# Berechenbarkeit und Unentscheidbarkeit

Die Funktion $f_1 : \{a, b\}^* \to a, b^*$ mit

$$f_1(w) = \begin{cases} w.w^{reverse} & \text{if } |w| \text{ is odd} \\ w_1 & \text{if } w = w_1w_2 \text{ with } |w_1| = |w_2| \end{cases}$$

kann von dem folgenden Algorithmus berechnet werden:

Eingabe: $w$
**while** Buchstaben unmarkiert sind **do**
  Markiere den linkesten unmarkierten Buchstaben.
  **if** Buchstaben unmarkiert sind **then**
    Markiere den rechtesten unmarkierten Buchstaben.
  **else**
    Die Eingabelänge war ungerade.
    **for** jeden markierten Buchstaben, von rechts nach links, **do**
      Hebe Markierung auf.
      Hänge eine Kopie an das rechte Ende an.
    **end for**
    **return** $w$.
  **end if**
**end while**
Die Eingabelänge war gerade, der Kopf befindet sich genau über der Mitte.
Lösche alle rechten Buchstaben, hebe alle Markierungen auf.
**return** die linken Buchstaben.

Da dieser Algorithmus mit einem Band verfasst wurde (siehe Buchstaben, Markierungen und Richtungen), sollte es nicht schwer sein, diesen als Turing-Maschine zu implementieren.

Die Funktion $f_2 : \mathbb{N} \to \mathbb{N}$ mit

$$f_2(n) = 2^n \cdot n$$

kann von dem folgenden Algorithmus berechnet werden:

Idee: Die Binärdarstellung (most-significant-bit-first) von $2^n$ ist $1.0^n$. Multiplikation mit $n$ ersetzt die 1 mit der Binärdarstellung von $n$.

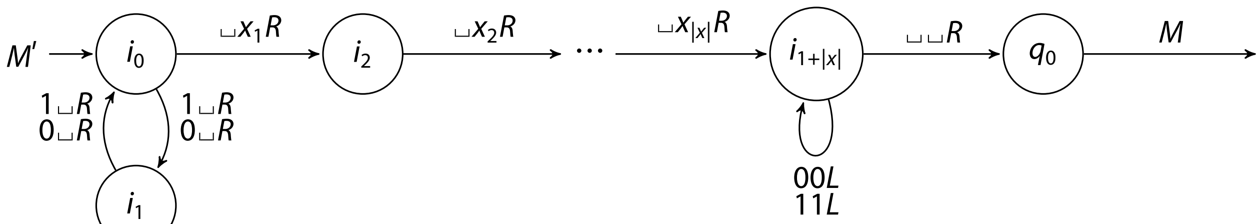
Eingabe: bin($n$) $\in \{0,1\}^*$
Schreibe bin($n$) in die Ausgabe.
**for** i = 1…n **do**
  Schreibe 0 in die Ausgabe.
**end for**
**return** bin($n$).$0^n$.

Eine Turing-Maschine, die diesen Algorithmus implementiert, bedient sich also eines Arbeitsbandes für einen Zähler $i$. Das Kopieren und das Erhöhen des Zählers wurde schon mal implementiert und kann hier als Subroutine benutzt werden.

Die Sprache $L_3 := \{w \in \{0,1\}^* \mid \exists x \in \mathcal{L}(M_w) : |x| \text{ is even}\}$ kann durch den folgende Algorithmus erkannt werden, ist also semi-entscheidbar:

Eingabe: kodierte Turing-Maschine $M_w$
**for** $k = 0 \ldots \infty$ **do**
  **for** $x \in \{0,1\}^{\leq k}$ **do**
    **if** |x| is even **then**
      Simulate $M_w$ on $x$ for at most $k$ steps.
      **if** Simulation accepted **then**
        **accept**
      **end if**
    **end if**
  **end for**
**end for**

$L_3$ ist allerdings nicht co-semi-entscheidbar. (Wir zeigen ACCEPT $\leq$ $L_3$): Dazu konstruieren wir eine berechenbare Funktion, die ein Paar aus einer kodierten Turing-Maschine $M$ und einem Eingabe-Wort $x$ erhält und daraus eine (andere) kodierte Turing-Maschine $M'$ erstellt.




Diese Transformation ist berechenbar, schließlich sind alle benötigten Informationen explizit gegeben.

Es gilt nun $x\#w \in$ ACCEPT genau dann, wenn $x \in \mathcal{L}(M_w)$ gilt. Das gilt nach Konstruktion genau dann, wenn $\mathcal{L}(M'_w) \neq \varnothing$ ist, bzw. wenn $\mathcal{L}(M'_w) = (\{0,1\}\{0,1\})^*$ ist. Wir charakterisieren weiter, denn das gilt genau dann, wenn es ein $x$ gerader Länge in $\mathcal{L}(M'_w)$ gibt, also wenn $\langle M'_w \rangle \in L_3$ gilt.