Theoretische Informatik: Blatt 2

Abgabe bis 2. Oktober 2015
Assistent: Sascha Krug, CHN D 42

Linus Fessler, Markus Hauptner, Philipp Schimmelfennig

Aufgabe 4

(a) Für $w_n = 0^{2^{2^{5 \cdot n^2}}} \in \{0, 1\}^*$ gilt, dass $|w_n| = 2^{2^{5 \cdot n^2}}$

$$\begin{aligned}
&& |w| &= 2^{2^{5\cdot n^2}} \\
\Rightarrow && log_2\ |w_n| &= 2^{5\cdot n^2} \\
\Rightarrow && log_2\ log_2\ |w_n| &= 5 \cdot n^2 \\
\Rightarrow && \frac{1}{5} \cdot log_2\ log_2\ |w_n| &= n^2 \\
\Rightarrow && \sqrt{\frac{1}{5} \cdot log_2\ log_2\ |w_n|} &= n
\end{aligned}$$

$$K(z_n) \leq \lceil log_2\ n + 1 \rceil + d \leq \lceil log_2\ n \rceil + d' \leq \lceil\ log_2 \left(\sqrt{\tfrac{1}{5} \cdot log_2\ log_2\ |w_n|}\ \right) \rceil + d'$$

Aufgabe 5

Wir betrachten Wörter über dem Alphabet $\Sigma^*_{bool} = \{0, 1\}^*$ mit höchstens Länge n, $|w| \leq n$
Nach der Definiton von Zufälligkeit ist $K(x_n) \geq |w_n| \Leftrightarrow x_n$ ist zufällig
Also soll $K(x_n) < |w_n|$ sein, damit w_n komprimierbar ist. Jedes Programm, das solch ein Wort darstellt ist selbst eine Bit-Sequenz in Σ_{bool} mit Länge bis $n-1$. Nähmen wir an, dass jede Bit-Sequenz in $\{\Sigma_{bool}\}^{n-1}$ sei in sinnvolles Programm, das ein Wort generieren kann, dann ist die maximale Anzahl dieser Programme

$$\sum_{k=1}^{n-1} 2^k = 2^n - 2$$

Die Anzahl aller möglichen Wörter bis Länge n ist:

$$1 + \sum_{k=1}^{n} 2^k = 2^n - 2 + 2^n$$

Falls ein Programm ein Wort komprimiert, gibt es eine Bijektion zwischen diesem Wort und dem Programm. Ziehen wir von der Anzahl aller möglicher Wörter die Anzahl aller möglicher -diese Wörter komprimierender-Programme ab, bleiben noch

$$1 + \sum_{k=1}^{n} 2^k \ - \sum_{k=1}^{n-1} 2^k = 1 + 2^n - 2 + 2^n - (2^n - 2) = 1 + 2^n$$

Wörter übrig, die nicht von Programmen in $\{\Sigma_{bool}\}^{n-1}$ generiert werden können. Dies sind mehr als die $2^n - 2$ möglichen Programme, also als die Anzahl potentiell komprimierbarer Wörter. Daher ist mehr als die Hälfte der Wörter nicht komprimierbar, also zufällig.

Aufgabe 6

Das Wort $w = 1^i 0^j 1^k$ ist eindeutig definiert durch i und k, da $i + j = 2k \Rightarrow j = 2k - i$. Als Eingabe für unser Programm wählen wir eine Codierung, die i, k beinhaltet:

$$X = \overline{\mathrm{Bin}}(i)\mathrm{Bin}(k) = i_1 0 i_2 0 i_3 0 \cdots a_{\lceil log_2 i+1 \rceil} 1 k_1 k_2 k_3 \cdots k_{\lceil log_2 k+1 \rceil}$$

Da $i \leq 2k$ ist, ist die Eingabelänge beschränkt Damit ist die Länge der Eingabe

$$|X| = 2\left(\lceil \log_2 (i+1) \rceil\right) + \lceil \log_2 (k+1) \rceil$$

und das Programm hat Kolmogorov Komplexität

$$\begin{aligned} K(w) &\leq 2 \cdot \lceil \log_2 (i+1) \rceil + \lceil \log_2 (k+1) \rceil + c \\ &\leq 2 \cdot \log_2 (i) + \log_2 (k) + c' \\ &\leq 2 \cdot \log_2 (2k) + \log_2 (k) + c' \end{aligned} \tag{1}$$