Complexiteit

Uitwerkingen

Opgave 8.

a. Beste geval: $\frac{1}{2}n(n+1) \in \Theta(n^2)$ als de array al gesorteerd is.
b. Slechtste geval: $\Theta(n^3)$, op een omgekeerd gesorteerde array.

Opgave 9.

a. Gegeven de functie $f(n)$ voor algoritme A en $g(n)$ voor algoritme B:

$$f(n) = n^2 + 4n, \qquad g(n) = 29n + 3.$$

We zien dat $f(n) \in \Theta(n^2)$ en $g(n) \in \Theta(n)$, waaruit we kunnen concluderen dat $g(n) < f(n)$ voor voldoende grote n, en algoritme B dus sneller is.
b. Gegeven de functie $h(n)$ voor algoritme A en $\ell(n)$ voor algoritme B:

$$h(n) = 5n^2 + n \log n + 3, \qquad \ell(n) = 2n^2 + n - 4.$$

We zien dat zowel $h(n) \in \Theta(n^2)$ als $\ell(n) \in \Theta(n^2)$, waaruit we niet kunnen concluderen welke sneller is. Wel zien we dat de constante factor voor de dominante term voor $\ell(n)$ een factor $2/5$ keer zo klein is als voor $h(n)$, dus algoritme B is sneller (onder de aanname dat een enkele stap van algoritme B even veel tijd kost als als een enkele stap van algoritme A).

Opgave 10.

Voor deze uitwerking gebruiken we de notatie $f(n) < g(n)$ om aan te geven dan een functie $f(n)$ voor alle waarden van $n \geq n'$ kleiner is dan $g(n)$. Oftewel

$$f(n) < g(n)$$

is kort voor

$$\exists n' \text{ s.t. } \forall n \geq n' : f(n) < g(n).$$

Om het vergelijken van alle gegeven functies wat makkelijker te maken groeperen we ze eerst (van snel naar langzaam) op constante, logaritmische, polynomiale, en exponentiële complexiteit.

Constant Complexiteit onafhankelijk van de grootte van de invoer n.

$$1 \in O(1)$$

Logaritmisch (We nemen hier ook de dubbele logaritmische functies en poly-logaritmische functies mee.)
Side note: Elke logaritmische functie $f(n) = \log_a(n)$ behoort tot dezelfde klasse $\Theta(\log n)$, ongeacht het grondtal a.

$$\log_2 n, \log_3 n, \ln n \in \Theta(\log n)$$

Bewijs: tot op een constante factor k zijn de functies $\log_a(n)$ en $\log_b(n)$ gelijk:

$$\log_a(n) = \frac{\log_b(n)}{\log_b(a)} = k \cdot \log_b(n), \text{ met } k = \frac{1}{\log_b(a)}.$$

Geordend op complexiteit:

$$\ln(\ln(n)) < \sqrt{\ln(n)} < \log_3(n) < \ln(n) < \log_2(n) < \log_2^2(n)$$

Polynomiaal Een polynomiale functie is een functie van de vorm $f(x) = a_k x^k + \cdots + a_2 x^2 + a_1 x + a_0$, met $k \in \mathbb{N}$. Let op dat bijvoorbeeld $n \log_2(n)$ niet aan deze definitie voldoet, maar wel qua complexiteit tussen polynomiale functies valt, dus deze nemen we ook hier mee.
Beginnend met duidelijke polynomiale functies:

$$n, n+9 \in \Theta(n) \qquad n^2 + \log_2(n) \in \Theta(n^2)$$
$$n^3 \in \Theta(n^3) \qquad n^3 + n^7 + 8 \in \Theta(n^7)$$

Minder duidelijke polynomiale functies:

$$2^{\log_2 n} = n$$
$$4^{\log_2 n} = (2^2)^{\log_2 n} = 2^{2\log_2 n} = (2^{\log_2 n})^2 = n^2$$

En niet-polynomiale functies die wel polynomiale complexiteit hebben:

$$n \log_2(n) \in \Omega(n)$$
$$n \log_2(n) \in O(n^2)$$

$$\log_2(n!) \in \Theta(n \log n)$$

Aantonen dat $\log_2(n!) \in \Theta(n \log n)$, en aantonen dat $\log_2(n!) < n \log_2(n)$ wordt gelaten als een oefening voor de lezer. (En besproken tijdens het werkcollege.)

$$n^{1+\varepsilon} = n \cdot n^{\varepsilon}$$
$$n \log(n) < n \cdot n^{\varepsilon} < n^2 \text{ (aangenomen dat } 0 < \varepsilon < 1)$$

Alles geordend op complexiteit:

$$2^{\log_2 n} = n < n + 9 < \log_2(n!) < n \log_2(n) <$$
$$n^{1+\varepsilon} < 4^{\log_2 n} < n^2 + \log_2(n) < n^3 < n^7 + n^3 + 8$$

Exponentiële functies Een exponentiële functie is een functie van de vorm $f(x) = ab^x$. De functie $n!$ is geen exponentiële functie maar valt qua complexiteit tussen een exponentiële functie en een dubbele exponentiële functie:

$$n! \in \Omega(2^n)$$
$$n! \in O(2^{2^n})$$

Geordend op complexiteit:

$$(3/2)^n < 2^n < n2^n < e^n < n! < (n+1)! < 2^{2^n} < 2^{2^{(n+1)}}$$