Enkel analyse med O-notasjon

Harald Askestad
haraldas@ifi.uio.no

1 Enkel analyse med O-notasjon – en motivasjon

Bare de *viktige* elementene teller i O-notasjonen.

Vi skriver:

> $T(N) = O(N)$ som betyr at
> $T(N) \leq f(N) * c$

I formlene over er T(N) en funksjon for hvor lang tid programmet tar, og det er avhengig av antall elementer N. f(n)*c er en eksakt funksjon for hvor lang tid programmet bruker i værste tilfelle. Når vi setter en *O* rundt denne funksjonen stryker vi alle konstanter og alle ledd med lavere orden. Da kan vi endre $\leq$ til $=$.

Hva er vitsen med dette?

> Vi fant ut at Hanoi krever $2^N - 1$ flytt. Ett flytt krever et konstant antall operasjoner (kall det c). Tiden det tar å kjøre programmet med N ringer blir da:
>
> $$T(N) = c(2^N - 1) = O(2^N)$$
>
> Dette er et **STORT** tall! Det tar dobblet så lang tid å kjøre programmet med én ring mer. Jeg prøver å løse problemet med å kjøpe en ny og raskere maskin:
>
> > Min gamle PC klarer 30 ringer på en "akseptabel tid".
> > Jeg kjøper en ny maskin som er 4 ganger så rask (ca. 20.000,- kr): Den klarer 32 ringer på samme tid!
> > Da kjøper jeg en supermaskin som er 128 ganger så rask(ca. 1.000.000,- kr)! Den klarer 37 ringer på samme tid!
>
> Med andre ord; jeg kaster masse penger ut av vinduet fordi programmet er treigt.

Dette eksemplet skal vise at 2^N leddet betyr mer enn alle de andre leddene. Om vi legger til aldri så store konstanter, så vil 2^N bety mest for tiden når N vokser.

2 Løsningsforslag

Oppgave 2.7 a) (fra boka)

(1) $T(n) = O(n)$

(2) $T(n) = O(n^2)$

(3) $T(n) = O(n^3)$

(4) $T(n) \approx 1+2+3+...+n = \frac{n(n+1)}{2} = O(n^2)$

(5) $T(n) = O(n^5)$

(6) $T(n) = O(n^4)$

Formelen $1+2+3+...+n = \frac{n(n+1)}{2}$ i punkt (4) kan forklares med følgende resonnement:

```
  S =  1  +  2  +  3   +...+  n
  S =  n  + n-1 + n-2  +...+  1
 2S = n+1 + n+1 + n+1  +...+  n+1
```

Hvor mange $n+1$ har vi? Jo, n stykker. Dermed

$$2S = n(n+1) \Leftrightarrow S = \frac{n(n+1)}{2}$$

Punkt (6) fortjener en forklaring. $O(n^5)$ er riktig, men $O(n^4)$ er et mer presist svar. De to første forløkkene vil sørge for at if-testen utføres $O(n^3)$ ganger. Men fordi det står en forløkke inni kan det hende at det antall ganger `sum++` utføres er av større orden. Det avhenger av hvor mange ganger if-testen er sann.

> Når i er 1 går j til 1 og if-testen er sann 1 gang.
> Når i er 2 går j til 4 og if-testen er sann 2 ganger.
> Når i er 3 går j til 9 og if-testen er sann 3 ganger.
> ...
> Når i er n går j til n*n og if-testen er sann n ganger.

D.v.s. at den indre for-løkka utføres $1+2+3+...+n = O(n^2)$ ganger og den gjør i værste tilfelle $O(n^2)$ repetisjoner. `sum++` utføres $O(n^2 n^2) = O(n^4)$ ganger.

Ukeoppgave 5

a) $T(n) = O(4^n n^2)$

b) $T(n) = O(n^2)$

c) $T(n) = O(\log_2(n))$