wird, so betrachtet man stattdessen die Sprache $L = \{(x, y, z) \mid x + y = z\}$. Hat man bereits eine Turingmaschine, die f berechnet, lässt sich leicht eine konstruieren, die L akzeptiert. Die neue Turingmaschine benutzt die alte als Subprogramm, um $x + y$ zu berechnen und vergleicht das Ergebnis mit z. Sie akzeptiert genau dann, wenn dieser Vergleich positiv ist.

Wir fassen wieder zusammen. In diesem Abschnitt haben wir gesehen, dass Turingmaschinen nicht nur Sprachen akzeptieren können, sondern, dass sie auch Funktionen berechnen können. Die Argumente und Bilder der Funktion müssen nur geeignet kodiert werden, so dass die Turingmaschine mit diesen Kodierungen arbeiten kann. Im Anschluss haben wir dann gesehen, dass die beiden Betrachtungen sich gut ineinander überführen lassen. Zu wissen, dass man mit Turingmaschinen Funktionen berechnen kann, ist aber hilfreich, um sich zu verdeutlichen, dass Turingmaschinen eine formale Abstraktion bzw. ein formales Modell für einen Algorithmus sind.

Fragen

1. Betrachten Sie folgende Definition von berechenbar.
   a) $f : \Sigma^* \to \Sigma^*$ ist **berechenbar** gdw.
   b) es eine DTM A gibt mit $z_0 w \vdash^* zv$
   c) genau dann, wenn
   d) $f(w) = v$ ist

   In welcher Zeile ist ein Fehler?
   a) 1
   b) 2
   c) 3
   d) 4

4.1.3 Die Church-Turing-These

Wir haben im letzten Abschnitt gesehen, wie eine DTM eine Funktion berechnet und haben mehrere Beispiele gesehen, wie eine DTM spezielle Funktionen berechnet. Man kann sich vielleicht bereits vorstellen, dass neben der Addition und der Subtraktion auch alle anderen Rechenoperationen möglich sind, die ein Mensch auf einem Papier machen kann. Im Grunde genommen folgt jede Rechnung, die ein Mensch auf einem Papier machen kann, einer genauen Vorschrift und ist damit ein Algorithmus. Dies lässt sich dann auch in eine Turingmaschine kodieren, die damit alles machen kann, was ein Mensch machen kann. Die Church-Turing-These, die wir eingangs schon erwähnten, besagt genau dies. Sie ist so bedeutend, dass sie uns diesen eigenen Abschnitt wert ist. Die Church-Turing-These besagt folgendes:

> Alles was intuitiv berechenbar ist, d.h. alles, was von einem Menschen berechnet werden kann, das kann auch von einer Turingmaschine berechnet werden. Ebenso ist alles, was eine andere Maschine berechnen kann, auch von einer Turingmaschine berechenbar.

Für uns von besonderer Bedeutung ist auch der Umkehrschluss: Was eine Turingmaschine nicht berechnen kann, kann auch kein Mensch berechnen!

Da die Turingmaschine ein formales Modell ist, kann man exakt argumentieren, was diese nicht kann bzw. versuchen dies zu tun. Gelingt dies, so hat man eine Funktion gefunden, die auch ein Mensch nicht berechnen kann. Wir werden später sehen, dass sich hier ganz grundlegende Problemstellungen der Informatik verstecken, die einer algorithmischen Lösung also nicht zugänglich sind.

Fragen

1. Wenn $f : \mathbb{N} \to \mathbb{N}$ und $g : \mathbb{N} \to \mathbb{N}$ Turing-berechenbar sind, sind dann auch $f + g$ und $f \cdot g$ Turing-berechenbar?
   a) Ja!
   b) Nein!
2. Wenn $f : \mathbb{N} \to \mathbb{N}$ und $g : \mathbb{N} \to \mathbb{N}$ Turing-berechenbar sind, ist dann auch $f \circ g$ Turing-berechenbar?
   a) Ja!
   b) Nein!