Einschub: Wie viele Ziffern benötigt man für eine Zahl in Binär- oder Dezimaldarstellung?

Beispiele:

Binär	Dezimal	Stellen (binär)	(dezimal)
1	1	1	1
10	2	2	1
101	5	3	1
1111111111	1.023	10	2
10000000000000000	65.536	17	5

Eine Zahl n mit b Binärstellen liegt zwischen 2^{b-1} und 2^b,
mit d Dezimalstellen liegt zwischen 10^{d-1} und 10^d:

$$2^{b-1} \le n < 2^b$$

bzw. $$10^{d-1} \le n < 10^d$$

Also gilt $$b-1 \le \log_2 n < b$$

bzw. $$d-1 \le \log_{10} n < d \,,$$

d.h. $$b = \lfloor \log_2 n \rfloor + 1$$

bzw. $$d = \lfloor \log_{10} n \rfloor + 1$$

Außerdem gilt mit $$2^b = \left(2^{\log_2 10}\right)^{\frac{b}{\log_2 10}} = 10^{\frac{b}{\log_2 10}} \,,$$

also $d \approx \frac{1}{\log_2 10} b$, d.h. zwischen der binären und dezimalen Stellenzahl gibt es eine Umrechnungskonstante von $\log_2 10 \approx 3{,}3219\ldots$

(Das gilt nur ungefähr, weil wir ja ganze Zahlen haben und runden!)

Für einen Graphen mit m Kanten und n Knoten ergibt sich in obiger (ausführlicher) Codierung ein Platzbedarf von

$$(6m-1) + 2m\left(\lfloor \log_{10} n\rfloor + 1\right).$$

Dabei kann man an ein paar Stellen sparen (z.B. „v" oder „{", „}" weglassen) aber auch mehr Platz investieren (z.B. in ASCII codieren bzw. binär statt dezimal codieren). So wäre auch

$$(2m-1) + 2m\left(\lfloor \log_2 n\rfloor + 1\right) \quad \text{denkbar.}$$

Was ist wirklich wichtig dabei?!

(i) Die Kantenliste ist sparsamer als die Inzidenzmatrix, denn wenn n nicht zu klein ist, dann ist

$$n \geq 2 + 2\left(\lfloor \log_2 n\rfloor + 1\right).$$

(was heißt „nicht zu klein"? $n \geq 16$ reicht!)

Also ist auch

$$mn > (2m-1) + 2m\left(\lfloor \log_2 n\rfloor + 1\right).$$

(ii) Unabhängig von der Codierung ist für die Größe des Speicherplatzes der zweite Ausdruck wichtig, denn

$$\begin{aligned} 2m\left(\log_2 n + 1\right) &\leq (2m-1) + 2m\left(\lfloor \log_2 n\rfloor + 1\right) \\ &\leq 4m\left(\log_2 n + 1\right) \quad \text{für } n \geq 2. \end{aligned}$$

(iii) Letztlich kommt es also gar nicht so sehr auf die Vorfaktoren an (die sind codierungsabhängig!), sondern auf den Ausdruck

$$m \log n .$$

(iv) In $m \log n$ steckt das Wesen der Kantenliste: Zähle für alle m Kanten die Nummern der beteiligten Knoten auf!

Wie wir im nächsten Abschnitt sehen werden, lohnt sich dafür eine eigene Notation:

Die Kantenliste benötigt $\Theta(m \log n)$ Speicherplatz.

3.7 Wachstum von Funktionen

Im letzten Abschnitt haben wir Funktionen abgeschätzt und auf die „wesentlichen" Bestandteile reduziert, um Größenordnungen und Wachstumsverhalten zu beschreiben. Ein bisschen formaler:

Definition 3.9 (Θ-Notation)

Seien $f, g : \mathbb{N} \to \mathbb{R}$ Funktionen.

Dann gilt

$f \in \Theta(g) :\Leftrightarrow$ Es gibt positive Konstanten c_1, c_2, n_0 mit $0 \le c_1 g(n) \le f(n) \le c_2 g(n)$ für alle $n \ge n_0$.

Man sagt: f wächst asymptotisch in derselben Größenordnung wie g.

Beispiele:

$$2n^2 - 1 \in \Theta(n^2)$$

$$\frac{n^3}{1000} + n^2 + n \log n \in \Theta(n^3)$$

Manchmal hat man nur untere oder obere Abschätzungen:

DEFINITION 3.10 (O-Notation)

Seien $f, g : \mathbb{N} \to \mathbb{R}$ Funktionen.

Dann gilt $\quad f \in \Theta(g) :\Leftrightarrow$ Es gibt positive Konstanten c, n_0 mit

$$0 \leq f(n) \leq c\, g(n) \quad \text{für alle} \quad n \geq n_0.$$

Man sagt: $\quad f$ wächst ~~exponentiell~~ höchstens in derselben Größenordnung wie g.

Beispiele:

$$2n^2-1 \in O(n^2)$$
$$2n^2-1 \in O(n^3)$$
$$n \log n \in O(n^2)$$

DEFINITION 3.11 (Ω-Notation)

Seien $f, g : \mathbb{N} \to \mathbb{R}$ Funktionen.

Dann gilt $\quad f \in \Omega(g) :\Leftrightarrow$ Es gibt positive Konstanten c, n_0 mit

$$0 \leq c\, g(n) \leq f(n) \quad \text{für alle} \quad n \geq n_0.$$

Beispiele: $2n^2-1 \in \Omega(n^2)$

$2n^2-1 \in \Omega(n^2)$

Einige einfache Eigenschaften:

SATZ 3.12

Seien $f, g : \mathbb{N} \to \mathbb{R}$ Funktionen.

Dann gilt

(1) $f \in \Theta(g) \Leftrightarrow g \in \Theta(f)$

(2) $f \in \Theta(g) \Leftrightarrow f \in O(g)$ und $f \in \Omega(g)$.

(3) $f \in O(g) \Leftrightarrow g \in \Omega(f)$

Beweis: Übung!