Friedrich-Schiller-Universität Jena
Fakultät für Mathematik und Informatik

Berechenbarkeit des Denkens?

Examensarbeit
zur Erlangung des akademischen Grades
Bakkalaureus der Mathematik

vorgelegt von: **Lars Mäurer**
geboren am: **11. September 1976**
Geburtsort: **Jena**
Betreuer: **Prof. Dr. G. Wechsung**
Tag der Einreichung: **22. August 2003**

Zusammenfassung

Die Idee einer denkenden Maschine übt eine besondere Faszination auf uns aus, macht sie doch eine Fähigkeit, die uns als Menschen definiert, zum Kopierbaren, zum Programm.

Die Fortschritte der Computertechnik in den letzten Jahrzehnten scheinen uns die Möglichkeit immer näher zu bringen, menschliches Denken künstlich zu erzeugen. Geschwindigkeit, Speicherkapazität und logisches Design werden sicherlich in den nächsten Jahren auch weiteren Fortschritten unterworfen. Doch ist künstliche Intelligenz wirklich nur ein Problem schnellerer Computer, größeren Speichers und ausgefeilterer Algorithmen?

Roger Penrose, Rouse-Ball-Professor an der Universität Oxford, führt in seinem Buch „Schatten des Geistes“ [3] starke (und schwache) Argumente gegen zuviel Optimismus ins Feld. Dieses vielfach kontrovers diskutierte Werk enthält einen faszinierend einfachen, aber bedeutenden Gedanken, der die Berechenbarkeit des menschlichen Denkens in Frage stellt. Diesen Gedanken einschließend soll diese Thematik in dieser Arbeit in gebührendem Maße beleuchtet werden.

Inhaltsverzeichnis

Kapitel 1

Einleitung / Vorbetrachtungen

In den Jahren 1900 und 1928 formulierte David Hilbert[1] das sogenannte *Entscheidungsproblem*. Er forderte ein allgemeines algorithmisches Verfahren zum Lösen mathematischer Probleme bzw. eine Antwort auf die Frage, ob ein solches Verfahren überhaupt existieren kann.

Um über die Leistungsfähigkeit von solchen algorithmischen Verfahren zu urteilen, ist es notwendig, eine formale Beschreibung dafür zu finden. Hier bieten sich die Arbeiten von Alan Turing[2] aus den Jahren 1935/36 an. Turing versuchte, die Funktionsweise solcher Verfahren mit elementaren mathematischen Ausdrücken zu beschreiben.

Dabei spielen die Begriffe Algorithmus und Turing-Maschine eine wichtige Rolle, die wir in den nächsten Abschnitten vorstellen wollen.

1.1 Algorithmen

Ein Algorithmus ist ein systematisches Verfahren, mit Hilfe dessen eine bestimmte (sagen wir mathematische) Fragestellung gelöst werden kann.

Beispiele für Algorithmen finden sich schon lange vor Beginn unserer Zeitrechnung in der griechischen Antike. Einer der bekanntesten ist der Euklid*ische Algorithmus*[3].

---

[1] David Hilbert (1862-1943), deutscher Mathematiker.

[2] Alan Mathison Turing (1912-1954), britischer Mathematiker. Turing gilt heute als einer der einflußreichsten Theoretiker der Berechenbarkeitstheorie und der frühen Computerentwicklung.

[3] nach Euklid, Eukleides von Alexandreia (ca. 300 v.u.Z.), griechischer Naturwissenschaftler. Das Verfahren zur Bestimmung des größten gemeinsamen Teilers wurde im 7. Buch seiner Elemente erstmals vorgestellt.

Dieser Algorithmus beschreibt eine Verfahrensweise, für zwei natürliche Zahlen beliebiger Größe nach einer gewissen Zeit deren größten gemeinsamen Teiler zu bestimmen. Für größere Zahlen wird das Verfahren sicher länger brauchen, als bei sehr kleinen Zahlen, aber in jedem Einzelfall wird es schließlich nach *endlich* vielen Schritten abbrechen und das korrekte Ergebnis liefern. Dabei gibt das Verfahren *absolut exakt* an, was bei jedem einzelnen Schritt der Rechnung zu tun ist. Außerdem ist das Verfahren durch *endlich* viele Ausdrücke beschreibbar, zum Beispiel durch ein (endliches) Flußdiagramm, welches den logischen Ablauf des Verfahrens beschreibt.

Dies sind genau die Anforderungen, die wir ganz allgemein an einen Algorithmus stellen wollen: Algorithmen sind also systematische Verfahren, die uns exakt beschreiben, wie zu jedem Zeitpunkt vorgegangen werden soll, die nach endlicher Zeit zum Stillstand (und damit einem Ergebnis) gelangen, und die mit endlich vielen Ausdrücken beschrieben werden können.

In den dreißiger Jahren entstanden mehrere präzise Beschreibungen des Begriffes Algorithmus. Der überzeugendste und auch historisch wichtigste Ansatz dieser Art wurde von dem bahnbrechenden Informatiker Alan Turing hervorgebracht und verwendet den Gedanken der Turing-*Maschine*.

1.2 Turing-Maschinen

Wir stellen uns ein Band vor, welches sich nach links und rechts frei und unendlich weit entfaltet und gleichmäßig in (wieder-)beschreibbare Felder unterteilt ist. In jedes Feld paßt genau ein Zeichen aus einem vorher definierten Alphabet.

Unser Alphabet sei $\{0, 1, \_\}$, wobei „_“ als Leerzeichen verstanden werden soll, welches sich auf allen Feldern befindet, die nicht durch die *(echten)* Zeichen „0“ und „1“ beschrieben sind. Wir werden später sehen, daß die Beschränkung auf dieses recht kleine Alphabet keine echte Einschränkung der prinzipiellen Leistungsfähigkeiten von Turing-Maschinen darstellt.

In Turings Vorstellung besteht das „Band“ aus einer linearen, in beide Richtungen unendlichen Folge von Quadraten.

Eine Turing-*Maschine* ist nun ein Apparat, der sich von Feld zu Feld auf diesem Band hin- und herbewegen kann. Dieser Apparat kann das Zeichen, über dem er sich befindet, lesen und durch ein anderes ersetzen. Außerdem befindet sich der Apparat in einem internen Zustand, den er selbst verändern kann. Dies alles wird gesteuert durch ein Programm.

Dieses Programm gibt für jeden Zeittakt (abhängig vom momentanen Zustand und dem gelesenen Zeichen) genau an, durch welches Zeichen das gelesene Zeichen ersetzt werden soll, in welche Richtung sich der Apparat bewegen soll (die

Maschine bewegt sich immer nur genau ein Feld nach rechts oder links) und in welchen internen Zustand er übergehen soll.

Diese einfache Beschreibung einer Maschine folgt bei genauer Betrachtung exakt der Handlungsweise eines Algorithmus in elementarer Form:

Man bekommt einen Input (in diesem Falle das Zeichen, welches die Maschine auf dem Band vorfindet), gibt einen Output aus (das Zeichen, welches sie auf dem Band hinterläßt) und begibt sich in einen anderen Zustand. Die Bewegungsfähigkeit der Maschine sichert, daß sie sich auch anderen Inputs zuwenden kann, ebenso wie selbst niedergeschriebenen Zwischenergebnissen.

Die Maschine beendet ihre Berechnungen (sie hält an), wenn sie den besonderen Zustand STOP, den sogenannten End-Zustand, erreicht.

An dieser Stelle ist ein kleines Beispiel nötig:

Die Eingabe für die TURING-*Maschine (nennen wir sie* NF*, wie „Nachfolger“) besteht aus einer endlichen Folge von Nullen und Einsen, die als Binärzahl aufgefaßt wird.* NF *soll zu dieser Binärzahl 1 addieren.*

Sie startet im Zustand z_1 und steht an einer beliebigen Stelle der Eingabe, also der auf das Band geschriebenen Binärzahl.

Hier zunächst einmal das Programm:

NF	0	1	_
z_1	$0 \rightarrow z_1$	$1 \rightarrow z_1$	$\_ \leftarrow z_2$
z_2	$1 \leftarrow z_3$	$0 \leftarrow z_2$	$1 \leftarrow z_3$
z_3	$0 \leftarrow z_3$	$1 \leftarrow z_3$	STOP

Hierbei stehen in den Zeilen der Tabelle die Zustände, in denen sich die Maschine befinden kann. In den Spalten findet man für jedes Zeichen, das gelesen wird, die Anweisungen, wie sich die Maschine verhalten soll. Man findet also für jedes gelesene Zeichen und jeden internen Zustand einen genauen Eintrag, wie die Maschine arbeitet.

Das erste Zeichen jedes Eintrages gibt an, wodurch das gelesene Zeichen ersetzt werden soll. Der Pfeil zeigt die Bewegungsrichtung der Maschine an, und zuletzt wird der Zustand angegeben, den der Apparat einzunehmen hat.

Beobachten wir, wie die Maschine auf eine Eingabe reagiert:

Im Zustand z_1 läuft NF *solange nach rechts, bis sie das erste „_“ erreicht. Dies wird erreicht, indem alle Zeichen, die sich von „_“ unterscheiden, ignoriert werden und weiter nach rechts gelaufen wird. Findet die Maschine ein „_“, geht sie einen Schritt nach links und wechselt zum Zustand z_2. Sie befindet sich nun auf der letzten Stelle der Binärzahl.*

Im Zustand z_2 geht das eigentliche Addieren der 1 vor sich. Die Maschine ersetzt nach links gehend solange Einsen durch Nullen, bis sie zum ersten Mal auf „0“ oder „␣“ trifft. Dieses ersetzt NF *durch „1“, geht einen Schritt nach links und wechselt zum Zustand z_3. In diesem Zustand läuft* NF *bis zum nächsten „␣“ nach links und hält dann an. Die Maschine steht nun auf dem Leerzeichen direkt vor der ersten Ziffer der Binärzahl.*

Man kann sich leicht davon überzeugen, daß die angegeben Maschine tatsächlich zu jeder beliebigen Binärzahl 1 addiert.

TURING-Maschinen sind, damit sie exakt arbeiten, auf gewisse Vereinbarungen angewiesen.

Auch die hier beschriebene Maschine NF arbeitet nicht korrekt, wenn sie am Anfang nicht auf einer Ziffer einer Binärzahl (endlich!) steht:
Besteht das Band anfangs ausschließlich aus Leerzeichen, stoppt die Maschine, nachdem sie eine 1 auf das Band geschrieben hat.
Eine unendlich große Eingabe aus Nullen und Einsen läßt die Maschine scheitern (sie hält dann nicht an), da sie das Ende der Zahl niemals erreichen kann.

Deshalb legt man für TURING-Maschinen gewisse sinnvolle Vereinbarungen fest.

In jedem einzelnen Fall muß der Input *endlich* sein. Wir verstehen als Input die von Leerzeichen verschiedenen Zeichen (also Nullen und Einsen), und zusätzlich einzelne (Trennungs-)Leerzeichen, die sich zwischen Nullen und Einsen befinden. Demnach enthält das Band, obwohl es laut Annahme unendlich groß ist, zu jedem Berechnungsschritt jeweils nur endlich viele von Leerzeichen verschiedene Zeichen. In beiden Richtungen kann das Band also jenseits eines gewissen Feldes (abhängig von den bis dahin vollführten „Berechnungen“) nur noch Leerzeichen enthalten. Es ist außerdem sinnvoll, die Maschine am Anfang direkt auf dem am weitesten links stehenden Zeichen der Eingabe starten zu lassen.

Die allgemeine Forderung der endlichen Beschreibbarkeit, die wir auch an einen Algorithmus stellen wollen, stellt sich im Falle der TURING-Maschinen wie folgt dar:

Jede TURING-Maschine besteht nur aus endlich vielen internen Zuständen.

TURING-Maschinen dürfen prinzipiell beliebig lange laufen, um zu einem Ergebnis zu gelangen. Wenn eine TURING-Maschine allerdings niemals anhält, ist das Ergebnis nicht definiert. Schließlich können wir erst dann den Inhalt des Bandes als endgültigen Output interpretieren, wenn die Maschine ihre Berechnung beendet hat.

Mit diesen Vereinbarungen kann man die Leistungsfähigkeit von TURING-Maschinen folgendermaßen umschreiben:

Eine TURING-Maschine findet auf dem Band eine Eingabe vor und läßt (sofern sie nach endlicher Zeit stoppt) auf dem Band eine Ausgabe zurück. Um eine

absolut deterministische Verhaltensweise der TURING-Maschine zu garantieren, vereinbaren wir nun, daß sich die Maschine am Anfang der Berechnung immer auf dem ersten (am weitesten links stehenden) Zeichen des Inputs befinden soll.

Die oben beschriebene Maschine NF errechnet somit in der Tat den Nachfolger einer auf dem Band befindlichen Binärzahl.

Es ist prinzipiell nicht schwer, TURING-Maschinen auch schwierigere Aufgaben ausführen zu lassen, zum Beispiel das Addieren von beliebig großen Zahlen, auch das Multiplizieren der Eingabe mit festen Faktoren, ebenso das Addieren oder Multiplizieren von mehreren Eingaben, die sich, durch Leerzeichen getrennt, hintereinander auf dem Eingabeband befinden.

Auch weitaus kompliziertere Rechnungen stellen kein prinzipielles Problem dar, sondern sind allenfalls mit einer beträchtlichen Erhöhung der Zustandsanzahl verbunden, so zum Beispiel die Berechung einer bestimmten Ziffer von π.

TURINGs Überlegungen zeigten, daß die Beschränkung auf ein bestimmtes endliches Alphabet (wir hatten uns weiter oben auf das Alphabet $\{0, 1, \_\}$ beschränkt) nichts an den prinzipiellen Fähigkeiten von TURING-Maschinen ändert, die lediglich dieses sehr kleine Alphabet verwenden dürfen:

Für jede TURING-Maschine, die ein anderes Alphabet verwendet, läßt sich eine neue TURING-Maschine finden, die lediglich das Alphabet $\{0, 1, \_\}$ benutzt, aber prinzipiell das gleiche leistet.

Gegebenenfalls müssen dabei auch Input und Output in das neue Alphabet übersetzt werden. Die neue Maschine wird eventuell länger als die alte Maschine brauchen, aber immer, wenn die alte Maschine zu einem bestimmten Ergebnis gelangt ist, wird die neue Maschine zum gleichen Ergebnis kommen.

Wir wollen uns deshalb im Rahmen dieser Arbeit auf das obige Alphabet $\{0, 1, \_\}$ beschränken.

Wo allerdings stößt man auf die Grenzen der Berechenbarkeit durch TURING-Maschinen? Ist jede mathematische Rechnung, die wir intuitiv als „berechenbar“ auffassen würden, auch durch TURING-Maschinen berechenbar?

1.3 TURING-Berechenbarkeit

Eine der wichtigsten Erkenntnisse über TURING-Maschinen ist die Überlegung, daß wir jede beliebige TURING-Maschine M als eine spezielle einstellige Funktion f_M interpretieren können, die bestimmte natürliche Zahlen auf natürliche Zahlen abbildet.

Unsere Vereinbarung, nur endlich große Inputs auf dem Band zuzulassen, gestattet uns folgende Vorgehensweise:

(a)
Wir codieren den kompletten *(endlichen!)* Input, welcher sich zu Beginn der Berechnung auf dem Band befindet, durch eine natürliche Zahl. Auch bei mehreren Binärzahlen, die sich (durch Leerzeichen getrennt) als Eingabe auf dem Band befinden, ist dies problemlos möglich.

(b)
Ebenso codieren wir den kompletten Output der Maschine, falls sie den End-Zustand erreicht hat, durch eine natürliche Zahl.

(c)
Die Funktion f_M bildet nun natürliche Zahlen folgendermaßen auf natürliche Zahlen ab:

Falls die Maschine M bei Eingabe (Input) von i den End-Zustand mit dem Output j erreicht, gilt $f_M(i) = j$.

Hält eine TURING-Maschine M bei Eingabe einer bestimmten natürlichen Zahl nicht an, so gehört diese Zahl nicht zum Definitionsbereich der entsprechenden Funktion f_M. Jede TURING-Maschine bildet also eine Teilmenge der natürlichen Zahlen $\mathbb{N}$ in die natürlichen Zahlen ab.

Definition 1.1. (**TURING-Berechenbarkeit**)

Eine Funktion $f : \mathbb{N} \to \mathbb{N}$ *heißt* TURING-**berechenbar**, *wenn es eine* TURING-*Maschine gibt, die für alle Eingaben aus dem Definitionsbereich der Funktion den Funktionswert berechnet und bei allen anderen Eingaben nicht anhält.*

Eine weitere entscheidende Eigenschaft von TURING-Maschinen ist, daß man TURING-Maschinen „GÖDELisieren“[4] kann. Damit ist die Möglichkeit gemeint, eine eineindeutige Funktion anzugeben, die jeder TURING-Maschine eine natürliche Zahl zuordnet.

[4]nach KURT GÖDEL (1906-1978), österreichischer Mathematiker und Logiker. GÖDEL wird von vielen als der bedeutendste Logiker des 20. Jahrhunderts angesehen.

TURING-Maschinen sind durch ihr (endliches) Programm eindeutig beschrieben. Es stellt kein besonders schwieriges Problem dar, diese Programme eineindeutig durch natürliche Zahlen zu codieren.

Natürlich werden dabei jede Menge „Blindgänger“ entstehen – Programme, die überhaupt nichts berechnen und bei keiner einzigen Eingabe anhalten. So werden z.B. verschiedene TURING-Maschinen ungeachtet dessen, was sie auf dem Band vorfinden, stur in eine Richtung laufen; andere können nicht zu einem Ende kommen, weil niemals in den End-Zustand STOP gewechselt wird.

Das alles stellt aber kein Problem dar, weil wir ja ausdrücklich auch Teilmengen der natürlichen Zahlen (also auch die leere Menge $\emptyset$) als Definitionsbereiche der zu den TURING-Maschinen gehörenden Funktionen erlaubt haben.

Unsere Beschränkung auf TURING-Maschinen, die mit dem sehr kleinen Alphabet $\{0, 1, \_\}$ arbeiten, vereinfacht natürlich die Codierung der Programme.

Jede TURING-Maschine wird also durch eine spezielle natürliche Zahl eindeutig beschrieben. Ebenso gehört zu jeder natürlichen Zahl eindeutig eine bestimmte TURING-Maschine.

Da jede TURING-Maschine M außerdem als eine gewisse Funktion f_M interpretierbar ist, gelangen wir zu folgender Definition:

Definition 1.2. (M_i, D_i)

Die eindeutig bestimmte Funktion, die von der i-ten TURING*-Maschine berechnet wird, bezeichnen wir mit M_i, deren Definitionsbereich mit D_i, also:*

$M_i : D_i \to \mathbb{N}$ *mit* $(i \in \mathbb{N},\ D_i \subseteq \mathbb{N})$.

Die Schreibweise $M_i(j)$ umschreibt also den Funktionswert der Funktion M_i an der Stelle j, d.h. den Output der i-ten Maschine, wenn sie den Input j erhält. Man spricht hier auch von der Berechnung $M_i(j)$.

Nehmen wir also einmal an, die oben beschriebene Maschine NF habe die Nummer 110976, dann können wir schreiben:

$M_{110976}(x) = x + 1$ für alle $x \in \mathbb{N}$

Obwohl wir die Bezeichnung M_i für die Funktion verwenden, die durch die i-te TURING-Maschine berechnet wird, ist es üblich, auch die Maschine selbst mit M_i zu bezeichnen. Eine Begriffsvertauschung ist normalweise durch den Kontext nicht möglich, zumal TURING-Maschinen und die jeweiligen Funktionen, die sie berechnen, sich ohnehin entsprechen.

Fassen wir noch einmal zusammen:

Ohne die prinzipielle Funktionalität beliebiger TURING-Maschinen einzuschränken, können wir uns auf die Betrachtung spezieller TURING-Maschinen beschrän-

ken, die mit dem Alphabet $\{0, 1, \_\}$ arbeiten und die zu Beginn der Berechnung auf dem am weitesten links stehenden Zeichen des Inputs stehen. Jede solche TURING-Maschine kann als Funktion aufgefaßt werden, die natürliche Zahlen (den Input) aus ihrem Definitionsbereich auf natürliche Zahlen (den Output) abbildet. Außerdem können wir jeder dieser Maschinen bzw. Funktionen in eindeutiger Art und Weise eine natürliche Zahl zuordnen.

Die TURING-berechenbaren Funktionen sind also genau die Funktionen $M_0, M_1, M_2, M_3, ...$, denen exakt sämliche TURING-Maschinen entsprechen.

Wir definieren nun eine Menge, die in unseren weiteren Untersuchungen eine wichtige Rolle spielen wird:

Definition 1.3. (RE)

$\mathrm{RE} := \{D_i : i \in \mathbb{N}\}$

Was bedeutet diese Definition?

In der Menge RE befinden sich die Definitionsbereiche aller TURING-berechenbaren Funktionen, gewissermaßen die Definitionsbereiche aller TURING-Maschinen. Zu jeder dieser Mengen gibt es also eine TURING-Maschine, die bei Eingabe von Elementen der Menge anhält (also den STOP-Zustand erreicht) und bei allen restlichen Eingaben nicht anhält. Da jedes D_i eine Teilmenge der natürlichen Zahlen ist, handelt sich also um eine Teilmenge der Potenzmenge von $\mathbb{N}$.

Die Elemente von RE (engl. *R*ecursively *E*numerable) nennen wir *aufzählbare* Mengen.

Gebräuchlich ist auch folgende Sprechweise: Die TURING-Maschine M_i zählt die Menge D_i auf.

Gibt es überhaupt Teilmengen von $\mathbb{N}$, die nicht Elemente von RE sind? Der folgende Satz wird uns bei der Beantwortung dieser Frage helfen:

Satz 1.1. (Die Kardinalität von RE)

RE *und* $\mathbb{N}$ *sind gleichmächtig*[5].

Beweis:

Die Definition von RE *zeigt sofort, daß die Kardinalität von* RE *nicht größer als die Kardinalität der natürlichen Zahlen ist.*

Jeder einzelne Definitionsbereich gehört zu vielen verschiedenen TURING*-Maschinen. Zum Beispiel kann man für eine bestimmte* TURING*-Maschine den* STOP*-Zustand durch einen weiteren Zustand ersetzen, der den Output unverändert*

[5]Zwei Mengen sind gleichmächtig, wenn ihre Kardinalität (oder Mächtigkeit) übereinstimmt. Es gibt dann eine eineindeutige Abbildung der einen Menge auf die andere.

läßt und in jedem Fall als nächstes in den STOP*-Zustand wechselt. Diese neue* TURING*-Maschine hat natürlich den gleichen Defintionsbereich wie die alte, hat aber eine andere (vermutlich größere) Nummer.*

Deshalb ist es nötig zu zeigen, daß die Kardinalität von RE *nicht kleiner als die Kardinalität von* IN *ist.*

Für Mengen der Form $\{n\}$ *(also Mengen, die nur eine einzige natürliche Zahl enthalten) kann man trivial* TURING*-Maschinen angeben, deren Definitionsbereich* $\{n\}$ *ist:*

Die TURING*-Maschinen werden einfach in eine unendlich lang laufende Schleife gelenkt, es sei denn, die Zahl* n *liegt als Input vor. Dann lassen wir die Maschine in den* STOP*-Zustand übergehen. Die Menge* $\{n\}$ *ist also für jedes* $n \in \mathbb{N}$ *aufzählbar.*[6]

Auf diese Art und Weise erhalten wir so viele echt verschiedene Definitionsbereiche, wie es natürliche Zahlen gibt.

Somit sind RE *und* IN *gleichmächtig.* □

Gemäß der mächtigen und schönen Theorie der transfiniten Zahlen, die gegen Ende des 19. Jahrhunderts von GEORG CANTOR[7] entwickelt wurde, ist die Kardinalität von IN gleich der transfiniten Zahl mit der Bezeichnung $\aleph_0$ („Aleph null"). Man sagt in diesem Zusammenhang auch: IN ist eine *abzählbar unendliche* Menge.

Außerdem ergibt sich, daß die Kardinalität der Menge aller Teilmengen der natürlichen Zahlen, also der *Potenzmenge* von IN, echt größer als die Kardinalität von IN ist[8]. Mathematiker bezeichnen diese transfinite Zahl mit $\aleph_1$. Es gilt also $\aleph_0 < \aleph_1$.

Hat eine Menge die Kardinalität $\aleph_1$, spricht man von einer *überabzählbar unendlichen* Menge. Andere überabzählbar unendliche Mengen sind z.B. die *reellen Zahlen* oder auch die *reellen Zahlen zwischen* 0 *und* 1.

Die Menge der aufzählbaren Mengen RE ist also eine *abzählbar unendliche* Menge; die Potenzmenge von IN ist dagegen eine *überabzählbar unendliche* Menge.

Indem wir jetzt auf unsere ursprüngliche Frage zurückkommen, ob es überhaupt nicht-aufzählbare Teilmengen von IN gibt, stellen wir fest, daß die aufzählbaren Mengen nur eine abzählbar unendliche Teilmenge der Potenzmenge von IN bilden.

---

[6] Auf ähnliche Art kann man leicht zeigen, daß *jede* endliche Menge aufzählbar ist.

[7] GEORG CANTOR (1845-1918), deutscher Mathematiker. CANTOR gilt als Begründer der Mengenlehre und lieferte wichtige Beiträge zur modernen Mathematik.

[8] Es gilt sogar ganz allgemein, daß die Kardinalität der Potenzmenge einer Menge echt größer ist als die Kardinalität der Menge selbst.

1.4 Das Halteproblem

Wir wollen nun eine spezielle nicht aufzählbare Menge direkt angeben. Dazu führen wir zunächst eine wichtige verkürzende Schreibweise ein, die in der gesamten vorliegenden Arbeit Verwendung finden wird:

$\boldsymbol{M_i(j) = \perp}$ soll bedeuten: die i-te TURING-Maschine *hält* bei Eingabe von j *nicht an*, $\boldsymbol{M_i(j) \neq \perp}$ bedeutet: die i-te TURING-Maschine *hält* bei Eingabe von j *an*.

Betrachten wir nun die folgende Menge:

Definition 1.4. (Halteproblem)

$\overline{K} := \{n \in \mathbb{N} : M_n(n) = \perp\}$

$\overline{K}$ enthält die Nummern aller TURING-Maschinen, die bei Eingabe ihrer eigenen Nummer nicht anhalten.

Diese Menge ist nicht aufzählbar, wie der folgende Satz zeigt:

Satz 1.2. (Nichtaufzählbarkeit des Halteproblems)

$\overline{K} \notin \mathrm{RE}$

Beweis: *(indirekt)*

Nehmen wir $\overline{K} \in \mathrm{RE}$ *an. Demnach ist* $\overline{K}$ *der Definitionsbereich einer* TURING*-Maschine, sagen wir* M_k *(die Maschine habe also die Nummer* k*):*

$\overline{K} = D_k$

Es gilt also für beliebige $n \in \mathbb{N}$*:*

$n \in \overline{K} \Leftrightarrow M_k(n) \neq \perp$*, d.h. für* $n \in \overline{K}$ *hält die Berechnung* $M_k(n)$ *an.*

Außerdem gilt $n \in \overline{K} \Leftrightarrow M_n(n) = \perp$ *für alle* $n \in \mathbb{N}$ *nach der Definition von* $\overline{K}$*. Daraus folgt für beliebige* $n \in \mathbb{N}$*:*

$M_k(n) \neq \perp \Leftrightarrow M_n(n) = \perp$

Setzen wir $n = k$*:*

$M_k(k) \neq \perp \Leftrightarrow M_k(k) = \perp$

Das ist ein Widerspruch. Daraus folgt: $\overline{K} \notin \mathrm{RE}$ □

1.5 Church-Turing-These

Wie wir schon festgestellt haben, können Turing-Maschinen auch sehr schwierige mathematische Algorithmen ausführen. Je länger man sich mit den Fähigkeiten von Turing-Maschinen auseinandersetzt, desto überzeugender erscheint die Vermutung, daß derartige Maschinen im Prinzip *jede beliebige algorithmische Operation* ausführen können. Solange ein bestimmtes algorithmisches Rechenverfahren genügend klar umrissen ist, kann man vernünftigerweise annehmen, daß sich zu seiner Ausführung eine Turing-Maschine konstruieren läßt.

Im Zuge des von David Hilbert formulierten „Entscheidungsproblems“ beschäftigten sich außer Turing auch andere Wissenschaftler mit dem Versuch, den Begriff der Berechenbarkeit auf eine theoretische Basis zu stellen. Dabei entstanden vielfältige Modelle algorithmischen Arbeitens (Lambda-Kalkül, Markov-Algorithmen[9], partiell rekursive Funktionen etc.).

Verblüffenderweise gelangt man zu der Erkenntnis, daß die Mengen der Funktionen, die nach diesen Modellen jeweils berechenbar sind, sich präzise gleichen. Die verschiedenen Modelle zur Beschreibung von Algorithmen erweisen sich als vollkommen äquivalent.

Dadurch ist man allgemein zu der durchaus naheliegenden Ansicht gelangt, daß unsere intuitive Vorstellung von einem algorithmischen (oder mechanischen, effektiven oder rekursiven) Verfahren sich exakt durch den Begriff der Turing-Maschine definieren läßt. Dieser Standpunkt wurde als Church-Turing-These[10] bekannt. Alonzo Church hatte unabhängig von Alan Turing die Theorie des sogenannten Lambda-Kalküls entwickelt.

Heute, im Zeitalter immer schnellerer Computer mit immer größeren Speichereinheiten, wird kaum jemand diese These anzweifeln.

Unter Verwendung der Church-Turing-These werden wir uns nun endgültig der Lösung des ursprünglichen Hilbertschen Entscheidungsproblems nähern.

1.6 Das Hilbertsche Entscheidungsproblem

Turings Ideen waren ursprünglich zu dem Zweck entwickelt worden, das Hilbertsche „Entscheidungsproblem“ zu lösen. Zur Erinnerung:

Nicht weniger als ein allgemeines algorithmisches Verfahren zum Lösen mathematischer Probleme war gesucht.

---

[9] nach Andrei Markov (1903-1979), russischer Mathematiker und Logiker.

[10] Alonzo Church (1903-1995), amerikanischer Mathematiker, einer der Begründer der theoretischen Informatik.

TURING entwickelte die folgende Version der ursprünglichen Frage: Können wir jemals ein algorithmisches Verfahren vorlegen, welches für jede natürliche Zahl n entscheidet, ob die n-te TURING-Maschine bei Eingabe von n anhält? Die Antwort, die uns TURING geben konnte, lautet: Nein.

Versuchen wir zu verstehen, wie er zu diesem Ergebnis gekommen war:

Nehmen wir dazu an, es gebe das gesuchte Verfahren. Die CHURCH-TURING-These sagt aus, daß jedes algorithmische Verfahren durch eine TURING-Maschine beschreibbar ist. Es ist nun nicht mehr schwer einzusehen, daß es also eine TURING-Maschine geben muß, die entscheidet, ob die n-te TURING-Maschine bei Eingabe von n anhält. Wir wollen diese Maschine H nennen.

Es muß sich bei der Maschine H um eine sogenannte *universelle* TURING-*Maschine* handeln. Damit sind TURING-Maschinen gemeint, die imstande sind, andere TURING-Maschinen zu simulieren (imitieren).

Wir können nun annehmen, H liefert (bei Eingabe von n) auf dem Band die Zahl 1, wenn die n-te TURING-Maschine anhält, anderenfalls die Zahl 0.

Es gilt dann für alle $n \in \mathbb{N}$:

$$H(n) = \begin{cases} 0 & \text{falls } M_n(n) = \perp \\ 1 & \text{falls } M_n(n) \neq \perp \end{cases}$$

Es ist leicht, nun eine TURING-Maschine H^* zu konstruieren, die für alle $n \in \mathbb{N}$ folgendermaßen arbeitet:

$$H^*(n) = \begin{cases} 0 & \text{falls } M_n(n) = \perp \\ M_n(n) + 1 & \text{falls } M_n(n) \neq \perp \end{cases}$$

Dies ist möglich, weil wir im Falle von $M_n(n) \neq \perp$ einfach die Rechnung $M_n(n)$ von H^* imitieren lassen können, bevor wir zum Schluß noch 1 addieren.

Wie wir wissen, besitzt jede TURING-Maschine eine Nummer. Die Nummer von H^* sei k (wobei natürlich $k \in \mathbb{N}$). H^* ist also genau die Maschine M_k.

Wir setzen speziell $n = k$, es gilt nun:

$$H^*(n) = M_k(k) = \begin{cases} 0 & \text{falls } M_k(k) = \perp \\ M_k(k) + 1 & \text{falls } M_k(k) \neq \perp \end{cases}$$

Der erste Fall ($M_k(k) = \perp$) kann allerdings nie erfüllt sein, weil dann $M_k(k) = 0$ wäre, obwohl $M_k(k) = \perp$ gilt, was bedeutet, daß $M_k(k)$ niemals anhält. Aber auch der zweite Fall ($M_k(k) \neq \perp$) ist ausgeschlossen, weil dann $M_k(k) = M_k(k) + 1$ gelten würde.

Also kann es das Verfahren, dessen Existens wir zunächst angenommen haben, nicht geben. Kein algorithmisches Verfahren ist fähig zu entscheiden, ob die n-te

TURING-Maschine bei Eingabe von n anhält oder nicht. Es existiert also kein allgemeines algorithmisches Verfahren zum Lösen mathematischer Probleme. HILBERTs Fragestellung ist gelöst.

Kapitel 2

Berechenbarkeit des Denkens?

Die Tatsache, daß die im Abschnitt über das Halteproblem beschriebene Menge

$$\overline{K} := \{n \in \mathbb{N} : M_n(n) = \bot\}$$

nicht aufzählbar ist, konnte das HILBERTsche Entscheidungsproblem lösen. Das Halteproblem wird uns in leicht abgewandelter Form auch weiterhin beschäftigen.

Wir wollen versuchen, eine Leistung des menschlichen Denkens zu beschreiben, zu der kein Algorithmus (also keine TURING-Maschine) fähig ist.

Insbesondere wird uns die menschliche Fähigkeit zum korrekten mathematischen Schlußfolgern interessieren.

2.1 Ein paar Vorbereitungen

Zunächst definieren wir noch eine weitere wichtige Menge.

Definition 2.1. (REC)

$\mathrm{REC} := \{\mathrm{M} \subseteq \mathbb{N} : \mathrm{M} \in \mathrm{RE} \wedge \overline{\mathrm{M}} \in \mathrm{RE}\}$

Dabei bezeichnet $\overline{M}$ das Komplement von M, also $\mathbb{N} \backslash M$. Sie enthält diejenigen natürlichen Zahlen, die nicht in M enthalten sind.

Die Elemente von REC werden *entscheidbare* Mengen genannt. Die Bezeichnung ist durchaus suggestiv gewählt.

Der Unterschied zu den Elementen von RE, den aufzählbaren Mengen, besteht in folgendem:

Eine Menge $M \in \mathrm{REC}$ muß zwei Eigenschaften erfüllen:

(1) $M \in \mathrm{RE}$, d.h. es gibt eine TURING-Maschine, sagen wir M_i, die M aufzählt, es gilt also für beliebige $n \in \mathbb{N}$:

$M_i(n) \neq \perp \Leftrightarrow n \in M$

(2) $\overline{M} \in \mathrm{RE}$, d.h. es gibt eine andere TURING-Maschine, sagen wir M_j, die $\overline{M}$ aufzählt. Hier gilt für alle $n \in \mathbb{N}$:

$M_j(n) \neq \perp \Leftrightarrow n \notin M$

Man kann nun eine dritte TURING-Maschine, sagen wir M_k, konstruieren, die für jede beliebige Eingabe n abwechselnd jeweils einen Berechnungsschritt von M_i und M_j simuliert. Die Maschine soll die Simulation beenden, wenn eine der beiden Maschinen anhält. Bevor sie endgültig anhält, soll sie nun noch eine Ausgabe nach folgendem Muster produzieren:

$$M_k(n) = \begin{cases} 1 & \text{falls } n \in M \\ 0 & \text{falls } n \notin M \end{cases}$$

Die Ausgabe der Maschine bei einer Eingabe von n teilt uns also mit, ob $n \in M$ oder $n \notin M$ gilt. In diesem Sinne ist die Bezeichnung *entscheidbare* Menge zu verstehen. Man spricht auch davon, daß die Maschine M_k die Menge M *entscheidet.*

Daß es aufzählbare, aber nicht entscheidbare Mengen gibt, zeigt folgender Hilfssatz:

Hilfssatz 2.1. $\mathrm{REC} \subset \mathrm{RE}$

Beweis:

$\mathrm{REC} \subseteq \mathrm{RE}$ *folgt sofort aus der Definition von* REC.

Es genügt eine aufzählbare, nicht entscheidbare Menge anzugeben:

$K := \{n \in \mathbb{N} : M_n(n) \neq \perp\}$

Diese Menge ist das Komplement zur Menge $\overline{K}$ *(siehe Seite 16) und enthält die Nummern aller* TURING*-Maschinen, die bei Eingabe ihrer eigenen Nummer anhalten. Zum Beweis von* $K \in \mathrm{RE}$ *konstruieren wir trivial eine* TURING*-Maschine, die bei Eingabe von* n *exakt* $M_n(n)$ *simuliert. Diese Maschine hält genau dann an, wenn* $M_n(n)$ *anhält.*

Da $\overline{K}$ *nicht aufzählbar ist (siehe Satz 1.2 auf Seite 16), ist* K *nicht entscheidbar.* □

Auch die Menge REC ist eine unendliche Menge, da die im Beweis von Satz 1.1 erwähnten endlichen Mengen, die nur eine einzige natürliche Zahl enthalten, entscheidbare Mengen sind. Wegen $\mathrm{REC} \subset \mathrm{RE}$ sind REC und RE somit gleichmächtig.

2.2 Menschliches Denken

In der Definition 1.4 (Seite 16) wurde die Menge $\overline{K}$ (Halteproblem) beschrieben. Die Menge $\overline{K}$ enthält die Nummern aller Maschinen, die bei Eingabe ihrer eigenen Nummer nicht anhalten. Diese Menge ist nicht aufzählbar. Dennoch kann man für viele natürliche Zahlen schlüssig und korrekt nachweisen, daß sie zur Menge $\overline{K}$ gehören.

Zum Beispiel gibt es viele Maschinen, die bei keiner einzigen Eingabe anhalten, weil ihr gesamter Befehlsaufbau keine vernünftige Berechnung zuläßt.[1] Ebenso kann man sicher bei vielen Maschinen leicht zeigen, daß sie unabhängig von der Eingabe nach einer gewissen Zeit in eine unendliche Schleife geraten:

S	0	1	_
z_1	$0 \rightarrow z_2$	$1 \rightarrow z_2$	$\_ \rightarrow z_2$
z_2	$0 \leftarrow z_1$	$1 \leftarrow z_1$	$\_ \leftarrow z_1$
z_3	STOP	STOP	STOP

Diese Maschine S läuft im ersten Zustand (unabhängig von der Eingabe) immer einen Schritt nach rechts, um dann in den zweiten Zustand zu wechseln. Dort läuft sie einen Schritt nach links und wechselt wieder in den ersten Zustand.

Diese Liste kann man beliebig fortsetzen.

Man könnte sogar meinen, wenn man eine vorgegebene Maschine hinreichend lange untersucht, wird man immer verstehen, wie sie genau arbeitet, und dementsprechend beweisen können, wenn sie tatsächlich bei Eingabe ihrer eigenen Nummer nicht anhält.

Diese Behauptung ist nicht haltbar, da man relativ leicht TURING-Maschinen angeben kann, die ein bestimmtes mathematisches (sagen wir zahlentheoretisches) Problem lösen. So kann man z.B. recht einfach eine Maschine konstruieren, die (*unabhängig* von der Eingabe) genau dann anhält, wenn die GOLDBACHsche[2] Vermutung[3] falsch ist. Könnte man nun beweisen oder wiederlegen, daß diese Maschine bei Eingabe ihrer eigenen Nummer anhält, hätte man damit einen Gegenbeweis oder Beweis der GOLDBACHschen Vermutung. Es ist also nicht in jedem Falle möglich zu entscheiden, ob eine natürliche Zahl zur Menge RE gehört.

---

[1] Diese sogenannten Blindgänger können unter anderem entstehen, wenn in keinem der Programmschritte in den STOP-Zustand gewechselt wird oder wenn das Programm insgesamt unvollständig ist.

[2] CHRISTIAN GOLDBACH (1690-1764), deutscher Mathematiker.

[3] Die GOLDBACHsche Vermutung ist eine der ältesten offenen Fragen der Zahlentheorie und der Mathematik überhaupt: *Jede gerade Zahl größer oder gleich 4 ist als Summe zweier Primzahlen darstellbar.* Die GOLDBACHsche Vermutung ist bisher weder bewiesen noch widerlegt worden. Die meisten Mathematiker halten die Vermutung aus statistischen Gründen für wahr.

Wichtig soll hier erst einmal nur sein, daß man für bestimmte TURING-Maschinen beweisen kann, daß sie bei Eingabe ihrer eigenen Nummer nicht anhalten. Wir können also für bestimmte natürliche Zahlen zeigen, daß sie zur Menge $\overline{K}$ gehören. Wir wollen versuchen, all diese Zahlen in einer Menge zusammenzufassen.

Zunächst wollen wir definieren, was wir unter der Beweisbarkeit von mathematischen Aussagen verstehen wollen.

Sei $\mathbb{A}$ die Menge der mathematischen Aussagen[4]. Wir führen nun folgende Funktion $\boldsymbol{Beweis} : \mathbb{A} \to \{wahr, falsch\}$ ein:

Definition 2.2. (menschliche Beweise)

$$\boldsymbol{Beweis}(A) := \begin{cases} wahr, & A \text{ ist wahr und menschliche Mathematiker sind} \\ & \text{prinzipiell in der Lage, dies zu beweisen} \\ falsch, & \text{sonst.} \end{cases}$$

Die Beschränkung auf menschliche *Mathematiker* ist nicht als echte Beschränkung zu verstehen, ebensogut sind Beweise von Nicht-Mathematikern zugelassen, die prinzipiell von Mathematikern verstanden werden können.

Die Formulierung „...menschliche Mathematiker sind *prinzipiell* in der Lage, ...“ soll bedeuten, daß ein korrekter (menschlicher) Beweis existiert oder nach endlicher Zeit gefunden wird.

Man kann sich $\boldsymbol{Beweis}(A)$ einfach vorstellen als:
„Es existiert ein Beweis für A, der durch korrektes menschliches Schlußfolgern gefunden werden kann.“

Sicher ist die Definition keine mathematische Definition im strengen Sinne. Wir wollen für die weiteren Betrachtungen aber davon ausgehen, daß sie in dieser Form sinnvoll ist, d.h. eine exakte Definition für eine bestimmte Funktion $\boldsymbol{Beweis} : \mathbb{A} \to \{wahr, falsch\}$ liefert, die mathematischen Aussagen den Wert „wahr“ oder „falsch“ zuordnet, je nachdem, ob die entsprechende Aussage von menschlichen Mathematikern prinzipiell bewiesen werden kann oder nicht.[5]

---

[4]Die Menge der mathematischen Aussagen ist natürlich nicht hinreichend definiert. Wichtig ist in diesem Zusammenhang jedoch nur, daß jede dieser mathematischen Aussagen einen Wahrheitswert (wahr oder falsch) hat, auch wenn wir diesen Wahrheitswert unter Umständen nicht kennen.

[5]Wenn man die prinzipielle Existenz korrekter mathematischer Beweise nicht in Frage stellt, wird man sicher auch nicht an der Korrektheit der Definition zweifeln. Im Grunde werden alle wahren Aussagen A, zu denen ein korrekter mathematischer Beweis existiert oder in der Zukunft gefunden wird, aufgesammelt und es gilt dann $\boldsymbol{Beweis}(A) =$ wahr.

Es bleibt zu erwähnen, daß der Wahrheitswert $\boldsymbol{Beweis}(A)$ von mathematischen Aussagen A zwar immer existiert, aber nicht in jedem Fall von menschlichen Mathematikern angegeben werden kann.

Falls die Aussage A falsch ist, gilt natürlich $\boldsymbol{Beweis}(A) =$ falsch, falls es einen *menschlichen* Beweis für A gibt, gilt $\boldsymbol{Beweis}(A) =$ wahr. In anderen Fällen ist der Wahrheitswert von $\boldsymbol{Beweis}(A)$ unter Umständen unbekannt[6]. Zum Beispiel könnte die oben beschriebene GOLDBACHsche Vermutung wahr sein, aber wir können es nicht beweisen.

Zwei einfache Schlußfolgerungen aus der Definition sind:

$$\boldsymbol{Beweis}(A) \Rightarrow A \tag{1}$$
$$[\boldsymbol{Beweis}(A) \wedge \boldsymbol{Beweis}(A \Rightarrow B)] \Rightarrow \boldsymbol{Beweis}(B) \tag{2}$$

In der Aussage (1) verstehen wir A als „Die Aussage A ist wahr.“ Damit gilt die Aussage (1) trivial, weil eine Aussage natürlich wahr sein muß, wenn ein Beweis für sie existiert.

Die Aussage (2) gilt, weil man an den Beweis für A einfach den Beweis für $A \Rightarrow B$ anhängen kann und somit einen Beweis für B erhält.

Wir definieren eine Menge, die uns im weiteren ständig beschäftigen wird:

Definition 2.3. ($\boldsymbol{D}$)
$D := \{n \in \mathbb{N} : \boldsymbol{Beweis}(M_n(n) = \perp)\}$

> In ROGER PENROSE's Buch „Schatten des Geistes“ [3] wird diese Menge in ähnlicher Form definiert. Dort heißt es:
>
> „(...) Nehmen wir also an, wir hätten ein Rechenverfahren A, das uns dann, wenn es erfolgreich zum Ende kommt, beweist, daß eine Berechnung wie $C(n)$ in der Tat niemals aufhört. Wir versuchen uns vorzustellen, daß A alle Verfahren enthält, mit deren Hilfe menschliche Mathematiker überzeugend beweisen können, daß Berechnungen nicht anhalten. Wenn A in einem bestimmten Fall je zu einem Ende kommt, hätten wir also einen Beweis dafür, daß diejenige Berechung, auf die A sich bezieht, niemals anhält. (...)“

[6] KURT GÖDEL bewies im sogenannten GÖDELschen Unvollständigkeitssatz: In jeder denkbaren Theorie der natürlichen Zahlen bleiben wahre arithmetische Aussagen übrig, für die man weder beweisen kann, ob sie wahr sind, noch, ob sie falsch sind. Durch diesen erstaunlichen Satz ist der Mathematik eine prinzipielle Grenze gesetzt: Nicht für jedes mathematische Problem gibt es auch eine mathematische Lösung!

> Dabei ist mit $C(n)$ nicht die Berechnung $M_n(n)$ gemeint, sondern noch etwas allgemeiner irgendeine Berechnung C mit Eingabe n. Alle Berechnungen C werden später durchnumeriert und entsprechen dann (mit einer etwas anderen Codierung) den TURING-Maschinen M_i, die in dieser Arbeit betrachtet werden.
>
> Penrose schreibt weiter:
>
> „(...) Das Verfahren A kann man sich jetzt als eine Berechnung vorstellen, die dann, wenn ihr das Zahlenpaar q, n vorgelegt wird, versucht zu zeigen, daß die Berechnung $C_q(n)$ niemals aufhören wird. Wenn die Berechnung A also *aufhört*, ist damit bewiesen, daß $C_q(n)$ *nicht anhält*. Für unsere Zwecke werden wir uns in Kürze vorzustellen versuchen, daß A eine Formalisierung *aller* Verfahren ist, mit denen menschliche Mathematiker gültig zu entscheiden vermögen, daß Berechnungen niemals aufhören. (...) Die Berechnung, die A durchführt, läßt sich, da sie von den beiden Zahlen q und n abhängt, als $A(q, n)$ schreiben, und wir haben:
>
> Wenn $A(q, n)$ anhält, hält $C_q(n)$ nicht an.
>
> Betrachten wir jetzt speziell die Aussagen, für die q gleich n gesetzt wird. (...)“
>
> Das Rechenverfahren A kann dann als die oben definierte Menge D verstanden werden, wenn man den Definitionsbereich von A betrachtet und zusätzlich nur diejenigen Berechnungen betrachtet, in denen $n = q$ gilt.

Trivial gilt die Inklusionsbeziehung

$D \subseteq \overline{K}$,

wobei $\overline{K}$ die in Definition 1.4 (Seite 16) definierte Menge ist. Wegen der Eigenschaft (1) auf Seite 24 muß jedes Element von D auch Element von $\overline{K}$ sein.

Diese Menge ist, etwas locker formuliert, in einem *menschlichen* Sinne aufzählbar. Das soll bedeuten:

Für alle $n \in D$ wird die Tatsache „n gehört zur Menge D“ nach endlicher Zeit von menschlichen Mathematikern festgestellt (bewiesen) werden.

Begründung:
Denn für alle $n \in D$ gilt (nach Definition 2.2): Menschliche Mathematiker sind prinzipiell in der Lage, $n \in \overline{K}$ zu beweisen. Das heißt, ein Beweis für $n \in \overline{K}$ existiert bereits oder wird nach endlicher Zeit gefunden.

Für gewisse $n \notin D$ kann auch die Tatsache „n gehört nicht zur Menge D“ nach endlicher Zeit festgestellt werden. Wenn die Berechnung $M_n(n)$ anhält, gilt

natürlich $n \notin D$. Dies kann in endlicher Zeit bewiesen werden, indem man einfach die Berechnung $M_n(n)$ ablaufen läßt.

Übrig bleiben die Fälle, in denen menschliche Mathematiker nach endlicher Zeit keinen Beweis für $n \in \overline{K}$ finden können, die Berechnung $M_n(n)$ aber auch nicht anhält.

2.3 Aufzählbarkeit von D

Im Satz 1.2 (Seite 16) wurde $\overline{K} \notin \mathrm{RE}$ gezeigt. Wenn man nun $D \notin \mathrm{RE}$ zeigen könnte, hätte man eine Leistung des menschliches Denkens gefunden, die von keiner TURING-Maschine nachvollzogen werden kann. Dabei beziehen wir uns auf die oben erwähnte Eigenschaft, die Menge D sei *menschlich* aufzählbar, während $D \notin \mathrm{RE}$ bedeuten würde, daß D nicht TURING-aufzählbar ist. Wir werden später sehen, daß die doch recht lockere Interpretation durchaus echten wissenschaftlichen Gehalt hat.

Leider kann $D \notin \mathrm{RE}$ nicht in dieser allgemeinen Form bewiesen werden[7], allerdings führt ein etwas schwächerer Ansatz zu folgendem Satz:

Satz 2.1. (Aufzählbarkeit von D?)

Für kein $k \in \mathbb{N}$ gilt $\boldsymbol{Beweis}(D_k = D)$[8]

Beweis: *(indirekt)*

Zunächst eine kurze, aber durchaus wichtige Vorbemerkung:

Hierbei ergibt die Aussage $\boldsymbol{Beweis}(D_k = D)$ *natürlich nur einen Sinn, wenn uns (d.h. uns als menschlichen Mathematikern) die Zahl $k \in \mathbb{N}$ direkt bekannt ist, also gewissermaßen eine spezielle natürliche Zahl ist.*

Es ist in der Tat so, daß

$\boldsymbol{Beweis}(D_k = D)$

eine wesentlich stärkere Aussage als

$\boldsymbol{Beweis}(\exists k \in \mathbb{N}(D_k = D))$

darstellt. Die Aussage $\boldsymbol{Beweis}(\exists k \in \mathbb{N}(D_k = D))$ *würde bedeuten, daß wir einen Beweis dafür haben, daß eine* beliebige *(uns unter Umständen völlig unbekannte) Zahl die Aussage $D_k = D$ erfüllt.*

---

[7] Ein zunächst vielversprechender, aber erfolgloser Versuch, $D \notin \mathrm{RE}$ zu beweisen, wird im Versuch eines Satzes 2.1 (Seite 32) besprochen.

[8] Die Behauptung des Satzes kann wie folgt umformuliert werden:
$\{k \in \mathbb{N} : \boldsymbol{Beweis}(D_k = D)\} = \emptyset$

Die Zahl k aus der Behauptung des Satzes steht also tatsächlich nur als Platzhalter für die (unendlich vielen) verschiedenen Sätze
$\boldsymbol{Beweis}(D_0 = D) = falsch,$
$\boldsymbol{Beweis}(D_1 = D) = falsch,$
$\boldsymbol{Beweis}(D_2 = D) = falsch,$
$\vdots$

Nehmen wir also an, es gelte $\boldsymbol{Beweis}(D_k = D)$ *für ein spezielles k. Damit gilt:*
$\boldsymbol{Beweis}(\forall n \in \mathbb{N}(n \in D_k \Leftrightarrow n \in D))$.
Wir setzen $n = k$*, damit gilt*
$\boldsymbol{Beweis}(k \in D_k \Leftrightarrow k \in D)$.
Weil $k \in D_k$ *äquivalent zu* $M_k(k) \neq \perp$ *ist, folgt:*
(a) $\boldsymbol{Beweis}(M_k(k) \neq \perp \Leftrightarrow k \in D)$
Weil außerdem $k \in D$ *äquivalent zu* $\boldsymbol{Beweis}(M_k(k) = \perp)$ *gilt:*
(b) $\boldsymbol{Beweis}(M_k(k) \neq \perp \Leftrightarrow \boldsymbol{Beweis}(M_k(k) = \perp))$
Aus (a) folgt (unter Verwendung von Schlußfolgerung (1), Seite 24)
$M_k(k) \neq \perp \Leftrightarrow k \in D$ $(*)$
Aus (b) folgt (ebenfalls unter Verwendung von Schlußfolgerung (1), Seite 24)
$\boldsymbol{Beweis}(M_k(k) \neq \perp \Rightarrow M_k(k) = \perp)$
und damit
$\boldsymbol{Beweis}(M_k(k) = \perp)$[9].
Hieraus lassen sich zwei Aussagen ableiten:
(c) (mit der Definition von D): $k \in D$,
(d) (mit Schlußfolgerung (1), Seite 24): $M_k(k) = \perp$ $(**)$
Aus (c) und (*) folgt $M_k(k) \neq \perp$*, im Widerspruch zu (**).*
Damit ist der Satz bewiesen. □

> In Roger Penrose's Buch [3] wird dieser Beweis in einer ganz ähnlichen Form geführt. Dabei muß allerdings angemerkt werden, daß der Penrosesche Beweis in einer sehr prosaischen Form dargeboten wird und zudem einer wohlwollenden Auslegung bedarf, da verschiedene wichtige Beweisbestandteile nicht explizit erwähnt worden sind.

[9] Dies folgt aus der einfachen logischen Tautologie $(A \Rightarrow \neg A) \Rightarrow \neg A$.

So wird zum Beipiel in der oben aufgeführten „Definition“ des Verfahrens $A_q(n)$ davon ausgegangen, daß uns menschlichen Mathematikern die Nummer eines solchen Verfahrens (nachdem q gleich n gesetzt wurde) direkt vorliegt. Außerdem verlangt PENROSE von seinem Rechenverfahren nicht nur die Eigenschaft:

„Wenn $A(q, n)$ anhält, so hält $C_q(n)$ nicht an“,

sondern in der Interpretation des Prosaischen „Beweises“ auch deren Umkehrung

„Wenn $C_q(n)$ nicht anhält, so hält $A(q, n)$ an“,

obwohl dies an keiner einzigen Stelle vorher erwähnt worden ist.

So gelangt PENROSE schließlich zu folgendem Schluß:

„(...) Wenn wir A und seine Korrektheit kennen, läßt sich eine Berechnung $C_k(k)$ konstruieren, von der wir sehen können, daß sie niemals aufhört; deshalb kann A keine Formalisierung der Beweisverfahren sein, die Mathematikern zur Verfügung stehen, um — unabhängig von der Beschaffenheit von A — zu beweisen, daß Berechnungen nicht aufhören. Es gilt also:

Menschliche Mathematiker verwenden zum Nachweis mathematischer Wahrheit keinen nachweislich korrekten Algorithmus. (...)“

Dabei ist mit der Korrektheit von A die Eigenschaft von A gemeint, daß, wenn $A(q, n)$ anhält, damit ein Beweis für „$C_q(n)$ hält nicht an.“ vorliegt. Außerdem steckt in dem kleinen Wort „nachweislich“ bei PENROSE nicht nur die Eigenschaft, daß die Korrektheit von A bewiesen worden ist, sondern auch die genaue Kenntnis der Nummer eines solchen Verfahrens (also Algorithmus) A.

Alles in allem stellt der obige Satz 2.1 eine ziemlich genaue, aber mathematisch wesentlich strengere Formulierung der PENROSEschen Gedanken dar.

Die Ausführungen von PENROSE in der Version seines Buches [3] würden einer strengen mathematischen Prüfung nicht standhalten.

2.4 Was haben wir erreicht?

Versuchen wir uns klarzumachen, was wir überhaupt bewiesen haben.

Wir haben zunächst die Menge D definiert. Diese Menge beschreibt *menschliche* Fähigkeiten, gewisse mathematische Beweise (diese Beweise betreffen TURING-Maschinen) zu führen. Sie soll in einem gewissen Sinne *menschlich* aufzählbar verstanden werden.

Dann haben wir $\{k \in \mathbb{N} : \boldsymbol{Beweis}(D_k = D)\} = \emptyset$ bewiesen, d.h. daß für kein spezielles $k \in \mathbb{N}$ die Aussage

$\boldsymbol{Beweis}(D_k = D)$

gilt. Ein Beweis für $D_k = D$ kann von menschlichen Mathematikern für kein spezielles $k \in \mathbb{N}$ geführt werden.

Dafür kann es mehrere Gründe geben:

(1)
Das kann zum Beispiel daran liegen, daß ganz allgemein $D \notin \mathrm{RE}$ gilt.
Das würde bedeuten, es gibt tatsächlich keine Maschine, die D aufzählt. Die Behauptung von Satz 2.1 gilt dann trivial: Man kann $D_k = D$ natürlich für kein $k \in \mathbb{N}$ zeigen, denn dann würde ja $D \in \mathrm{RE}$ gelten.

Menschliche Beweise für $M_k(k) = \perp$ wären dann nicht komplett von Computern nachvollziehbar. Der Glaube an diese Variante läßt vermuten, daß man geneigt ist, die menschlichen Denkfähigkeiten grundsätzlichen über die (Denk-)Fähigkeiten von Maschinen zu stellen.

(2)
Im Gegensatz dazu kann aber durchaus $D \in \mathrm{RE}$ gelten.
Es gibt dann eine Maschine M_k, die D aufzählt, nur sind menschliche Mathematiker nach obigem Beweis nicht imstande, für dieses $k \in \mathbb{N}$ einen Beweis für $D_k = D$ anzugeben.

Im Grunde läßt sich diese Tatsache auch mit gesundem Menschenverstand gut vereinbaren. Hätten wir einen Beweis, daß eine ganz bestimmte *(uns bekannte)* natürliche Zahl k die Eigenschaft $D_k = D$ erfüllt, dann hätten wir eine exakte mathematische Beschreibung der Menge D gefunden. Dies ist wirklich schwer vorstellbar, da wir dann eine exakte mathematische Beschreibung der *prinzipiellen* menschlichen Fähigkeit haben, gewisse mathematische Beweise zu führen. Wir hätten also die Möglichkeit, alle natürlichen Zahlen k gewissermaßen im Voraus zu kennen, zu denen irgendwann menschlichen Beweise für $M_k(k) = \perp$ gefunden werden.

Welche der beiden Varianten (1) oder (2) man favorisiert, bleibt natürlich jedem selbst überlassen.

2.5 Ein wenig Philosophie

Auf weitere interessante Tatsachen soll hier noch einmal speziell hingewiesen werden:

(a)
$D \in \mathrm{RE}$ bedeutet nicht sofort, daß wir eines Tages mit *menschlich denkenden* Maschinen konkurrieren müssen.

Zum einen zeigt Satz 2.1, daß Menschen grundsätzlich nicht in der Lage sind, ein $k \in \mathbb{N}$ und einen Beweis für $D_k = D$ anzugeben. Wir könnten die entsprechende Maschine M_k also nur *sozusagen zufällig* finden, ohne einen Beweis für $D_k = D$ zu haben.

Zum anderen bedeutet selbst das Vorhandensein der entsprechenden Maschine M_k mit $D_k = D$ nicht automatisch, daß diese Maschine unser menschliches Denken perfekt simulieren kann. Wir haben uns ja bisher nur auf einen sehr kleinen Teil unserer menschlichen Denkfähigkeiten beschränkt, nämlich auf die Fähigkeit, Beweise für ganz bestimmte mathematische Sachverhalte anzugeben. Alle anderen menschlichen Denkleistungen bleiben von diesen Überlegungen unberührt.

Allein die Versuche, die menschliche Fähigkeit zur sprachlichen Äußerung (und seien es auch nur sprachliche Äußerungen in schriftlicher Form) mit Computern nachzuvollziehen, sind bisher von sehr geringem Erfolg gekennzeichnet. Es gibt sehr gute Algorithmen der Spracherkennung, wenn es darum geht, aus einem akustischen Signal die entsprechenden Wörter zu erkennen, es gibt auch ganz brauchbare Algorithmen, um diesen Sequenzen eine grammatische Struktur zuzuordnen. Aber aus diesen Strukturen dann die logisch transparente Bedeutungsrepräsentation abzuleiten, ist bisher kaum gelungen; ganz zu schweigen von verschiedenen anderen kommunikativen und sozialen *menschlichen* Fähigkeiten.

(b)
Im Gegensatz dazu schließt $D \notin \mathrm{RE}$ natürlich nicht aus, daß es eines Tages Maschinen gibt, die menschliches Denken nahezu perfekt simulieren können.

Die Tatsache, daß menschliche Mathematiker unter Kenntnis der Nummer k der Maschine und mit Hilfe des Beweises für $D_k = D$ eine ganz bestimmte mathematische Aussage beweisen können (nämlich $\boldsymbol{Beweis}(M_k(k) = \perp)$), die die Maschine nicht nachvollziehen kann, stellt eine *zutiefst theoretische* Aussage dar. Sie gibt kaum Auskunft darüber, in welcher Art und Weise mögliche zukünftige Maschinen menschliches Denken simulieren könnten.

(c)
Die weiter oben erwähnte lockere Formulierung „Die Menge D ist *menschlich* aufzählbar“, während die Menge D nur dann *im klassischen Sinne* aufzählbar ist, wenn menschliche Mathematiker für kein $k \in \mathbb{N}$ die Eigenschaft $D_k = D$ be-

weisen können, findet im Beweis von Satz 2.1 eine stärkere, eher *wissenschaftliche* Bedeutung.

Falls nämlich $\boldsymbol{Beweis}(D_k = D)$ für ein spezielles $k \in \mathbb{N}$ gilt, so folgt mit Hilfe des Beweises $\boldsymbol{Beweis}(M_k(k) = \perp)$. Wir *(d.h. wir Menschen)* hätten dann also einen Beweis für $M_k(k) = \perp$; während, wie der weitere Verlauf des Beweises zeigt, die Maschine M_k diesen Beweis nicht führen kann. Wir sind also in Bezug auf die Aufzählbarkeit der Menge D jeder Maschine M_k überlegen, deren Nummer k wir kennen. In abgeschwächter Form könnte man also folgende Interpretation wagen:

Menschliches Denken ist nicht durch von Menschen gebaute Maschinen berechenbar.

(Denn wir können die Nummer jeder von Menschen gebauten Maschine angeben.)

(d)
Der Satz 2.1 behauptet nicht, daß ganz allgemein niemals $D \notin \mathrm{RE}$ oder $D \in \mathrm{RE}$ gezeigt werden kann.

Wie oben bereits formuliert, kann natürlich $D \notin \mathrm{RE}$ gelten, und auch ein *menschlicher* Beweis dafür ist nicht auszuschließen.

Ebenso könnte auch für $D \in \mathrm{RE}$ ein Beweis gefunden werden! Es ist allerdings unmöglich, daß dieser Beweis die Möglichkeit beinhaltet, konstruktiv eines der $k \in \mathbb{N}$ anzugeben, für das $D_k = D$ gilt.

Populärwissenschaftlich formuliert:
Die Maschine M_k, die D aufzählt (das heißt $D_k = D$), könnte vor uns stehen, aber wir wissen es nicht oder können es nicht beweisen.

(e)
Wir sprechen hier nach wie vor natürlich nur von Computerprogrammen, die Turing-berechenbare Funktionen berechnen. Dies schließt nicht aus, daß eines Tages ein Computer entwickelt wird, der in seinen Fähigkeiten über die einer Turing-Maschine hinausgeht. Für solche Computer sind die obigen Überlegungen unter Umständen nicht mehr anwendbar.

Eine solche Entwicklung ist im Moment allerdings noch nicht abzusehen.

2.6 Der Versuch eines schärferen Beweises

Wenn man sich den Satz 2.1 (Seite 26) und den dazugehörigen Beweis genau anschaut, könnte man auch den nun folgenden Versuch eines schärferen Beweises wagen. Wir wollen zeigen, an welcher Stelle der schärfere Beweis scheitert:

Versuch eines Satzes 2.1.

$D \notin \mathrm{RE}$

Beweis: *(indirekt)*

Nehmen wir also an, es gelte $D \in \mathrm{RE}$.

Dann existiert ein $k \in \mathbb{N}$ *derart, daß für alle* $n \in \mathbb{N}$

$M_k(n) \neq \perp \Leftrightarrow n \in D$ $\quad (*)$

gilt. Mit $n = k$ *gilt*

$M_k(k) \neq \perp \Leftrightarrow k \in D$

Nach der Definition für D *gilt weiter:* $k \in D \Rightarrow M_k(k) = \perp$, *also:*

$M_k(k) \neq \perp \Rightarrow M_k(k) = \perp$, *woraus*

$M_k(k) = \perp$ $\quad (**)$

folgt[10].

Da diese Tatsache soeben durch menschliche Mathematiker (wir selbst und der verstehende Leser) herausgefunden wurde, liegt also offensichtlich ein Beweis für $M_k(k) = \perp$ *vor, also gilt* $\boldsymbol{Beweis}(M_k(k) = \perp)$, *und damit*

$k \in D$ $\quad (F)$

Mit $(*)$ *folgt* $M_k(k) \neq \perp$, *im Widerspruch zu* $(**)$.

Es ist in der Tat nicht einfach, den Fehler im Beweis zu finden. Er findet sich in der Begründung, mit deren Hilfe der Schritt von $(*)$ nach (F) vollzogen wird.

Üblicherweise werden in Beweisen gewisse Beweisschritte verkürzend formuliert, um die Übersichtlichkeit zu erhöhen. In diesem Fall führt uns echte mathematische Exaktheit den Fehler im Beweis vor Augen. Exakt betrachtet wurde bis zum Schritt $(**)$ Folgendes gezeigt:

$D \in \mathrm{RE} \Rightarrow \exists \mathrm{k} \in \mathbb{N}(\mathrm{M_k(k)} = \perp)$ $\quad (**')$

Wenn wir nun daraus

$D \in \mathrm{RE} \Rightarrow \exists \mathrm{k} \in \mathbb{N}(\boldsymbol{Beweis}(\mathrm{M_k(k)} = \perp))$ $\quad (F')$

[10] Dies folgt wieder aus $(A \Rightarrow \neg A) \Rightarrow \neg A$.

folgern könnten, hätten wir

$D \in \mathrm{RE} \Rightarrow \exists \mathrm{k} \in \mathbb{N}(\mathrm{k} \in \mathrm{D})$

und mit $(*)$[11] auch

$D \in \mathrm{RE} \Rightarrow \mathrm{M_k(k)} \neq \perp$.

Zusammen mit $(**')$ würde dann wie gewünscht $D \notin \mathrm{RE}$ folgen.

Von $(**')$ nach (F') tritt nun folgender Fehler auf:

Es ist schlicht falsch, daß uns im Beweisschritt $(**')$ die Tatsache

$\exists k \in \mathbb{N}(M_k(k) = \perp)$

direkt vorliegt. Wir wissen nur, daß dies aus unserer Annahme $D \in \mathrm{RE}$ folgen würde. Um nun (F') zu folgern[12], müßte man mindestens

$\exists k \in \mathbb{N}(M_k(k) = \perp) \Rightarrow \exists k \in \mathbb{N}(\boldsymbol{Beweis}(M_k(k) = \perp))$

zeigen.

Diese Tatsache ist nicht schwer zu verifizieren. Führen wir hierzu vereinfachende Bezeichnungen ein:

A	sei die Aussage	$D \in \mathrm{RE}$,
B	sei die Aussage	$\exists k \in \mathbb{N}(M_k(k) = \perp)$,
C	sei die Aussage	$\exists k \in \mathbb{N}(\boldsymbol{Beweis}(M_k(k) = \perp))$.

Wir wissen $A \Rightarrow B$ und wollen $A \Rightarrow C$ schlußfolgern. Das ist gleichbedeutend mit einem Beweis für $(A \Rightarrow B) \Rightarrow (A \Rightarrow C)$. Dieser Term kann wie folgt umgeformt werden:

$$
\begin{aligned}
&(A \Rightarrow B) \Rightarrow (A \Rightarrow C) \\
\Leftrightarrow \quad &\neg(\neg A \vee B) \vee (\neg A \vee C) \\
\Leftrightarrow \quad &(A \wedge \neg B) \vee \neg A \vee C \\
\Leftrightarrow \quad &(A \vee \neg A) \wedge (\neg B \vee \neg A) \vee C \\
\Leftrightarrow \quad &\neg B \vee \neg A \vee C \\
\Leftrightarrow \quad &\neg A \vee (B \Rightarrow C) \\
\Leftrightarrow \quad &A \Rightarrow (B \Rightarrow C)
\end{aligned}
$$

Ohne an dieser Stelle einen Beweis für die Aussage $B \Rightarrow C$ (also $\exists k \in \mathbb{N}(M_k(k) = \perp) \Rightarrow \exists k \in \mathbb{N}(\boldsymbol{Beweis}(M_k(k) = \perp))$) anzugeben, kann der Beweisschritt (F')

---

[11] Exakt betrachtet steht $(*)$ für die Aussage:
$D \in \mathrm{RE} \Rightarrow \exists \mathrm{k} \in \mathbb{N}\ \forall n \in \mathbb{N}\ (M_k(n) \neq \perp \Leftrightarrow n \in D)$, daraus folgt dann:
$D \in \mathrm{RE} \Rightarrow \exists \mathrm{k} \in \mathbb{N}\ (M_k(k) \neq \perp \Leftrightarrow k \in D)$

[12] Dies ist die schwächste Aussage, die uns zum Ziel führt.

also nicht vollzogen werden, da über den Wahrheitswert der Aussage A (also $D \in \mathrm{RE}$) an dieser Stelle nichts ausgesagt werden kann.

Natürlich ist es unmöglich, aus

$B \qquad \Leftrightarrow \qquad \exists k \in \mathbb{N}(M_k(k) = \perp)$

den Schluß

$C \qquad \Leftrightarrow \qquad \exists k \in \mathbb{N}(\boldsymbol{Beweis}(M_k(k) = \perp))$

zu ziehen.

Hierzu sollte man sich klarmachen, was die verschiedenen Aussagen eigentlich exakt bedeuten.

$\exists k \in \mathbb{N}(M_k(k) = \perp)$ bedeutet:
Es gibt ein (uns unter Umständen völlig unbekanntes) $k \in \mathbb{N}$, welches die Aussage $M_k(k) = \perp$ erfüllt. Man kann die Aussage auch wie folgt umformulieren:

$\exists k \in \mathbb{N}(M_k(k) = \perp) \qquad \Leftrightarrow \qquad \overline{K} \neq \emptyset$

Dabei ist $\overline{K}$ die in Definition 1.4 (Seite 16) eingeführte Menge, die das Halteproblem beschreibt.

$\exists k \in \mathbb{N}(\boldsymbol{Beweis}(M_k(k) = \perp))$ bedeutet:
Es gibt ein $k \in \mathbb{N}$, welches die Aussage $\boldsymbol{Beweis}(M_k(k) = \perp)$ erfüllt. Wie bereits weiter oben beschrieben, impliziert die Formulierung $\boldsymbol{Beweis}(M_k(k) = \perp)$, daß wir das spezielle $k \in \mathbb{N}$, für das $M_k(k) = \perp$ gilt, auch tatsächlich direkt kennen.

Es genügt also nicht, von der prinzipiellen Existenz eines $k \in \mathbb{N}$ mit $M_k(k) = \perp$ zu wissen, um den Beweis von $M_k(k) = \perp$ für ein spezielles *(sozusagen vorgegebenes)* $k \in \mathbb{N}$ zu führen. Stellen wir uns dazu vor, jemand legt uns eine bestimmte natürliche Zahl n vor, mit der Aufgabe, $M_n(n) = \perp$ zu entscheiden. Das Wissen um irgendeine (uns natürlich völlig unbekannte) Zahl $k \in \mathbb{N}$ mit $M_k(k) = \perp$ trägt zu der Entscheidung von $M_n(n) = \perp$ überhaupt nicht bei. Mit Hilfe dieses Wissens kann also kein Beweis für $M_n(n) = \perp$ gefunden werden. Dies gilt für jede beliebige natürliche Zahl n. Für *keine* einzige natürliche Zahl n kann ein Beweis für $M_k(k) = \perp$ auf diese Art und Weise gefunden werden.

Ein weiterer Aspekt der fehlerhaften Beweisführung besteht in folgendem:

Aus

$D \in \mathrm{RE} \Rightarrow \exists \mathrm{k} \in \mathbb{N}(\mathrm{M_k(k)} = \perp) \qquad (**')$

kann noch nicht einmal die Aussage

$D \in \mathrm{RE} \Rightarrow \boldsymbol{Beweis}(\exists \mathrm{k} \in \mathbb{N}(\mathrm{M_k(k)} = \perp)) \qquad (f)$

hergeleitet werden.

Wie oben bereits angedeutet, wurde die Aussage $\exists k \in \mathbb{N}(M_k(k) = \bot)$ nicht bewiesen. Wir wissen nur, daß sie aus der Aussage $D \in \mathrm{RE}$ folgt. Man kann für $(**')$ auch schreiben:

$(D \notin \mathrm{RE}) \vee (\exists \mathrm{k} \in \mathbb{N}(\mathrm{M_k(k)} = \bot))$

Da der Wahrheitswert von $D \notin \mathrm{RE}$ (die Behauptung des Satzes) an dieser Stelle natürlich nicht bekannt ist, kann auch über den Wahrheitswert von $\exists k \in \mathbb{N}(M_k(k) = \bot)$ nichts ausgesagt werden. Es liegt also nach wie vor kein Beweis der Aussage $\exists k \in \mathbb{N}(M_k(k) = \bot)$ vor, damit kann auch nicht $\boldsymbol{Beweis}(\exists k \in \mathbb{N}(M_k(k) = \bot))$ geschlußfolgert werden.

Die Aussage (f) ist schwächer als die benötigte Aussage

$$D \in \mathrm{RE} \Rightarrow \exists \mathrm{k} \in \mathbb{N}(\boldsymbol{Beweis}(\mathrm{M_k(k)} = \bot)) \qquad (F')$$

und würde im weiteren Verlauf des Beweises auch nicht zum Ziel führen. Trotzdem zeigt dies eine weiteren Fehler des Beweises. Im nächsten Abschnitt soll noch einmal auf einfache Art gezeigt werden, welche Folgen Beweise hätten, die diesen fehlerhaften Beweisschritt verwenden.

2.7 Ein weiterer Beweisversuch

Es wird nun ein Beweis angegeben, der das gleiche Beweisprinzip wie Satz 2.1 (Seite 32) verwendet, aber eine eindeutig falsche Aussage beweist:

Versuch eines Satzes 2.2.

Für jede beliebige mathematische Aussage A gilt:

$\boldsymbol{Beweis}(A) \vee \boldsymbol{Beweis}(\neg A)$

Beweis: *Für den Beweis der Aussage beweisen wir für beliebige Aussagen A die Hilfsaussage* $(*)$*:*

$(*) \qquad A \vee \boldsymbol{Beweis}(\neg A)$

Beweis: (indirekt)
Annahme: $\neg A \wedge \neg \boldsymbol{Beweis}(\neg A)$

$\Rightarrow \qquad \neg A \qquad (1)$

$\Rightarrow \qquad \neg \boldsymbol{Beweis}(\neg A) \qquad (2)$

Da die Tatsache (1) *soeben durch menschliche Mathematiker (wir selbst und der verstehende Leser) herausgefunden wurde, liegt also offensichtlich ein Beweis für* $\neg A$ *vor*[13]*, damit gilt* $\boldsymbol{Beweis}(\neg A)$*, im Widerspruch zu* (2)*.* □

[13]Hier wird bewußt der gleiche Wortlaut wie im oben aufgeführten Beweis für Satz 2.1 (Seite 32) verwendet.

Da $(*)$ *für beliebige Aussagen A bewiesen wurde, gilt außerdem (für* $\neg A$*):*

$(**)$ $\quad \neg A \vee \boldsymbol{Beweis}(A).$

Jede mathematische Aussage A hat einen eindeutigen Wahrheitswert, damit entstehen hier also 2 Fälle:

1. $A = wahr$

mit $(**)$ *folgt* $\boldsymbol{Beweis}(A)$.

2. $A = falsch$

mit $(*)$ *folgt* $\boldsymbol{Beweis}(\neg A)$.

Daraus folgt die Behauptung $\boldsymbol{Beweis}(A) \vee \boldsymbol{Beweis}(\neg A)$

Es muß wohl nicht näher erläutert werden, warum der Satz offensichtlich eine falsche Aussage beweist[14], die Auflösung des Problems liegt also im verwendeten (falschen) Beweis.

2.8 Abschlußbemerkungen

Obwohl interpretierende Aspekte der vorliegenden Arbeit schon im Abschnitt 2.4 (Seite 29) und im philosophischen Abschnitt 2.5 (Seite 30) angeschnitten wurden, soll hier noch einmal eine kurze Zusammenfassung des Erreichten erfolgen.

Wir haben einen Teil der menschlichen Denkens, speziell die Fähigkeit zum korrekten mathematischen Schlußfolgern in der Menge D beschrieben, welche die natürlichen Zahlen n enthält, für die $\boldsymbol{Beweis}(M_n(n) = \perp)$ gilt.

Es konnte erfolgreich der Satz 2.1 (Seite 26) gezeigt werden.

Dieser Satz sagt sinngemäß aus, daß kein $i \in \mathbb{N}$ gefunden werden kann, für das ein mathematisch korrekter Beweis für $D_i = D$ angegeben werden kann, also daß für keine bekannte TURING-Maschine bewiesen werden kann, daß sie D aufzählt.

Ein Beweis 2.1 (Seite 32) der wesentlich schärferen Behauptung $D \notin \mathrm{RE}$ gelang nicht. Es wurden allerdings genauere Untersuchungen von Fehlern in einem speziellen Beweisversuch dieser Behauptung vorgenommen.

Eine lohnenswerte Aufgabe für die Zukunft könnte darin bestehen, einen Beweis für $D \in \mathrm{RE}$ oder $D \notin \mathrm{RE}$ zu finden. Beide Möglichkeiten stehen nach wie vor

[14] Auch hier sei noch einmal auf den bahnbrechenden Satz von KURT GÖDEL hingewiesen, der besagt, daß es in genügend komplizierten mathematischen Systemen Aussagen A gibt, für die weder $\boldsymbol{Beweis}(A)$ noch $\boldsymbol{Beweis}(\neg A)$ gilt.

offen (siehe Abschnitt 2.5, Seite 30), wobei nach Meinung des Autors wohl eher der Variante $D \notin \mathrm{RE}$ in Frage kommt.

Zum Abschluß soll noch einmal KURT GÖDEL zu Wort kommen, der sich passend zum Thema der Arbeit äußerte[15]:

„Der menschliche Verstand ist nicht fähig, alle seine mathematischen Intuitionen zu formulieren (oder zu mechanisieren). Gelingt es ihm, einige davon zu formulieren, so liefert diese Tatsache neues intuitives Wissen, zum Beispiel die Konsistenz des Formalismus.

Diesen Sachverhalt könnte man die „Unvollendbarkeit“ der Mathematik nennen.

Andererseits bleibt es auf Basis des bislang Bewiesenen möglich, daß es eine Theorem-Beweismaschine geben könnte (die möglicherweise sogar empirisch nicht zu entdecken ist), welche tatsächlich der mathematischen Intuition gleichwertig ist. Das kann aber nicht bewiesen werden, ebensowenig wie bewiesen werden kann, daß diese Maschine in der Zahlentheorie nur korrekte Sätze liefert.“

[15]Zitiert nach HAO WANG, *From Mathematics to Philosophy* (New York: Humanities Press, 1974) Seite 324. WANG entnahm das Zitat dem unveröffentlichten Text einer Vorlesung, die GÖDEL in Providence am 26. Dezember 1951 gehalten hat.

Abkürzungen / mathematische Symbole

$\mathbb{N}$ Die natürlichen Zahlen (selbstverständlich inklusive „Null“):

$\mathbb{N} = \{0, 1, 2, 3, 4, 5, 6, 7, \ldots\}$

RE Die Menge der aufzählbaren Mengen (siehe Definition 1.3, Seite 14)

REC Die Menge der entscheidbaren Mengen (siehe Definition 2.1, Seite 20)

$\emptyset$ Die leere Menge, auch $\{\}$.

Alphabet Der Begriff wird im übertragenen Sinne für eine endliche nichtleere Mengen benutzt. Meist bestehen Alphabete aus bestimmten Symbolen: Buchstaben, Zahlen, auch Sonderzeichen. So ist z.B. $\{0, 1, A, \_\}$ ein Alphabet, bestehend aus den vier Zeichen 0, 1, A und „_“.

Binärzahl (auch Dualzahl) Natürliche Zahlen, die im Dualsystem (Zweiersystem) dargestellt werden, bezeichnet man als Binärzahlen. Solche Zahlen bestehen lediglich aus Nullen und Einsen und zwar mit folgender Vereinbarung:

$n = \sum_{k=0}^{m} z_k 2^k$

Dabei ist z_k die k-te Ziffer der Zahlendarstellung von rechts (begonnen wird mit 0) nach links. Mit der zusätzlichen Vereinbarung, daß jede von Null verschiedene Binärzahl mit einer 1 beginnen muß, ist m die Stellenanzahl der Zahl minus 1. Die natürlichen Zahlen stellen sich als Binärzahlen also wie folgt dar:

$\mathbb{N} = \{0, 1, 10, 11, 100, 101, 110, 111, 1000, \ldots\}$

Definitionsbereich Der Definitionsbereich einer Funktion ist die Menge aller Werte, für die die Funktion definiert ist, also die Menge der Urbilder bzw. die Menge der Argumentwerte der Funktion. Für Funktionen $f : A \to B$ ist A der Definitionsbereich, B der Wertebereich der Funktion.

deterministisch Eine deterministische (=„vorbestimmte“, =„nicht zufällige“) Maschine hat zu keinem Zeitpunkt ihres Programmablaufs irgendwelche Wahlmöglichkeiten. Bei gleicher Eingabe gelangt sie immer über einen exakt gleichen Ablauf zum gleichen Ergebnis. Im Gegensatz dazu kann man auch nichtdeterministische Maschinen betrachten, die an gewissen Stellen im Programmablauf zwischen mehreren Möglichkeiten der Programmabarbeitung zufällig wählen.

eindeutig Eindeutige Relation:

$R \subseteq X \times Y$ ist eindeutig, wenn für beliebige $a \in X$ und $b, c \in Y$ gilt:

Aus $(a, b) \in R$ und $(a, c) \in R$ folgt $b = c$.

Funktionen sind eindeutige Relationen.

eineindeutig Eineindeutige Abbildung, Funktion:

Eine Funktion $f : A \to B$ ist eineindeutig, wenn für beliebige $a, b \in A$ und $c \in B$ gilt:

Aus $f(a) = c$ und $f(b) = c$ folgt $a = b$.

gleichmächtig Zwei Mengen A und B sind gleichmächtig, wenn ihre Mächtigkeit übereinstimmt. Es gibt dann eine eineindeutige Funktion $f : A \to B$ von A auf B.

Input Eingabe

Komplement Das Komplement einer Menge A, die Teilmenge einer Menge M ist, besteht aus den Elementen von M, die nicht zu A gehören, und wird mit $\overline{A}$ bezeichnet:

$\overline{A} = M \setminus A$

Mächtigkeit (auch Kardinalität) Die Mächtigkeit einer endlichen Menge entspricht der Anzahl der Elemente. die Mächtigkeit von $\mathbb{N}$ ist $\aleph_0$, die Mächtigkeit von $\mathbb{R}$ ist $\aleph_1$. Die Mächtigkeit einer Menge A wird mit $|A|$ bezeichnet.

Output Ausgabe

Potenzmenge Die Potenzmenge einer Menge M ist die Menge aller Teilmengen von M. Sie enthält z.B. die leere Menge $\emptyset$ und die Menge M selbst. Als verkürzende Bezeichnung ist auch 2^M gebräuchlich. Enthält die Menge M endlich viele, sagen wir m, Elemente, so gilt: $|2^M| = 2^m$.

Literaturverzeichnis

[1] L. Mäurer, B. Schlingelhof; Berechenbarkeit des Denkens?
Wurzel, Bd. 01/00, 2000.

[2] R. Penrose; Computerdenken – Des Kaisers neue Kleider oder Die Debatte um Künstliche Intelligenz, Bewußtsein und die Gesetze der Physik.
Spektrum Heidelberg, 1991.

[3] R. Penrose; Schatten des Geistes – Wege zu einer neuen Physik des Bewusstseins.
Spektrum Heidelberg, 1995.

[4] R. Rucker; Die Ufer der Unendlichkeit.
Wolfgang Krüger Verlag, 1989.

[5] G. Wechsung; C. Beckstein; Berechenbarkeit des Denkens? (Material zum Seminar).
FSU Jena, 1999.

Danksagung

Zuallererst danke ich meinem Betreuer Prof. Dr. G. Wechsung für die vielen Anregungen, die am Ende zu diesem Arbeitsthema geführt haben. Ohne ihn hätte diese Arbeit nicht stattgefunden.

Weiterhin gilt mein Dank Ina Mäurer, Andreas Zollmann, Karsten Krüger und Ben Schlingelhof für die ausschweifenden Diskussionen zum Thema.

Frank Mäurer, Nadja Großbach, Johannes Haschke, Ina Mäurer, André Große und Thomas Schneider haben Kommentare und Korrekturen zu früheren Versionen der Arbeit geliefert und mir sehr geholfen.

Konrad Schöbel hat mir über Jahre(!) das Buch „Schatten des Geistes“ von R. Penrose geliehen. Dafür ein ganz besonderer Dank.

Zu guter Letzt sei allen direkten und indirekten Familienmitgliedern und vielen Freunden Dank für ihre Aufmunterungen gesagt.

Erklärung

Hiermit erkläre ich, daß ich die am heutigen Tage eingereichte Arbeit zum Thema „Berechenbarkeit des Denkens?“ selbständig erarbeitet, verfasst und Zitate kenntlich gemacht habe. Andere als die angegebenen Hilfsmittel wurden nicht verwendet.

Datum Unterschrift