Serie 2

André Graubner, Lukas Walker, Leonard von Kleist

28. September 2018

1 Aufgabe 4

Sei L eine unendliche Sprache über einem Alphabet Σ.

Richtung 1 ($\Rightarrow$): Sei L rekursiv. Das heisst per Definition, dass ein Algorithmus A_{erk} existiert, der L erkennt. Wir konstruieren einen Aufzählungsalgorithmus (mit Eingabe n) für L: Iteriere in kanonischer Reihenfolge durch Σ^* und überprüfe mit A_{erk} für jedes Element, ob es $\in L$ ist. Falls ja, gebe es aus. Fahre so lange fort, bis wir n Elemente ausgegeben haben.

Richtung 2 ($\Leftarrow$): Sei A_{enum} (mit Eingabe n) ein Aufzählungsalgorithmus für L. Wir konstruieren einen Algorithmus (mit Eingabe x), der L erkennt: Bezeichne k die Stelle, an der das Wort x kanonisch in Σ^* steht. Rufe A_{enum} mit Eingabe k auf. Wenn eine der von A_{enum} ausgegebenen Wörter x ist, so ist $x \in L$, also geben wir 1 (bzw. "ja", etc) aus, sonst 0.

2 Aufgabe 5

Im Intervall $[2^n, 2^{n+1} - 1]$ gibt es $2^{n+1} - 2^n - 1$ verschiedene Zahlen. Wir betrachten jeweils lediglich die kürzesten Maschinencodes, die diese Zahlen generieren. Weiterhin ist klar, dass diese Codes für zwei verschiedene Zahlen unterschiedlich sein müssen. Es reicht also zu zeigen, dass von $2^{n+1} - 2^n - 1$ paarweise unterschiedlichen binären Strings für jedes $i < n$ mindestens $2^n - 2^{n-i}$ die Länge $n - i$ haben.

Die Anzahl der (nichtleeren) Bitstrings von Länge kleiner als $n - i$ ist

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

Wenn wir diese Anzahl nun jedoch von der Menge an Zahlen in unserem Intervall abziehen, sehen wir:

$$2^{n+1} - 2^n - 1 - 2^{n-i} + 2 = 2^{n+1} - 2^n + 1 - 2^{n-i} \geq 2^n - 2^{n-i}$$

Also gilt für mindestens $2^n - 2^{n-i}$ der Zahlen $K(x) \geq n - i$.