Korrekthet och komplexitet

Isak Livner Mäkitalo

15 November 2023

1. a) Vill visa att:

$$VL(n) = HL(n)$$

Där:

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

$$HL(n) = \frac{n(n+1)(2n+1)}{6}$$

Basfall $n = 1$:

$$VL(1) = \sum_{i=1}^{1} i^2 = 1^2 = 1$$

$$HL(1) = \frac{1(1+1)(2+1)}{6} = \frac{6}{6} = 1$$

Induktionsantagande: $VL(s) = HL(s)$ för något $s \geq 1$
Induktionssteg: Visa att $VL(s+1) = HL(s+1)$

$$VL(s+1) = \sum_{i=1}^{s+1} i^2 = \sum_{i=1}^{s} i^2 + (s+1)^2 = VL(s) + (s+1)^2 =$$

$$\{\text{Enligt induktionsantagandet}\}$$

$$= HL(s) + (s+1)^2 = \frac{s(s+1)(2s+1)}{6} + (s+1)^2 =$$

$$= \frac{s(s+1)(2s+1)}{6} + \frac{6(s+1)(s+1)}{6} = \frac{s(s+1)(2s+1) + 6(s+1)(s+1)}{6} =$$

$$= \frac{(s+1)(s(2s+1) + 6(s+1))}{6} = \frac{(s+1)(s(2s+1) + 2(2s+1) + 2(s+2))}{6} =$$

$$= \frac{(s+1)((s+2)(2s+1) + 2(s+2))}{6} = \frac{(s+1)(s+2)(2s+3)}{6} =$$

$$= \frac{(s+1)((s+1)+1)(2(s+1)+1)}{6} = HL(s+1)$$

Med hjälp av induktionssteget har vi då visat att

$$VL(s) = HL(s) \implies VL(s+1) = HL(s+1)$$

Eftersom vi vet att $VL(1) = HL(1)$ innebär då detta att $VL(n) = HL(n)$ för $n \geq 1$

b) Vill visa att:

$$VL(n) = HL(n)$$

Där:

$$VL(n) = \sum_{j=1}^{n}(2j-1)$$

$$HL(n) = n^2$$

Basfall $n = 1$:

$$VL(1) = \sum_{j=1}^{1}(2j-1) = 2 - 1 = 1$$

$$HL(1) = 1^2 = 1$$

Induktionsantagande: $VL(s) = HL(s)$ för något $s \geq 1$
Induktionssteg: Visa att $VL(s+1) = HL(s+1)$

$$VL(s+1) = \sum_{j=1}^{s+1}(2j-1) = \sum_{j=1}^{s}(2j-1) + 2(s+1) - 1 = VL(s) + 2s + 1 =$$

$$\{\text{Enligt induktionsantagandet}\}$$

$$= HL(s) + 2s + 1 = s^2 + 2s + 1 = (s+1)^2 = HL(s+1)$$

Med hjälp av induktionssteget har vi då visat att

$$VL(s) = HL(s) \implies VL(s+1) = HL(s+1)$$

Eftersom vi vet att $VL(1) = HL(1)$ innebär då detta att $VL(n) = HL(n)$ för $n \geq 1$

2. Vi vill bevisa att följande program alltid beräknar x^n för $n \geq 0$

```
double expIterative(double x, int n) {
    double res = 1.0;

    for (int i = 0; i < n; i++) {
        res *= x;
    }
    return res;
}
```

Basfall $n = 0$:

Algoritmen ignorerar loopen och returnerar det initiella värdet av `res`, d.v.s 1. Eftersom $x^0 = 1$ för alla x, så gäller basfallet.

Induktionsantagande: $expIterative(x, s) = x^s$ för något $s \geq 0$
Induktionssteg: Visa att $expIterative(x, s + 1) = x^{s+1}$

Enligt induktionsantagandet så kommer `res` efter de s första iterationerna av loopen att vara lika med x^s. Därefter multipliceras `res` med x och programmet returnerar sedan `res`.

$$x^s \cdot x = x^{s+1}$$

Med hjälp av induktionssteget har vi då visat att

$$expIterative(x, s) = x^s \implies expIterative(x, s + 1) = x^{s+1}$$

Eftersom vi vet att $expIterative(x, 0) = x^0$ innebär då detta att $expIterative(x, n) = x^n$ för $n \geq 0$

Komplexitet:

Eftersom algoritmen endast innehåller en loop av `n` iterationer så är det lätt att se att algoritmen har $O(n)$ komplexitet.

3. Vi vill bevisa att följande program alltid beräknar x^n för $n \geq 0$

```
double expRecursive(double x, int n) {
   if (n <= 4) {
      return expIterative(x, n);
   }

   return expRecursive(x, n/2) *
          expRecursive(x, (n + 1)/2);
}
```

Basfall $0 \leq n \leq 4$:

om $0 \leq n \leq 4$ så kallar algoritmen på $expIterative(x, n)$ som vi tidigare bevisade var korrekt, så algoritmen måste vara korrekt för $0 \leq n \leq 4$

Induktionsantagande: Det finns något $s \geq 4$ sådant att $expRecursive(x, k) = x^k$ om $0 \leq k \leq s$
Induktionssteg: Visa att $expRecursive(x, s + 1) = x^{s+1}$

Då $s \geq 4$ ger algoritmen att

$$expRecursive(x, s+1) = expRecursive(x, d_1) \cdot expRecursive(x, d_2)$$

där

$$d_1 = \lfloor \frac{s+1}{2} \rfloor$$

$$d_2 = \lfloor \frac{(s+1)+1}{2} \rfloor = \lfloor \frac{(s+1)}{2} + \frac{1}{2} \rfloor$$

Om $s+1$ är jämn har vi att $d_1 + d_2 = \frac{s+1}{2} + \frac{s+1}{2} = s+1$
Om $s+1$ är udda har vi att $d_1 + d_2 = \frac{s}{2} + \frac{s+2}{2} = s+1$
I båda fallen har vi alltså att $d_1 + d_2 = s+1$

Eftersom $d_1 \leq s$ och $d_2 \leq s$ gäller det enligt induktionsantagandet att

$$expRecursive(x, d_1) \cdot expRecursive(x, d_2) = x^{d_1} \cdot x^{d_2} = x^{s+1}$$

Med hjälp av induktionssteget har vi då visat att $expRecursive(x, s+1) = x^{s+1}$ om $expRecursive(x, k) = x^k$ för alla $0 \leq k \leq s$ där $s \geq 4$
Eftersom vi vet att $expRecursive(x, n) = x^n$ för $0 \leq n \leq 4$ så gäller det då att $expRecursive(x, n) = x^n$ för $n \geq 0$

Komplexitet:

$f(n)$ har $O(1)$ komplexitet vilket innebär att $d = 0$
Antalet delproblem $a = 2$
Storleken av varje delproblem ges av $\frac{n}{2}$ vilket innebär att $b = 2$

Då $a > b^d$ så säger mästarsatsen att algoritmen har komplexiteten:

$$O(n^{log_b(a)}) = O(n^{log_2(2)}) = O(n)$$