Indfør en variabel, inv , som sættes lig 0 i starten af mergesort-algoritmen. Allersidst i merge-algoritmen (som står på s. 29 i bogen) tilføjes en linie:

$$inv \leftarrow inv + n_1 - i + 1$$

Dvs. linie 13–18 bliver følgende:

if $L[i] \leq R[j]$

$$A[k] \leftarrow L[i]$$

$$i \leftarrow i + 1$$

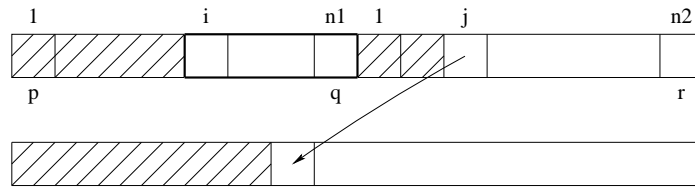
else

$$A[k] \leftarrow R[j]$$

$$j \leftarrow j + 1$$

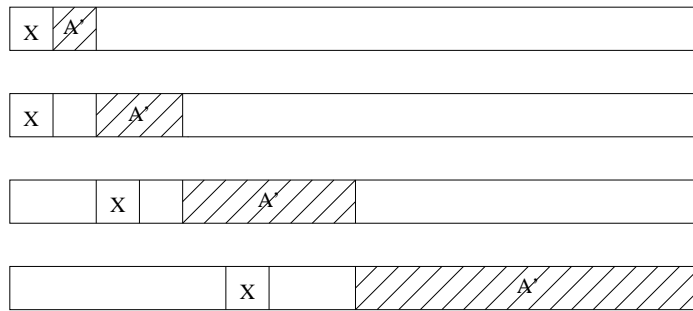
$$inv \leftarrow inv + n_1 - i + 1$$

Hvis $L[i] > R[j]$ betyder det nemlig, at tallene $L[i..n_1]$ er større end $R[j]$, og disse tal stod før $R[j]$ i den oprindelige følge:



Ingen inversioner tælles mere end en gang:

Hvert element x “oplever” ca. $\log_2 n$ gange at være i et del-array, som bliver flettet med et andet del-array A' . Hver gang gælder, at ingen af tallene i A' har optrådt i nogen af de delarrays, som x tidligere er blevet flettet med:



□

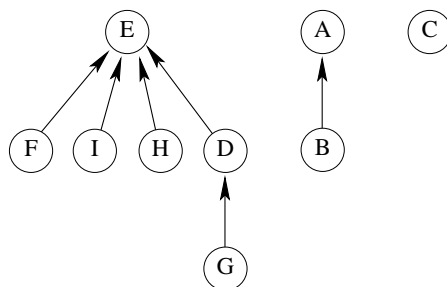
Spørgsmål c:

1. Insertion Sort har køretid $\Theta(n)$: $S[j]$ flyttes $\sum_{i=1}^{j-1} f(i, j)$ pladser; dvs. antallet af swaps er lig antallet af inversioner.
2. Radix Sort. Brug f.eks. radix 2^k , hvor $2^{k-1} \leq n \leq 2^k$. Så kan man nøjes med to gennemløb (passes): i første gennemløb sammenlignes de k mindst betydende cifre, og i sidste gennemløb sammenlignes de mest betydende cifre. D.v.s. køretiden bliver $\Theta(n)$.
3. Heapsort og Mergesort har køretid $O(n \log n)$.

□

Opgave 3

Spørgsmål a: DH er den næste kant, der bliver inkluderet.



□

Spørgsmål b: AB, BD, DG, DH, HI, HE, EF, FC

□

Spørgsmål c: Algoritmen består af tre trin:

1. Først undersøges, om kanterne i $\{e_1, \dots, e_k\}$ danner en kreds:
 - Konstruer delgraf G' af G , som netop indeholder kanterne $\{e_1, \dots, e_k\}$: Kanterne tilføjes en for en til adjacenslister.
 - Udfør f.eks. BFS på G' . Hvis nogen knude opdages mere end en gang, indeholder G' en kreds.

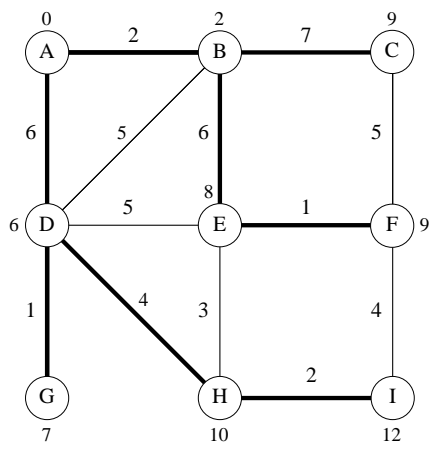
Begge dele kan udføres i $O(n+k)$ tid. Hvis G' ikke indeholder en kreds, går man videre til trin 2.

2. Derefter udføres eksempelvis Kruskals algoritme på G , og den samlede vægt af det resulterende udspændende træ huskes.
3. Endelig udføres Kruskals algoritme med $\{e_1, \dots, e_k\}$ som start-kanter. Hvis det resulterende træ har samme samlet vægt som træet fra trin 2, er det et letteste udspændende træ (MST).

Hvis ikke, findes der ikke et MST, som indeholder $\{e_1, \dots, e_k\}$. Hvis $\{e_1, \dots, e_k\}$ er indeholdt i et MST, vil Kruskals algoritme nemlig kun tilføje sikre kanter til træet (Thm 23.1), og dermed vil det resulterende træ være et MST.



Spørgsmål d:



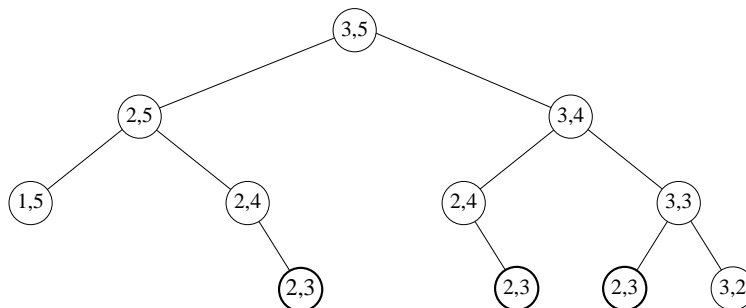
Opgave 4

Spørgsmål a:

		j					
		0	1	2	3	4	5
i	0	S	F	F	F	F	F
	1	S	S	S	F	F	F
	2	F	F	S	S	S	F
	3	F	F	F	S	S	S

□

Spørgsmål b: FLET("akt", "beate", "abekatte", i , j) kalder FLET("akt", "beate", "abekatte", $i - 1$, j) og FLET("akt", "beate", "abekatte", i , $j - 1$), hvis $i, j \geq 1$. Dvs. der foretages **tre** kald til FLET("akt", "beate", "abekatte", 2, 3):



Kan også beregnes sådan:

Lad $T(i, j)$ være antallet af kald til FLET("akt", "beate", "abekatte", i , j). Hvis $i < 3$ og $j < 5$, kaldes FLET("akt", "beate", "abekatte", i , j) af FLET("akt", "beate", "abekatte", $i + 1$, j) og FLET("akt", "beate", "abekatte", i , $j + 1$). Dvs.

$$T(i, j) = \begin{cases} 1, & \text{hvis } i = 3 \vee j = 5 \\ T(i + 1, j) + T(i, j + 1), & \text{ellers.} \end{cases}$$

T		j					
		0	1	2	3	4	5
i	0	56	35	20	10	4	1
	1	21	15	10	6	3	1
	2	6	5	4	3	2	1
	3	1	1	1	1	1	1

□

Spørgsmål c:

FLET(x, y, z, n, m)

Tabel[0,0] \leftarrow Sand

For $i \leftarrow 1$ til n

 Tabel[$i, 0$] \leftarrow FX($i, 0$)

For $j \leftarrow 1$ til m

 Tabel[0, j] \leftarrow FY(0, j)

For $i \leftarrow 1$ to n

 For $j \leftarrow 1$ to m

 Tabel[i, j] \leftarrow (FX(i, j) \vee FY(i, j))

Returner Tabel[n, m]

FX(i, j)

 returner (Tabel[$i - 1, j$] \wedge $z[i + j] = x[i]$)

FY(i, j)

 returner (Tabel[$i, j - 1$] \wedge $z[i + j] = y[j]$)

Hvert felt i tabellen udfyldes i tid $O(1)$.

Der er $O(mn)$ felter, så den samlede køretid bliver $O(mn)$.

□