Institut for Matematik og Datalogi
Syddansk Universitet, Odense

7. februar 2010
RF

DM507 – Ugeseddel 2

Forelæsninger i uge 5

Mandag 1/2

- Introduktion af kurset (slides, Cormen et al. kapitel 1).
- Kort repetition af induktion.

Onsdag 3/2

- Algoritmeanalyse. Eksempel på algoritmeanalyse: InsertionSort (Cormen et al. afsnit 2.1–2).
- Algoritmeanalyse vs. målt køretid i praksis for simple testprogrammer (slides).

Bemærk: Det kan være en fordel at orientere sig om indholdet i Cormen et al. afsnit 3.2 og appendix A–D. Hvis der i kurset bruges matematiske formler, definitioner og metoder, man har glemt, kan de sandsynligvis findes opsummeret der.

Forelæsninger i uge 6

Mandag 8/2 (forventet indhold)

- Asymptotisk notation (Cormen et al. afsnit 3.1).
- Divide-and-conquer algoritmer. Eksempel: MergeSort (Cormen et al. afsnit 2.3).
- Start på rekursionsligningen (Cormen et al. afsnit 4.3–5).

Øvelsesopgaver i uge 6

Onsdag 10/2 (S7) og torsdag 11/2 (M1)

- Cormen et al. øvelse 2.1-3 (side 22).
- Cormen et al. øvelse 2.2-2 (side 29).
- Cormen et al. øvelse 2.2-3 (side 29). Svar også for best-case køretid.
- Cormen et al. øvelse 2.3-1 (side 37).
- Cormen et al. øvelse 2.3-7 (side 39). Hint: brug sortering.
- Cormen et al. opgave 2-1 (side 39). Til spørgsmål d: tanken her er at bruge køretid i praksis til at fintune det endelige valg af k.

Torsdag 11/2 (S7) og fredag 12/2 (M1)

- Eksamen januar 2004, opg. 1. Denne er essentielt den samme som Cormen et al. opgave 2.3 (side 41), som evt. kan læses for øget forståelse.
- Cormen et al. øvelse 3.1-1 (side 52).
- Cormen et al. øvelse 3.1-2 (side 52).
- Cormen et al. øvelse 3.2-3 (side 60). Hint: brug Stirlings formel (formel (3.18)) for at vise formel (3.19).
- Cormen et al. øvelse 2.3-4 (side 39).
- Cormen et al. øvelse 2.3-5 (side 39). Lav *både* en iterativ (dvs. med en **while**-løkke) og en rekursiv version. For analyse af køretid for den rekursive version, opskriv en rekursionsligning og løs den (tegn rekursionstræet). For den iterative version, argumenter f.eks. at dens køretid er givet ved samme rekursionsligning. Implementer begge dine algoritmer i Java, og test dem for korrekthed. Afprøv hvilken der er hurtigst ved at allokere et array af heltal af længde n, indsætte tallene fra 1 til n i sorteret orden, og derefter tage tid for n søgninger, f.eks. efter tallene fra 1 til n. Tag tid via metoden `System.currentTimeMillis()`. Mål den totale tid for alle søgninger, uden tiden for at fylde arrayet op. Sæt n til en passende størrelse, som giver en køretid på mindst et sekund. Tag gennemsnit af 5 kørsler når du sammenligner den målte køretid for de to implementationer.