Institut for Matematik og Datalogi
Syddansk Universitet, Odense

13. februar 2025
RF

Opgaver Uge 9

DM507/DM578/DS814/SE4-DMAD

Opsummering fra forelæsninger: man kan med fordel opdele udviklingen af en algoritme i fire niveauer:

1. Idé.
2. Detaljeret tegning, der fanger det væsentlige i algoritmen.
3. Pseudo-kode.
4. Færdig kode (i f.eks. Python eller Java).

Princippet i en algoritme kan normalt beskrives godt og præcist allerede på niveau 2, og korrekthedsanalyse og asymptotisk analyse med grovsortering efter voksehastighed kan finde sted der. Niveau 3 tilføjer flere detaljer (præcist valg af variable, deres initialisering, deres opdatering under løkker, etc.), som skal være på plads for at sikre korrekt koden i sidste ende. Det kan ofte tage overraskende lang tid, men det kan betale sig at være omhyggelig og gennemtænke korrektheden af disse tilføjede detaljer, før man går videre til næste niveau. Hvis niveau 3 er lavet grundigt, bliver niveau 4 ofte relativt nemt.

Til forelæsningerne udtrykker vi os primært på niveau 1 og 2. Lærebogen gør også dette, men tilføjer derudover detaljeret pseudo-kode, dvs. niveau 3. I programmeringsopgaverne til øvelsestimerne og (for DM507) i projektet arbejder vi med niveau 4 – ofte baseret på niveau 3 fra bogen, andre gange baseret på niveau 1–3, som man selv skal lave først.

Til øvelsestimerne er det i opgaver, hvor man skal udvikle, beskrive og/eller analysere algoritmer, *nok at gøre dette på niveau 2*, medmindre andet er angivet (dvs. medmindre man direkte bliver bedt om pseudo-kode eller Python/Java-implementation).

A: Løses i løbet af øvelsestimerne i uge 9

1. Eksamen juni 2008, opgave 2 (tidligere eksamensopgaver kan findes øverst på kursets hjemmeside).

2. Hvilke af følgende udsagn er sande?

   (a) $2n + 5 = \Theta(n)$
   (b) $n = o(n^2)$
   (c) $n = O(n^2)$
   (d) $n = \Theta(n^2)$
   (e) $n = \Omega(n^2)$
   (f) $n = \omega(n^2)$
   (g) $\sqrt{n}\log(n) = O(\sqrt{n})$
   (h) $\sqrt{n}\log(n) = \omega(\sqrt{n})$
   (i) $\sqrt{n} + \log(n) = O(n)$
   (j) $(\log(n))^3 = O(n \log n)$

3. Eksamen juni 2011, opgave 2. [Hint: For et sandt svar, argumenter ud fra definitionerne på O og Ω. For et falsk svar, find modeksempler.] Opgaven bruger notationen $f(n) \in O(g(n))$, hvilket betyder det samme som $f(n) = O(g(n))$. Se evt. diskussion i Cormen et al., 4. udgave, side 55 [Cormen et al., 3. udgave: side 45].

4. Angiv for hver af følgende to algoritmer deres asymptotiske køretid i Θ-notation som funktion af n.

```
ALGORITME1(n)
    s = 0
    for i = 1 to ⌊n/2⌋
        for j = i to n − i
            s = s + 1
    return s
```

```
ALGORITME2(n)
    s = 0
    for i = 1 to n
        for j = 1 to n
            for k = 1 to n
                s = s + 1
    return s
```

5. Implementer Mergesort i Python eller Java ud fra bogens pseudokode (Cormen et al., 4. udgave, side 36 og 39 [Cormen et al., 3. udgave: side 31 og 34]). Lad input være en liste (Python) eller et array (Java) af heltal.

For Cormen et al., 3. udgave: I denne udgave optræder værdien ∞ i pseudokoden på side 31. Brug her et stort tal, og sørg for at kun heltal under denne værdi optræder i input.[1] Alternativt (og bedre, da man undgår at skulle tage hensyn til værdien valgt som ∞), brug til MERGE pseudokoden fra Cormen et al., 4. udgave, som kan findes i slides om sortering. Denne pseudokode anvender ikke ∞.

Test at din kode fungerer ved at generere små arrays/lister med forskelligt indhold (f.eks. stigende, faldende, tilfældige), sortere dem og checke resultatet.

B: Løses hjemme inden øvelsestimerne i uge 10

1. Eksamen januar 2007, opgave 3. Opgaven bruger notationen $f(n) \in O(g(n))$, hvilket betyder det samme som $f(n) = O(g(n))$.

2. Angiv for hver af følgende to algoritmer deres asymptotiske køretid i Θ-notation som funktion af n.

```
ALGORITME3(n)
    s = 0
    for i = 1 to n
        for j = 1 to n
            if i == j
                for k = 1 to n
                    s = s + 1
    return s
```

```
ALGORITME4(n)
    s = 0
    for i = 1 to n
        for j = 1 to n
            if i != j
                for k = 1 to n
                    s = s + 1
    return s
```

3. Fortsæt opgaven ovenfor med implementation af Mergesort på denne måde:

   Tilføj tidtagning af din kode. Du skal kun tage tid på selve sorteringen, ikke den del af programmet som genererer array'ets/listens indhold. Kør derefter din kode med input, som er tilfældige heltal, samt input

[1] Faktisk har både Python og Java en indbygget værdi for ∞ (i Python `float("inf")`, i Java `Double.POSITIVE_INFINITY`). Disse er teknisk set kommatal, men fungerer efter hensigten, hvis de sammenlignes med heltal. I Java kan et kommatal og et heltal dog ikke optræde i samme array, så pseudokoden fra Cormen et al., 3. udgave kan ikke implementeres vha. denne værdi. I Python kan et kommatal og et heltal godt optræde i samme liste, så pseudokoden fra Cormen et al., 3. udgave kan godt implementeres vha. denne værdi.

som er sorteret og omvendt sorteret. Gør dette for mindst 5 forskellige værdier af n (antal elementer at sortere), vælg værdier som får programmet til at bruge fra ca. 100 til ca. 5000 millisekunder. Gentag hver enkelt kørsel tre gange og find gennemsnittet af antal millisekunder brugt ved de tre kørsler. Divider de fremkomne tal med $n \log_2 n$, og check derved, hvor godt analysen passer med praksis – de resulterende tal burde ifølge analysen være konstante.

[Hint: Se seddel med opgaver til uge 8 for informationer om bogens indeksering i forhold til Pythons/Javas, samt om metoder i Python og Java til tidstagning og til generering af tilfældige heltal. For at beregne logaritmer: Man kan i Python bruge metoden `math.log(x,2)` til at beregne $\log_2(x)$ (husk at importere `math` modulet). Man kan i Java bruge metoden `java.lang.Math.log(x)`, til at beregne $\log_e(x)$, dvs. logaritmen med grundtal e. For at beregne $\log_2(x)$ kan man så bruge at $\log_2(x) = \log_e(x)/\log_e(2)$ (jvf. Cormen et al., 4. udgave, formel 3.19 side 66 [Cormen et al., 3. udgave: formel 3.15 side 56]).]

4. ($*$) Cormen et al., 4. udgave, opgave 2-4 (side 47) [Cormen et al., 3. udgave: opgave 2-4 (side 41)], spørgsmål **d**.