Die Unentscheidbarkeit extensionaler Eigenschaften von Turingmaschinen: der Satz von Rice

Holger Arnold*

Dieser Text befasst sich mit der Frage, unter welchen Bedingungen das Problem, zu bestimmen, ob die von einer Turingmaschine akzeptierte Sprache oder die von einer Turingmaschine berechnete Funktion eine bestimmte Eigenschaft besitzt, entscheidbar oder aufzählbar ist. Als Werkzeug für den Beweis, dass eine Sprache unentscheidbar oder nicht aufzählbar ist, werden wir die Technik der funktionalen Reduktion einführen und an einigen Beispielen anwenden. Es wird sich herausstellen, dass sich mit dieser Technik tatsächlich für *alle* interessanten Eigenschaften aufzählbarer Sprachen und partiell berechenbarer Funktionen zeigen lässt, dass sie unentscheidbar sind. Dieses Ergebnis, das als Satz von Rice bekannt ist, ist aus theoretischer Sicht von Bedeutung, weil es eine grundlegende Eigenschaft aller Turing-äquivalenten Berechenbarkeitsmodelle charakterisiert. Darüber hinaus besitzt es weitreichende praktische Auswirkungen, weil es bedeutet, dass sich prinzipiell kein Programm konstruieren lässt, dass automatisch prüft, ob ein gegebenes Programm ein bestimmtes Verhalten besitzt.

1. Extensionale Eigenschaften von Turingmaschinen. Als *extensionale Eigenschaften* von Turingmaschinen bezeichnet man Eigenschaften, die sich ausschließlich auf die von einer Turingmaschine akzeptierte Sprache oder berechnete Funktion beziehen. Die Eigenschaft, eine unendliche Sprache zu akzeptieren, ist beispielsweise eine extensionale Eigenschaft von Turingmaschinen; die Eigenschaft, weniger als 10 Zustände zu besitzen, dagegen nicht. Jede extensionale Eigenschaft charakterisiert also eine Teilmenge der aufzählbaren Sprachen oder der partiell berechenbaren Funktionen.

Das Problem, zu bestimmen, ob eine Turingmaschine eine bestimmte Eigenschaft besitzt, wird durch die Menge aller Beschreibungen von Turingmaschinen repräsentiert, die diese Eigenschaft besitzen. Für eine extensionale Eigenschaft P von Turingmaschinen gilt dann: akzeptieren zwei Turingmaschinen M und M' die gleiche Sprache oder berechnen sie die gleiche Funktion, dann ist $\langle M' \rangle$ genau dann in P enthalten, wenn $\langle M \rangle$ in P enthalten ist.

Eigenschaften, die alle Turingmaschinen besitzen und Eigenschaften, die keine Turingmaschine besitzt, bezeichnet man als *triviale Eigenschaften*. Man kann sich leicht überlegen, dass diese entscheidbar sind — erstere werden durch jede immer haltende Turingmaschine entschieden, die alle wohlgeformten Beschreibungen von Turingmaschinen akzeptiert (dabei wird vorausgesetzt, dass entscheidbar ist, ob eine Zeichenkette tatsächlich eine Turingmaschine codiert) und letztere von jeder immer haltenden Turingmaschine, die überhaupt kein Wort akzeptiert.

Datum: 15. Januar 2010.

*Kontaktdaten unter `http://harnold.org/`.

Interessanter als die trivialen Eigenschaften sind natürlich Eigenschaften, die echte nichtleere Teilmengen der aufzählbaren Sprachen oder der berechenbaren Funktionen charakterisieren. Ist es beispielsweise entscheidbar, ob die Sprache einer Turingmaschine leer, regulär, kontextfrei, entscheidbar oder endlich ist, oder ob die Funktionswerte der von einer Turingmaschine berechneten Funktion ein bestimmtes Prädikat erfüllen? Tatsächlich lässt sich für jede dieser Eigenschaften zeigen, dass dies nicht der Fall ist. Der Beweis, dass diese Eigenschaften unentscheidbar sind, kann jeweils durch Reduktion des Akzeptanzproblems für Turingmaschinen auf die entsprechende Eigenschaft erfolgen.

Wir werden zunächst Eigenschaften aufzählbarer Sprachen betrachten und später sehen, dass sich die dabei erzielten Ergebnisse unmittelbar auf Eigenschaften partiell berechenbarer Funktionen übertragen lassen.

2. Funktionale Reduktion. Man bezeichnet ein Problem A als reduzierbar auf ein Problem B, wenn eine Lösung für B verwendet werden kann, um A zu lösen. Beispielsweise ist das Problem, zu bestimmen, ob ein Wort in der durch einen regulären Ausdruck beschriebenen Sprache enthalten ist, reduzierbar auf das Problem, zu bestimmen, ob ein Wort von einem endlichen Automaten akzeptiert wird, weil sich für jeden regulären Ausdruck ein endlicher Automat konstruieren lässt, der die von diesem Ausdruck beschriebene Sprache akzeptiert. Ein weiteres Beispiel ist die Multiplikation zweier natürlicher Zahlen; diese lässt sich auf die Addition zweier natürlicher Zahlen reduzieren, weil $a \cdot b = \sum_{i=1}^{b} a$ für alle a und b gilt.

Es gibt verschiedene Arten, Probleme aufeinander zu reduzieren. Eine für den Beweis der Unentscheidbarkeit oder Nichtaufzählbarkeit von Eigenschaften aufzählbarer Sprachen und partiell berechenbarer Funktionen verwendete Variante ist die *funktionale Reduktion* einer Sprache auf eine andere.

Definition 1. Eine *funktionale Reduktion* einer Sprache A auf eine Sprache B ist eine total berechenbare Funktion t, so dass jedes Wort x genau dann in A enthalten ist, wenn $t(x)$ in B enthalten ist. Existiert eine funktionale Reduktion von A auf B, dann heißt A *funktional reduzierbar* auf B und man schreibt $A \leq_{\mathrm{f}} B$.

Für „funktional reduzierbar“ werden auch die Begriffe „many-one-reduzierbar“ oder vereinfacht „reduzierbar“ verwendet.

Eine funktionale Reduktion einer Sprache A auf eine Sprache B ermöglicht es, die Frage nach dem Enthaltensein eines Wortes in A durch die Frage nach dem Enthaltensein eines Wortes in B auszudrücken. Lässt sich diese Frage für die Sprache B beantworten, liefert die Reduktion einen Weg, diese Frage auch für die Sprache A zu beantworten. Dies erklärt die Verwendung des Symbols $\leq_{\mathrm{f}}$: gilt $A \leq_{\mathrm{f}} B$, dann ist A bezüglich Entscheidbarkeit oder Aufzählbarkeit nicht schwerer als B. Der folgende Satz präzisiert diese Aussage.

Satz 2. *Die Sprache A sei funktional reduzierbar auf die Sprache B. Wenn B aufzählbar ist, dann ist auch A aufzählbar. Wenn B entscheidbar ist, dann ist auch A entscheidbar.*

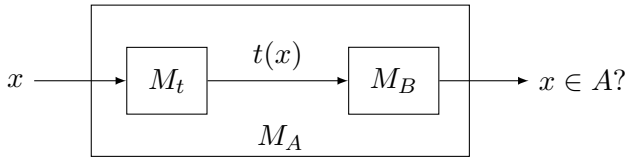

Abbildung 1: Die im Beweis von Satz 2 konstruierte Turingmaschine M_A. Die von M_A simulierte Turingmaschine M_B — und damit auch M_A — akzeptiert genau dann, wenn $t(x)$ in B enthalten ist, was genau dann der Fall ist, wenn x in A enthalten ist.

Beweis. Sei t eine funktionale Reduktion von A auf B. Weil t eine total berechenbare Funktion ist, gibt es eine Turingmaschine M_t, die für jedes Eingabewort x den Wert $t(x)$ berechnet und danach anhält. Wenn B aufzählbar ist, dann gibt es eine Turingmaschine M_B, deren akzeptierte Sprache B ist. Aus den Turingmaschinen M_t und M_B lässt sich eine Turingmaschine M_A konstruieren, die zunächst M_t und danach M_B simuliert. Abbildung 1 illustriert die Konstruktion von M_A. Weil jedes Wort x genau dann in A enthalten ist, wenn $t(x)$ in B enthalten ist, akzeptiert die Turingmaschine M_A genau die Wörter aus A. Die Sprache A ist also aufzählbar. Ist B sogar entscheidbar, dann kann für M_B eine Turingmaschine gewählt werden, die auf jeder Eingabe anhält, wodurch auch M_A auf jeder Eingabe anhält. Die Sprache A ist in diesem Fall also ebenfalls entscheidbar. □

Satz 2 besitzt eine unmittelbare Konsequenz, die verwendet werden kann, um zu beweisen, dass eine Sprache unentscheidbar oder nicht aufzählbar ist.

Korollar 3. *Die Sprache A sei funktional reduzierbar auf die Sprache B. Wenn A unentscheidbar ist, dann ist auch B unentscheidbar. Wenn A nicht aufzählbar ist, dann ist auch B nicht aufzählbar.*

3. Beweis der Unentscheidbarkeit eines Problems. Die Unentscheidbarkeit eines Problems P lässt sich also durch Konstruktion einer funktionalen Reduktion eines geeigneten unentscheidbaren Problems auf P beweisen. „Geeignet“ deshalb, weil sich nicht jedes unentscheidbare Problem auf jedes andere reduzieren lässt. Vielmehr gibt es gewissermaßen verschiedene Schwierigkeitsgrade unentscheidbarer Probleme. Als Ausgangspunkt für die Konstruktion von Reduktionen soll hier das Akzeptanzproblem für Turingmaschinen dienen, dessen Unentscheidbarkeit wir bereits gezeigt haben. Es wird durch die Sprache A_{TM} beschrieben, die definiert ist als

$$A_{\mathsf{TM}} = \{\langle M, w\rangle \mid M \text{ ist eine Turingmaschine, die das Wort } w \text{ akzeptiert}\}.$$

A_{TM} ist die von universellen Turingmaschinen akzeptierte Sprache und wird deshalb auch als universelle Sprache bezeichnet.

Die Vorgehensweise bei der Konstruktion einer Reduktion der Sprache A_{TM} auf eine andere Sprache wird durch den Beweis des folgenden Satzes illustriert. Er zeigt, dass

nicht entscheidbar ist, ob die Sprache einer Turingmaschine nichtleer ist. Die Menge $P_{\neq\emptyset}$ der Beschreibungen von Turingmaschinen, die eine nichtleere Sprache akzeptieren, ist definiert als

$$P_{\neq\emptyset} = \{\langle M\rangle \mid M \text{ ist eine Turingmaschine, deren Sprache nichtleer ist}\}.$$

Wir zeigen zunächst, dass dieses Problem aufzählbar ist.

Satz 4. *Die Sprache $P_{\neq\emptyset}$ ist aufzählbar.*

Beweis. Sei $M_{\neq\emptyset}$ folgende Turingmaschine:

$M_{\neq\emptyset}$ = „Bei Eingabe $\langle M\rangle$:

1. Besitzt x nicht die Form $\langle M\rangle$ für eine Turingmaschine M, dann lehne die Eingabe ab.

2. Sonst sei $\langle M\rangle = x$. Zähle in aufsteigender Ordnung Paare aus Zahlen und Zeichenketten auf. Simuliere für jedes aufgezählte Paar $\langle i, w\rangle$ die Turingmaschine M auf w für i Schritte. Wenn M das Wort w akzeptiert, akzeptiere die Eingabe.“

Wenn die Sprache einer Turingmaschine M nichtleer ist, akzeptiert sie mindestens ein Wort w und dieses Wort wird in einer bestimmten Anzahl i von Schritten akzeptiert. Das Paar $\langle i, w\rangle$ wird von der Turingmaschine $M_{\neq\emptyset}$ irgendwann aufgezählt und damit die Eingabe $\langle M\rangle$ akzeptiert. Dagegen kann die Beschreibung einer Turingmaschine, deren Sprache leer ist, von $M_{\neq\emptyset}$ niemals akzeptiert werden. Die Turingmaschine $M_{\neq\emptyset}$ akzeptiert also die Sprache $P_{\neq\emptyset}$. □

Satz 5. *Die Sprache $P_{\neq\emptyset}$ ist unentscheidbar.*

Beweis. Wir konstruieren eine funktionale Reduktion t der Sprache A_{TM} auf $P_{\neq\emptyset}$. Dazu muss die Funktion t so gewählt werden, dass für jedes Wort x gilt: x besitzt genau dann die Form $\langle M, w\rangle$ für eine Turingmaschine M, die das Wort w akzeptiert, wenn $t(x)$ die Form $\langle M'\rangle$ für eine Turingmaschine M' besitzt, die eine nichtleere Sprache akzeptiert. Die Turingmaschine M_t, welche die Funktion t berechnet, arbeitet wie folgt:

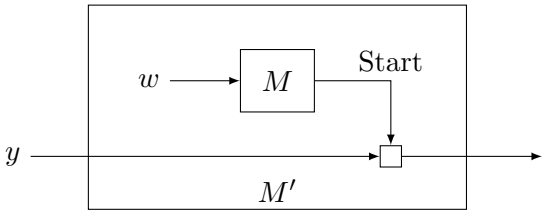

Abbildung 2: Die im Beweis zu Satz 5 von M_t konstruierte Turingmaschine M'. Sie simuliert die Turingmaschine M auf dem Wort w und akzeptiert die Eingabe y, falls w von M akzeptiert wird.

M_t = „Bei Eingabe x:

1. Besitzt die Eingabe nicht die Form $x = \langle M, w\rangle$ für eine Turingmaschine M und ein Wort w, dann gib die Beschreibung einer Turingmaschine aus, die jede Eingabe ablehnt, und halte an.

2. Sonst sei $\langle M, w\rangle = x$. Konstruiere folgende Turingmaschine: M' = „Bei Eingabe y: Simuliere M auf w und akzeptiere y, falls w von M akzeptiert wird.“

3. Gib $\langle M'\rangle$ aus und halte an.“

Abbildung 2 stellt die von M_t konstruierte Turingmaschine M' dar. Die von M_t berechnete Funktion t erfüllt die geforderte Bedingung: Ist $x = \langle M, w\rangle$ für eine Turingmaschine M, die das Wort w akzeptiert, dann ist $t(x)$ die Beschreibung einer Turingmaschine, die jedes Wort akzeptiert, deren Sprache also nichtleer ist. Akzeptiert M dagegen das Wort w nicht oder besitzt x überhaupt nicht die Form $x = \langle M, w\rangle$, dann ist $t(x)$ die Beschreibung einer Turingmaschine, die kein Wort akzeptiert, deren Sprache also leer ist. □

4. Beweis der Nichtaufzählbarkeit eines Problems. Auch die Nichtaufzählbarkeit eines Problems P kann mit der Technik der funktionalen Reduktion bewiesen werden. Dazu ist eine funktionale Reduktion eines geeigneten nicht aufzählbaren Problems auf P zu konstruieren. Aus der Unentscheidbarkeit der Sprache A_{TM} hatten wir bereits die Nichtaufzählbarkeit ihres Komplements $\overline{A_{\mathsf{TM}}}$ gefolgert. Wir werden diese Sprache nun verwenden, um zu zeigen, dass das Problem, zu bestimmen, ob die Sprache einer beliebigen Turingmaschine entscheidbar ist, nicht aufzählbar ist. Die Menge $P_{\text{decidable}}$ der Beschreibungen von Turingmaschinen, die eine entscheidbare Sprache akzeptieren, ist definiert als

$$P_{\text{decidable}} = \{\langle M\rangle \mid M \text{ ist eine Turingmaschine, deren Sprache entscheidbar ist}\}.$$

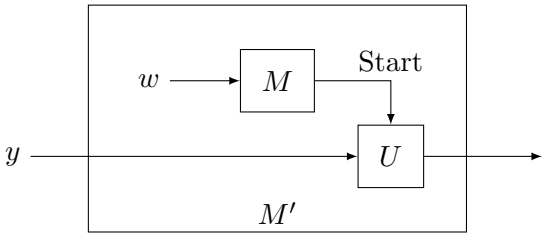

Abbildung 3: Die im Beweis zu Satz 6 von M_t konstruierte Turingmaschine M'. Sie simuliert zunächst die Turingmaschine M auf dem Wort w. Falls M akzeptiert, simuliert sie die universelle Turingmaschine U auf der Eingabe y und akzeptiert diese, falls y von U akzeptiert wird.

Satz 6. *Die Sprache $P_{\text{decidable}}$ ist nicht aufzählbar.*

Beweis. Wir konstruieren eine funktionale Reduktion t der Sprache $\overline{A_{\mathsf{TM}}}$ auf $P_{\text{decidable}}$. Dazu muss t so gewählt werden, dass für jedes Wort x gilt: x besitzt genau dann *nicht* die Form $\langle M, w\rangle$ für eine Turingmaschine M, die das Wort w akzeptiert, wenn $t(x)$ die Form $\langle M'\rangle$ für eine Turingmaschine M' besitzt, die eine entscheidbare Sprache akzeptiert. Dabei ist übrigens nicht von Bedeutung, ob M' selbst auf jeder Eingabe anhält, sondern nur, dass es irgendeine Turingmaschine gibt, welche die von M' akzeptierte Sprache entscheidet. Die Turingmaschine M_t, welche die Funktion t berechnet, arbeitet wie folgt:

M_t = „Bei Eingabe x:

1. Besitzt die Eingabe nicht die Form $x = \langle M, w\rangle$ für eine Turingmaschine M und ein Wort w, dann gib die Beschreibung einer Turingmaschine aus, die jede Eingabe ablehnt, und halte an.

2. Sonst sei $\langle M, w\rangle = x$. Konstruiere folgende Turingmaschine: M' = „Bei Eingabe y: Simuliere M auf w. Falls M akzeptiert, simuliere die universelle Turingmaschine U auf y und akzeptiere y, falls U akzeptiert.“

3. Gib $\langle M'\rangle$ aus und halte an.“

Abbildung 3 stellt die von M_t konstruierte Turingmaschine M' dar. Die von M_t berechnete Funktion erfüllt die geforderte Bedingung: Ist $x = \langle M, w\rangle$ für eine Turingmaschine M, die das Wort w nicht akzeptiert, oder besitzt x überhaupt nicht die Form $x = \langle M, w\rangle$, dann ist $t(x)$ die Beschreibung einer Turingmaschine, die kein Wort akzeptiert, deren Sprache also leer und damit entscheidbar ist. Akzeptiert M dagegen das Wort w, dann ist $t(x)$ die Beschreibung einer Turingmaschine, welche die gleiche Sprache akzeptiert, wie die universelle Turingmaschine U. Diese akzeptiert aber die unentscheidbare Sprache A_{TM}. □

5. Der Satz von Rice. Die im Beweis der Unentscheidbarkeit von $P_{\neq\emptyset}$ verwendete Konstruktion lässt sich tatsächlich auf *alle* nichttrivialen extensionalen Eigenschaften von Turingmaschinen verallgemeinern. Dieses bemerkenswerte Ergebnis ist als Satz von Rice bekannt [2]:

Satz 7 (Satz von Rice)**.** *Sei $\mathcal{P}$ eine Teilmenge der aufzählbaren Sprachen. Dann ist die Sprache $P = \{\langle M\rangle \mid L_M \in \mathcal{P}$ für eine Turingmaschine $M\}$ genau dann entscheidbar, wenn $\mathcal{P}$ trivial ist.*

Beweis. Wir hatten bereits argumentiert, warum die trivialen Eigenschaften entscheidbar sind. Sei $\mathcal{P}$ also eine nichttriviale Teilmenge der aufzählbaren Sprachen. Enthält $\mathcal{P}$ die leere Sprache, dann betrachten wir stattdessen das Komplement von $\mathcal{P}$ bezüglich der Menge der aufzählbaren Sprachen. Dieses ist ebenfalls nichttrivial und genau dann entscheidbar, wenn $\mathcal{P}$ entscheidbar ist. Wir können daher o.B.d.A. annehmen, dass $\mathcal{P}$ nicht die leere Sprache enthält und dass jede Turingmaschine, deren Beschreibung in P enthalten ist, eine nichtleere Sprache akzeptiert (für $P_{\neq\emptyset}$ in Satz 5 war das trivialerweise der Fall).

Wir konstruieren eine funktionale Reduktion t der Sprache A_{TM} auf die Sprache P. Dazu muss die Funktion t so gewählt werden, dass für jedes Wort x gilt: x besitzt genau dann die Form $\langle M, w\rangle$ für eine Turingmaschine M, die das Wort w akzeptiert, wenn $t(x)$ die Form $\langle M'\rangle$ für eine Turingmaschine M' besitzt, die in P enthalten ist.

Weil $\mathcal{P}$ nichttrivial ist, enthält P mindestens für eine Sprache die Beschreibungen der Turingmaschinen, die diese Sprache akzeptieren. Sei N eine solche Turingmaschine (die im Beweis zu Satz 5 gewählte Turingmaschine war eine, die jedes Wort akzeptiert). Die Turingmaschine M_t, welche die Funktion t berechnet, arbeitet wie folgt:

M_t = „Bei Eingabe x:

1. Besitzt die Eingabe nicht die Form $x = \langle M, w\rangle$ für eine Turingmaschine M und ein Wort w, dann gib die Beschreibung einer Turingmaschine aus, die jede Eingabe ablehnt, und halte an.

2. Sonst sei $\langle M, w\rangle = x$. Konstruiere folgende Turingmaschine: M' = „Bei Eingabe y: Simuliere M auf w. Falls M akzeptiert, simuliere N auf y und akzeptiere y, falls N akzeptiert.“

3. Gib $\langle M'\rangle$ aus und halte an.“

Abbildung 4 stellt die von M_t konstruierte Turingmaschine M' dar. Die von M_t berechnete Funktion erfüllt die geforderte Bedingung: Ist $x = \langle M, w\rangle$ für eine Turingmaschine M, die das Wort w akzeptiert, dann ist $t(x)$ die Beschreibung einer Turingmaschine, welche die gleiche Sprache akzeptiert, wie die Turingmaschine N, deren Beschreibung in P enthalten ist. Weil $\mathcal{P}$ eine extensionale Eigenschaft von Turingmaschinen charakterisiert, muss auch

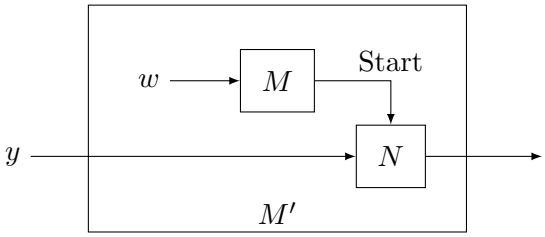

Abbildung 4: Die im Beweis zu Satz 7 von M_t konstruierte Turingmaschine M'. Sie simuliert zunächst die Turingmaschine M auf dem Wort w. Falls M akzeptiert, simuliert sie die Turingmaschine N auf der Eingabe y und akzeptiert diese, falls y von N akzeptiert wird.

die Beschreibung $t(x)$ in P enthalten sein. Akzeptiert M dagegen das Wort w nicht oder besitzt x überhaupt nicht die Form $x = \langle M, w\rangle$, dann ist $t(x)$ die Beschreibung einer Turingmaschine, die kein Wort akzeptiert, deren Sprache also leer ist. Wir hatten aber angenommen, dass $\mathcal{P}$ nicht die leere Sprache enthält. Folglich kann in diesem Fall die Beschreibung $t(x)$ nicht in P enthalten sein. □

Der Satz von Rice lässt sich unmittelbar auf Eigenschaften partiell berechenbarer Funktionen übertragen. Dazu verwenden wir, dass eine Funktion f genau dann partiell berechenbar ist, wenn der Graph von f, also die Menge $\operatorname{graph} f = \{\langle x, f(x)\rangle \mid x \in \operatorname{dom} f\}$, eine aufzählbare Sprache ist.

Korollar 8. *Sei $\mathcal{P}$ eine Teilmenge der partiell berechenbaren Funktionen. Dann ist die Sprache $P = \{\langle M\rangle \mid f_M \in \mathcal{P}\}$ genau dann entscheidbar, wenn $\mathcal{P}$ trivial ist.*

Beweis. Wir nehmen wieder an, dass $\mathcal{P}$ nichttrivial ist. Wir betrachten die Mengen $\mathcal{P}' = \{L \mid L = \operatorname{graph} f$ für eine partiell berechenbare Funktion $f \in \mathcal{P}\}$ und $P' = \{\langle M\rangle \mid L_M \in \mathcal{P}'\}$. Die Menge $\mathcal{P}'$ ist eine nichttriviale Teilmenge der aufzählbaren Sprachen und P' ist die ihr zugeordnete Menge von Turingmaschinen-Beschreibungen. Nach Satz 7 ist P' unentscheidbar.

Um zu zeigen, dass daraus die Unentscheidbarkeit von P folgt, konstruieren wir eine funktionale Reduktion t von P' auf P. Weil $\mathcal{P}$ nichttrivial ist, enthält P mindestens für eine partiell berechenbare Funktion keine Beschreibung einer Turingmaschine, die diese Funktion berechnet. Sei K eine solche Turingmaschine, deren Beschreibung nicht in P enthalten ist. Die Turingmaschine M_t, welche die Funktion t berechnet, arbeitet wie folgt:

M_t = „Bei Eingabe x:

1. Besitzt die Eingabe nicht die Form $x = \langle M \rangle$ für eine Turingmaschine M, dann gib $\langle K \rangle$ aus und halte an.

2. Sonst sei $\langle M \rangle = x$. Konstruiere folgende Turingmaschine: M' = „Bei Eingabe y: Zähle in aufsteigender Ordnung Paare aus Zahlen und Zeichenketten auf. Simuliere für jedes aufgezählte Paar $\langle i, z \rangle$ die Turingmaschine M auf $\langle y, z \rangle$ für i Schritte. Wenn M akzeptiert, gib z aus und halte an.“

3. Gib $\langle M' \rangle$ aus und halte an.“

Die Funktion t erfüllt die an eine funktionale Reduktion von P' auf P gestellte Bedingung: Ist $x = \langle M \rangle$ für eine Turingmaschine M, deren Beschreibung in P' enthalten ist, dann akzeptiert M die Sprache graph f für eine partiell berechenbare Funktion $f \in \mathcal{P}$. Damit hält M' auf einer Eingabe y genau dann mit einem Funktionswert z an, wenn die Turingmaschine M das Paar $\langle y, z \rangle$ akzeptiert, was genau dann der Fall ist, wenn $z = f(y)$ gilt. In diesem Fall ist $t(x)$ also die Beschreibung einer Turingmaschine, welche die Funktion f berechnet und somit in P enthalten. Ist M dagegen nicht in P' enthalten, dann unterscheidet sich die Sprache von M für jede Funktion $f \in \mathcal{P}$ von der Sprache graph f. Die Turingmaschine, deren Beschreibung $t(x)$ ist, berechnet in diesem Fall keine der Funktionen aus $\mathcal{P}$ und folglich ist $t(x)$ nicht in P enthalten. Besitzt x überhaupt nicht die Form $x = \langle M \rangle$, dann ist $t(x) = \langle K \rangle$, was ebenfalls nicht in P enthalten ist. □

6. Konsequenzen des Satzes von Rice. Der Satz von Rice besagt, dass sich die Frage, ob die von einer Turingmaschine akzeptierte Sprache oder berechnete Funktion eine bestimmte nichttriviale Eigenschaft besitzt, nicht mechanisch beantworten lässt. Das bedeutet aber nicht, dass alle Fragen die Turingmaschinen betreffen unentscheidbar sind, denn der Satz von Rice bezieht sich eben ausschließlich auf *extensionale* Eigenschaften. Intensionale Eigenschaften, die den Aufbau von Turingmaschinen oder den Ablauf von Berechnungen charakterisieren, können durchaus entscheidbar sein. Trotzdem liefert der Satz von Rice eine negative Antwort auf eine Frage von erheblicher praktischer Bedeutung, nämlich auf die Frage, ob es möglich ist, ein Programm zu konstruieren, dass automatisch prüft, ob ein beliebiges Programm ein bestimmtes extensionales Verhalten besitzt.[1]

7. Der Satz von Rice für aufzählbare Eigenschaften. Auch für die Aufzählbarkeit extensionaler Eigenschaften von Turingmaschinen ist eine Charakterisierung möglich, die allerdings etwas komplizierter ist als die für die Entscheidbarkeit. Der folgende Satz, der ebenfalls von Rice [3] gefunden wurde, gibt die Bedingungen an eine Teilmenge der

[1] Wegen der Turing-Äquivalenz der üblichen Programmiersprachen und Turingmaschinen lässt sich der Satz von Rice direkt auf Programme übertragen.

aufzählbaren Sprachen an, unter denen die ihr zugeordnete Menge von Turingmaschinen-Beschreibungen aufzählbar ist. Wir verzichten an dieser Stelle auf einen Beweis. Dieser kann in Hopcrofts und Ullmans Lehrbuch [1] nachgelesen werden.

Satz 9 (Satz von Rice für aufzählbare Eigenschaften)**.** *Sei $\mathcal{P}$ eine Teilmenge der aufzählbaren Sprachen. Dann ist die Sprache $P = \{\langle M\rangle \mid L_M \in \mathcal{P}$ für eine Turingmaschine $M\}$ genau dann aufzählbar, wenn folgende Bedingungen erfüllt sind:*

1. *Wenn $\mathcal{P}$ eine Sprache L enthält, dann enthält $\mathcal{P}$ auch jede aufzählbare Obermenge von L.*
2. *Wenn $\mathcal{P}$ eine unendliche Sprache L enthält, dann enthält $\mathcal{P}$ auch eine endliche Teilmenge von L.*
3. *Die Menge $\{w_1\#\ldots\#w_n\# \mid \{w_1,\ldots,w_n\}$ ist eine endliche Sprache in $\mathcal{P}\}$ ist aufzählbar.*

Literatur

1. John E. Hopcroft and Jeffrey D. Ullman. *Einführung in die Automatentheorie, Formale Sprachen und Komplexitätstheorie*. Oldenbourg Verlag, fourth edition, 2000. Basiert auf der 1979er Ausgabe des englischsprachigen Originals.
2. H. G. Rice. Classes of recursively enumerable sets and their decision problems. *Transactions of the American Mathematical Society*, 74(2):358–366, 1953. doi: 10.1090/S0002-9947-1953-0053041-6.
3. H. G. Rice. On completely recursively enumerable classes and their key arrays. *Journal of Symbolic Logic*, 21(3):304–308, 1956.