39

Nichtberechenbare Funktionen

Das Busy-beaver-Problem

Biber[1] sind für ihre Fähigkeit bekannt, bei einer Aufgabe zu bleiben, bis sie vollendet ist. Sie durchqueren fleißig das Wasser zwischen Wald und Damm und tragen bei jeder Tour einen neuen Stock oder Zweig für die Konstruktion des Dammes mit sich.

Turing-Maschinen (vgl. Kapitel 31), die entlang ihres Bandes hin- und herlaufen, hier ein Symbol lesen und dort ein Symbol schreiben, erinnern uns an Biber. Wie fleißig kann nun eine Turing-Maschine sein? Einige Turing-Maschinen sind in gewissem Sinn unendlich fleißig, da sie nie anhalten. Darüber hinaus können viele der Turing-Maschinen, die anhalten, für beliebig lange Zeit beschäftigt werden, indem man vor jedem Lauf den Anfangszustand ihres Bandes ändert. Daher erscheint es vernünftig, diese Frage im Zusammenhang mit einem anfänglich leeren Band für alle Maschinen zu stellen, die mit solch einem Band als Eingabe anhalten.

1962 dachte sich Tibor Rado, ein ungarischer Mathematiker, das aus, was heute als das Busy-beaver-Problem (Abbildung 39.1) bezeichnet wird: Gegeben sei eine Turing-Maschine mit n Zuständen und einem zweielementigen Alphabet {0, 1}; was ist die maximale Anzahl von Ein-

1 Engl. busy beaver = fleißiger Biber. Im Deutschen ist die englische Bezeichnung gebräuchlich, so daß man von Busy-beaver-Problem und Busy-beaver-Funktion spricht. A.d.Ü.

A. K. Dewdney, *Der Turing Omnibus*

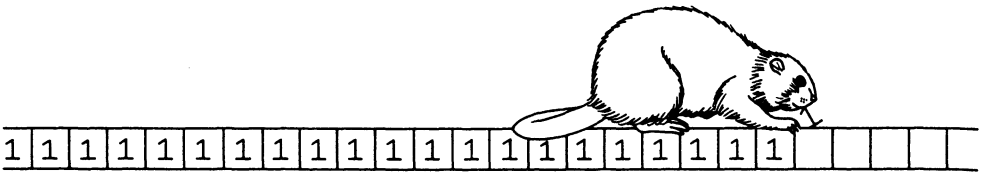

Abbildung 39.1 Ein fleißiger Biber schreibt eine weitere 1

sen, welche die Maschine auf ein anfangs leeres (mit Nullen gefülltes) Band schreiben kann, bevor sie anhält? Es steht außer Frage, daß diese Zahl, die wir als $\Sigma(n)$ bezeichnen, existiert, denn die Anzahl der Turing-Maschinen ist endlich.

Neben anderen Gründen ist es eine Aufgabe dann wert, gelöst zu werden, wenn sie schwierig ist. Das Busy-beaver-Problem kann allgemein mit einem Computer nicht gelöst werden, da die Funktion $\Sigma(n)$ schneller als jede berechenbare Funktion $f(n)$ wächst. Um erkennen zu können, daß dies zutrifft, nehmen wir an, daß B eine Turing-Maschine ist, welche die Busy-beaver-Funktion $\Sigma(n)$ berechnet und daß B q Zustände besitzt. Dies bedeutet, daß B, wenn sie mit einem Eingabeband konfrontiert wird, auf dem die ganze Zahl n geschrieben ist, die Zahl $\Sigma(n)$ ausgibt. Ohne Verlust der Allgemeingültigkeit können wir annehmen, daß sowohl n als auch $\Sigma(n)$ binär geschrieben sind.

Nun sei B_n eine Turing-Maschine, die n Zustände besitzt und $\Sigma(n)$ Einsen ausgibt, bevor sie anhält; C sei eine Maschine, die eine binäre in eine unäre Zahl umwandelt, und A sei eine Maschine, die ein leeres Band in ein anderes umwandelt, auf das die Zahl n binär geschrieben ist. Fügt man die drei Maschinen zusammen, können sie durch ABC dargestellt werden. Hierbei handelt es sich um eine einzige Maschine, die mit einem leeren Band beginnt und mit $\Sigma(n)$ Einsen auf dem Band anhält. Außerdem besitzt sie nur $[\log n] + q + \mathrm{r}$ Zustände, da $[\log n]$ Zustände von A benötigt werden, und eine konstante Anzahl von Zuständen, q und r, für B bzw. C erforderlich sind.

Für jede ganze Zahl n, so daß gilt $n > [\log n] + q + r$, schreibt Maschine ABC genauso viele Einsen wie B_n und benutzt trotzdem weniger Zustände! Da es jedoch leicht zu zeigen ist, daß $\Sigma(n) > \Sigma(m)$, wenn $n > m$, erhält man einen offensichtlichen Widerspruch. Da die Maschinen A und C eindeutig existieren, kann B nicht existieren. Entsprechend ist $\Sigma(n)$ keine berechenbare Funktion.

Die nachfolgende Tabelle faßt unseren augenblicklichen Wissensstand über die Werte von $\Sigma(n)$ zusammen:

n	$\Sigma(n)$
1	1
2	4
3	6
4	13
5	≥ 4098

Der Sprung von Σ=13 bei n=4 auf $\Sigma \geq 4098$ bei n=5 ist symptomatisch für die nichtberechenbare Natur von Σ'. Die Zahl 4098 hat eine interessante Geschichte hinter sich.

In der Augustausgabe 1984 des *Scientific American* erschien ein Artikel über den damals fleißigsten bekannten 5-Zustands-Biber, der 1984 von Uwe Schult, einem deutschen Informatiker, gefunden wurde. Schults fleißiger Biber erzeugte 501 Einsen, bevor er anhielt. Als Antwort auf den Artikel führte George Uhing, ein amerikanischer Programmierer, eine Computersuche nach fleißigen Bibern mit fünf Zuständen durch und fand einen, der 1915 Einsen ausgab, bevor er anhielt. Später, 1989, wurde Uhings Rekordbiber durch einen neuen, fleißigeren Biber verdrängt, der von Jürgen Buntrock und Heiner Marxen in Deutschland entdeckt wurde. Der Buntrock-Marxen-Biber, der bei einer dreitägigen Suche auf einem Hochgeschwindigkeitsrechner gefunden wurde, schreibt 4098 Einsen, bevor er anhält! Im folgenden ist diese Turing-Maschine angegeben, und zwar als Übergangstabelle. Die Schreibweise A0 → B1L bedeutet: „Wenn die Maschine im Zustand A eine 0 liest, gehe in Zustand B, schreibe eine 1 und bewege dich eine Zelle nach links."

A0 → B1L	A1 → A1L
B0 → C1R	B1 → B1R
C0 → A1L	C1 → D1R
D0 → A1L	D1 → E1R
E0 → H1R	E1 → C0R

Allgemein kann man eine beliebige berechenbare Funktion wie 2^n wählen und

$$\Sigma(n) \geq 2^n$$

(für hinreichend großes n) schreiben und völlig darauf vertrauen, daß dies richtig ist. Aber das ist nicht alles.

Der Satz, den wir gerade bewiesen haben, wurde 1962 von Rado gefunden. Er sollte in gewissem Sinn bald von einem wesentlich drama-

tischeren Ergebnis übertroffen werden, das 1964 von M. W. Green gefunden wurde:

Für jede berechenbare Funktion f gilt

$$f(\Sigma(n)) < \Sigma(n+1)$$

für unendlich viele Werte von n.

Wenn wir z.B. als f die Funktion

$$f(m) = m^{\overset{m}{\cdot^{\cdot^{\cdot^{m}}}}}$$

für jede feste Anzahl von Potenzierungen benutzen, müssen wir infolgedessen trotzdem schließen, daß $f(\Sigma(n)) < \Sigma(n+1)$. Mit anderen Worten gilt

$$\Sigma(n+1) > \Sigma(n)^{\overset{\Sigma(n)}{\cdot^{\cdot^{\cdot^{\Sigma(n)}}}}}$$

für unendlich viele Werte von n.

Der Bereich, der am vielversprechendsten für einen Versuch zur Lösung des Busy-beaver-Problems zu sein scheint, ist die Verbesserung der unteren Schranken in der obigen Tabelle. Allerdings hat die in den letzten zehn Jahren durchgeführte Computersuche gezeigt, daß dies keine leichte Aufgabe ist, selbst bei $n = 5$.

Ein Grund für die enormen Schwierigkeiten beim Busy-beaver-Problem liegt in der Fähigkeit relativ kleiner Turing-Maschinen, tiefgehende unbewiesene mathematische Vermutungen zu codieren, wie z.B. Fermats letzten „Satz"[2] oder die Goldbachsche Vermutung (jede gerade Zahl größer als 2 ist die Summe zweier Primzahlen). Das Wissen darüber, ob derartige Maschinen anhalten, entspricht dem Beweisen oder Widerlegen derartiger Vermutungen. Wenn bei einer solchen Vermutung $\Sigma(n)$ für einen Wert n bekannt wäre, dann wären wir (zumindest im Prinzip) in der Lage, über diese Vermutung eine Entscheidung zu treffen, da wir wüßten, wir lange wir warten müssen, bis die Maschine anhält.

Einige Theoretiker bezweifeln sogar, ob wir $\Sigma(6)$ jemals berechnen können.

2 Statt der im angelsächsischen Sprachbereich üblichen Bezeichnung „Fermats letzter Satz" wird im Deutschen „Fermatsche Vermutung" oder „großer Fermatscher Satz" benutzt. A.d.Ü.

Aufgaben

1. Bestimmen Sie die fleißigen Biber mit einem, zwei und drei Zuständen.

2. Bestimmen Sie einen nicht-so-fleißigen Biber, d.h. eine allgemeine Vorschrift für eine Turing-Maschine mit n ($n > 4$) Zuständen, die 2^n Einsen erzeugt, bevor sie anhält.

Literatur

D. Wood. *Theory of Computation.* Harper and Row, New York, 1987.

F. Hennie. *Introduction to Computability.* Addison-Wesley, Reading, Mass., 1977.

D. Wood. *Grammar and L Forms.* Springer, Berlin, 1980.

W. Felscher. *Berechenbarkeit. Theorie der rekursiven und programmierbaren Funktionen.* Springer, Berlin, 1993.