SI-möte #6, Programkonstruktion och datastrukturer

Elias Castegren
elca7381@student.uu.se

Begrepp

i) Vad innebär det matematiskt att $f(n) = O(g(n))$ eller att $y(x) = \Omega(u(x))$?

ii) Vad betyder det att $f(x) = \Theta(g(x))$? Hur kan man uttrycka samma sak med hjälp av bara O och Ω?

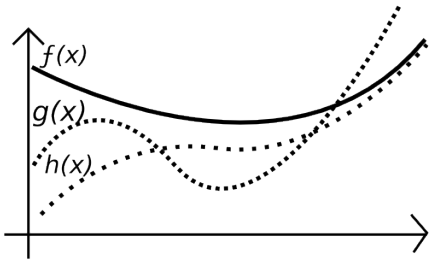
iii) Vad innebär det att prata om en funktions komplexitet? Vad är komplexiteten ett mått på?

iv) Vad kan man säga om två funktioners körtid om man vet att den ena har högre komplexitet än den andra?

v) Vad är Master Theorem och när och hur använder man det?

Övningar

1.
Vilka av nedanstående påståenden är sanna (antag att graferna inte svänger mer än vad som visas):



i) $g(x) = O(f(x))$

ii) $f(x) = \Theta(h(x))$

iii) $g(x) = \Theta(h(x))$

iv) $h(x) = \Omega(f(x))$

v) $h(x) = O(f(x))$

2.
Fyll i de blanka fälten så att nedanstående samband stämmer (ange så enkla funktioner som möjligt). Använd de formella definitionerna av Θ, Ω och O om du är osäker och vill bevisa något samband. Du kan anta att alla n är heltal större än 0.

$i)\ 3n^3 + 15n^2 + 3 = \Theta(\ldots)$ $\quad ii)\ \ldots = O(42)$ $\quad iii)\ n \cdot \log_{10} n + 3 = \Omega(\ldots)$

$iv)\ n^2 + 100n + 15 = O(\ldots)$ $\quad v)\ n \cdot \log_2 n + n = \Theta(\ldots)$

$vi)\ n^2 + \log_2 n = \ldots (n \cdot \log_2 n)$ $\quad vii)\ 2^n = \ldots (n!)$

3.
Lös följande rekursioner med hjälp av Master Theorem. Ange först värdena på a, b och $f(n)$. Om MT inte är applicerbar för någon rekursion, ange varför (du behöver då inte heller lösa rekursionen).

$i)\ T(n) = 2T(\frac{n}{2}) + n$ $\quad ii)\ T(n) = 4T(\frac{n}{2}) + n^2 \cdot \lg n$ $\quad iii)\ T(n) = 10T(\frac{n}{3}) + 2n^2$

$iv)\ T(n) = T(\frac{n}{2}) + n$ $\quad v)\ T(n) = 2^n \cdot T(\frac{n}{2}) + n^4$

4.
Antag att du har tvättat och ska sortera dina strumpor som nu ligger i en hög. Varje par är unikt (en strumpa har exakt en annan strumpa som den hör ihop med) men eftersom du bara har svarta strumpor är det enda sättet att hitta ett passande par att jämföra en strumpa med alla andra strumpor i högen en och en tills du hittar rätt. Hur många fler jämförelser måste man göra (i värsta fall) om man fördubblar antalet strumpor? Gör en komplexitetsanalys och ange komplexiteten (med avseende på antalet strumpjämförelser) för det värsta, bästa och genomsnittliga fallet för att sortera strumpor på det här sättet!

5.
Betrakta nedanstående funktioner:

```
fun f(0) = 0
  | f(n) = if 2*n>42 then f(n-1) else f(n-1)+1;

fun g(0) = 1
  | g(n) = g(n-1) + g(n-1);
```

Ge en rekursiv beskrivning av tidskomplexiteten $T(n)$ och finn en sluten formel för varje rekursion, t.ex. genom att gissa en sluten formel och bevisa den med induktion. Bestäm tidskomplexiteten med asymptotiskt närliggande gränser (Θ).

Lycka till!