Mitschrift zur Vorlesung
Theoretische Informatik vom 19.01, 26.01 und 02.02.2001 (Abschnitt 7 und 7.1)

Stefan Liske

17.02.2001

Inhaltsangabe

7 Aufzählbarkeit & Entscheidbarkeit

Bisher ist bei der Frage nach der Entscheidbarkeit bekannt, daß z.B. das Selbstanwendungsproblem oder das Halteproblem nicht entscheidbare Probleme sind. Differenziert man diese Problematik, so kann man die einseitige Entscheidbarkeit, die sogenannte Aufzählbarkeit untersuchen. Die Klasse der entscheidbaren Sprachen ist eine echte Teilmenge der Klasse der aufzählbaren Sprachen.

Def. A „aufzählbare Menge“:

Sei X ein endliches Alphabet.
Eine Menge $M \subseteq X^*$ heißt aufzählbar, wenn $M \neq \emptyset$ oder wenn es eine TM $\tau = (S, X, \Gamma, \delta, s_0, b)$ gibt, die folgenden Bedingungen genügt:

1. $X \subseteq \Gamma \wedge b \notin X$
2. τ terminiert für jede Eingabe $|^n$ mit $n \geq 0$
3. $M = \{ Res_\tau(|^n) \mid n \geq 0 \}$
   M sei der Wertebereich der TM τ, stellt also die Vereinigung aller Ergebnisse der TM τ dar. Es ist nicht ausgeschlossen, daß hierbei unterschiedliche Eingaben das gleiche Wort aus M als Ergebnis haben. ($Res_\tau(|^m) = Res_\tau(|^n) \wedge m \neq n$) Dadurch kann diese Definition auch endliche Mengen M beschreiben, indem ab einem bestimmten Index immer wieder die selben Wörter ausgegeben werden. Es bleibt also bei der bisher ausgegebenen (endlichen) Zahl von Wörtern aus M.

Beispiel einer aufzählbaren Menge:

Sei $\tau = (\{s_0,s_1\}, \{|\}, \{|,b\}, \delta, \{s_0,b\})$ mit folgendem δ:

s	x	δ(s,x)
s_0	|	s_1|R
s_0	b	s_0|H
s_1	|	s_0|R
s_1	b	s_1bH

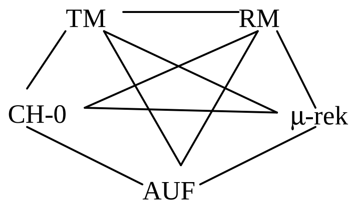
Es ist also $Res_\tau(|^n) = \begin{cases} |^{n+1}, \text{ wenn n gerade} \\ |^n, \text{ wenn n ungerade} \end{cases}$

Für M als Menge aller Ergebnisse von τ ergibt sich mit $M = \{|^{2n+1} \mid n \in \mathbb{N}_0\}$ die Menge aller ungeraden Zahlen, als eine aufzählbare Menge.
In diesem Beispiel kommt jedes Wort $x \in M$ zweimal als Ergebnis der TM τ vor.

Satz B „CH-0⇔Def$_{TM}$⇔Res$_{TM}$⇔RM⇔AUF⇔μ-rek“:

Sei X ein endliches Alphabet.
Eine Menge $M \subseteq X^*$ ist gegeben.
Dann sind folgende Aussagen äquivalent:

i) M ist aufzählbar
ii) M ist Ergebnisbereich einer TM
iii) M ist Haltebereich einer TM
iv) ψ_M ist partiell rekursiv
v) $M \in$ CH-0



Es ergibt sich folgender Ringschluß:
i) ⇒ ii) ⇒ v) ⇒ iv) ⇒ iii) ⇒ i)

Beweis:

„i) ⇒ ii)“: Ist $M=\emptyset$, so wird dieser Fall durch eine prinzipiell nichthaltende TM erledigt. Andernfalls ist dieser Schluß durch die Definition A gegeben.

„ii) ⇒ v)“: Nach Satz 6.Q kann zu jeder TM $\tau = (S, X, \Gamma, \delta, s_0, b)$ eine Grammatik $G_\tau = (N, T, P, \sigma)$ konstruiert werden, mit $L(G_\tau) = \{w\#v \mid Res_\tau(w)=v\}$.
Nun ergänze man G_τ um weitere Regeln zu G_τ‘, so daß der erste Teil von allen Wörter aus $L(G_\tau)$, nämlich die Zeichenkette „w#“ entfernt wird. Es folgt also: $\exists w \in X^*$: $L(G_\tau‘) = \{v \mid Res_\tau(w)=v\}$.
G_τ ist CH-0, G_τ‘ ist es daher auch.

„v) ⇒ iv)“: Dies ist durch Satz 6.Q schon gegeben.

„iv) ⇒ iii)“: Nach der Definition von ψ_M gibt es eine TM, die für alle Wörter $w \in M$ mit der Ausgabe $tt \in X^*$ anhält.

$$\psi_M(w) = \begin{cases} tt, \text{ wenn } w \in M \\ \perp, \text{ wenn } w \notin M \end{cases}$$

„iii) $\Rightarrow$ i)“: Sei $\tau=(S,X,\Gamma,\delta,q_0,b)$ eine TM. Sei $M \subseteq X^*$ der Haltebereich der TM τ: $M = \{v \in X^* \mid Res_\tau(v) \text{ existiert}\}$.

Falls $M=\emptyset$, dann ist M aufzählbar. (als Ergebnisbereich einer nichtterminierende TM)

Sei also $M \neq \emptyset$.

Um M als Ergebnisbereich zu erhalten, betrachte man folgenden Algorithmus:

a) Da $M \neq \emptyset$, kann man ein beliebiges aber festes Wort $z \in M$ wählen, welches immer dann auszugeben ist, wenn der Algorithmus in einem Zustand terminiert, in dem gerade kein gültiges Wort aus M ermittelt wurde. z ist also ein Dummy-Wort.

b) Sei $a_1 \in X$ der erste Buchstabe des Alphabets bezüglich der lexikographischen Reihenfolge[1].

c) Es sei $N:X^* \rightarrow X^*$ die Nachfolgeoperation bezüglich der lexikographischen Ordnung mit $N(\varepsilon)=a_1$. Diese Operation ist durch eine TM realisierbar.

d) Nun geht es um die TM τ‘, welche zu jeder Eingabe $|^n$ das n.te Wort aus der Menge M (dem Haltebereich der gegebene TM τ) liefert.

Anfänglich steht $|^n$ auf dem Band. Nun schreibe τ‘ die Zeichenkette „$\#z\#\varepsilon\beta¢q_0b\$\#a\#$“ dahinter. Hierbei sind #, β, ¢, \$ Begrenzer. ε ist das erste Zeichen aus X*. q_0b ist der Startzustand der TM τ. a sei das erste „echte“ Wort aus X*. Der LSK stehe auf dem ¢. Auf dem Band steht komplett: „$|^n\#z\#\varepsilon\beta¢q_0b\$\#a\#$“.

Allgemeiner läßt sich ein von τ‘ erreichbarer Zwischenzustand auf dem Band darstellen als:

„$|^m\#z\#w_1\beta¢u_1q_1v_1\$w_2\beta¢u_2q_2v_2\$w_3\beta¢u_3q_3v_3\$w_4\beta¢...\beta¢w_s\beta¢u_sq_sv_s\$\#w\#$“

$|^m$ ist die Zahl der übriggebliebenen eingegebenen |. (anfangs noch n Stück)

z ist das Dummy-Wort. (wie zu Anfang auch)

$w_1, w_2, ..., w_s$ sind die noch zu bearbeitenden Wörter mit den zugehörigen (durch Simulation der TM τ) bisher erreichten Konfigurationen $u_1q_1v_1, u_2q_2v_2, ..., u_sq_sv_s$. (anfangs ist dies nur ε mit dem zugehörigen „bisher erreichten“ Startzustand q_0b)

w ist das (n-m).te Wort der Sprache X*. Dies entspricht $N^{n-m}(\varepsilon)$. (anfangs ist es $N(\varepsilon)$)

Diese Konfigurationen $u_iq_iv_i$ der TM τ sind nun nacheinander in die jeweiligen Folgezustände $u_i‘q_i‘v_i‘$ zu überführen. (Diese Folgezustände ergeben sich aus der δ-Tabelle von τ.)

Sollte die TM τ hierbei enden, also einen Haltezustand liefern, so setzte die simulierende TM τ‘ den Folgezustand $q_i‘$ auf ε.

Nachdem alle simulierten TM τ auf dem Band einen Schritt ausgeführt haben, ist das Ende der Bandinschrift „#w#“ durch „$w\beta¢q_0b\$\#N(w)\#$“ zu ersetzen. Das Wort w wird also zum Eingabewort einer neu zu startenden TM τ. Das Nachfolgewort von „w“ wird ermittelt und hinten auf das Band geschrieben. Die TM τ‘ hat nun folgende Bandinschrift:

„$|^m\#z\#w_1\beta¢u_1‘q_1‘v_1‘\$w_2\beta¢u_2‘q_2‘v_2‘\$w_3\beta¢u_3‘q_3‘v_3‘\$w_4\beta¢...\ \beta¢w_s\beta¢u_s‘q_s‘v_s‘\$¢w\beta¢q_0b\$\#N(w)\#$“

Nun sind folgende iterative Schritte auszuführen:

<u>if</u> m==0 <u>then</u> „lösche alles bis auf das z“; <u>stop</u>; *// wenn nötig gib das Dummy-Wort aus*
else m:=m-1; *// streiche ein | weg*
    <u>while</u> $\exists q_i‘=\varepsilon$ <u>do</u> *// solange es noch erreichte Endzustände gibt*
        „nimm das kleinste j mit $q_i‘=\varepsilon$“; *// nimm den kleinsten Index der Endzustände*
        <u>if</u> m==0 <u>then</u> „lösche alles bis auf w_i“; <u>stop</u>; *// wenn erlaubt gib das zugehörige Wort aus*
        <u>else</u> „lösche $w_i\beta¢u_i’q_i’v_i‘\$$ “; m:=m-1; *// andernfalls lösche zugehörige TM und reduziere m um eins*
        <u>fi</u>
    <u>od</u>
    „stelle LSK auf das erste ¢“; *// geh wieder zurück vor die erste TM*
<u>fi</u>

Ist bei der Abarbeitung kein <u>stop</u> erreicht worden so starte die TM τ‘ die ganze Iteration erneut. Es ist also zuerst wieder der Folgeschritt für jede auf dem Band befindliche (da zu simulierende) TM τ auszu-

---

[1] Man beachte, daß „bx“ < „aaa“ gilt, denn die „≤“-Relation der Menge X* ist definiert durch:

1. $a_1 \leq a_2 \leq a_3 \leq ... \leq a_n$, hierbei seien a_1 bis a_n die Buchstaben aus dem Alphabet X
2. fortgesetzt auf Wörter gilt für beliebige $u,v \in X^*$ zunächst bei der „<“-Relation:

$$u<v \Leftrightarrow (\ |u|<|v|\ ) \vee (\ |u| = |v| = n \wedge u = u_1u_2...u_n \wedge v = v_1v_2...v_n \wedge \exists i \in \{1,2,...,n-1\}(u_1=v_1 \wedge u_2=v_2 \wedge ... \wedge u_i=v_i \wedge u_i<v_i)\ )$$

Dies kann leicht auf die „≤“-Relation fortgesetzt werden.

führen. Dabei werden wieder die möglicherweise erreichten Endzustände durch ein „ε“ als Folgezustand markiert.

An diesem „ε“ kann im Anschluß jede gerade angehaltene TM erkannt werden, welche dann wieder gelöscht wird (für den Fall, daß m noch größer als Null ist) oder aber deren zugehöriges erkanntes Wort ausgegeben wird (im Fall, daß m gerade Null geworden ist).

Die TM τ' kann nach der churchschen These[2] simuliert werden. Diese TM τ' wiederum simuliert die Konfigurationsübergänge der TM τ für alle Wörter aus X*.
Wenn ein beliebiges Wort w ein Element aus dem Haltebereich der TM τ ist, so muß die Teilkomponente von τ' welche mit w gestartet wurde auch irgendwann terminieren. Dieses Wort w wird auch bei Eingabe eines ganz bestimmten $|^n$ ausgegeben. $\forall m \in \mathbb{N} \exists n \in \mathbb{N}_0 (Res_{\tau}(|^n)=w_m)$
Es werden allerdings auch nur solche Wörter ausgegeben, für welche die TM τ terminiert. Hierzu gehört auch das häufig ausgegebenen Dummy-Wort z, welches immer dann ausgegeben wird, wenn die Zahl der Striche am Anfang des Bandes (die Anzahl wird in jedem Schritt um mindestens eins reduziert) gerade Null wird, aber keine der simulierten TM τ im gleichen Schritt terminiert hat. Dies ist insbesondere dann der Fall, wenn M eine endliche Menge ist. Ab dem Index n, der mit der Eingabe $|^n$ das letzte Wort aus M als Ausgabe zur Folge hat, wird bei Eingabe von $|^{n+m}$ mit m>0 nur noch z ausgegeben.
Es ist also M = $Res_{\tau'}(|^n)$ mit $n \geq 0$

□

Def. C „entscheidbare Menge“:

Sei X ein endliches Alphabet.
Eine Menge $M \subseteq X^*$ heißt entscheidbar, wenn ihre charakteristische Funktion berechenbar ist.

Eigenschaften aufzählbarer und entscheidbarer Mengen:

Aufzählbarkeit	Entscheidbarkeit
• asymetrisches Phänomen • ist M eine aufzählbare Menge, so gibt es also eine TM welche für alle Wörter aus M hält • Für die Menge $\overline{M}$ gibt es keine haltende TM	• symetrisches Phänomen • für eine entscheidbare Menge M gibt es sowohl haltende TM für Wörter aus M ($w \in M$), als auch für Wörter, die nicht in M ($w \notin M$) liegen.
ψ_M ist die zugehörige partiell charakteristische Funktion	χ_M ist die zugehörige total charakteristische Funktion

Satz D „$M \in ENT \Leftrightarrow M \in AUF \wedge \overline{M} \in AUF$“:

Eine Menge $M \subseteq X^*$ ist genau dann entscheidbar, wenn die Menge M selbst und das Komplement ($\overline{M} = X^* \setminus M$) aufzählbar sind.

Beweis:

„$\Rightarrow$“: Ist M entscheidbar, so ist χ_M mit einer TM τ berechenbar. Nun ist τ so abzuwandeln, daß τ in eine Endlosschleife gerät, wenn $\chi_M(w)$=t. (Hat also die total charakteristische χ_M Funktion ergeben, daß w nicht zu M gehört, so wird nun in einer Endlosschleife weitergearbeitet.) Dies hat zur Folge, daß nur noch ψ_M berechnet wird.
Analog kann die partiell charakteristische Funktion von $\overline{M}$ bestimmt werden, indem die zu χ_M gehörende TM τ endlos weiterarbeitet, wenn $\chi_M(w)$=tt, also w aus M ist.
„$\Leftarrow$“: Idee: Es ist M aufzählbar, also existiert eine TM τ_1 für die partiell charakteristische Funktion von M, um festzustellen, daß ein Wort w aus M ist. M ist Ergebnisbereich der TM τ_1.
Es ist auch $\overline{M}$ aufzählbar, also gibt es ebenso eine TM τ_2 für die partiell charakteristische Funktion von $\overline{M}$, welche entscheidet, ob ein Wort w aus $\overline{M}$ ist. $\overline{M}$ ist Ergebnisbereich der TM τ_2.
Nun bilde man die TM τ, welche die Arbeitsschritte von beiden TM parallel abarbeitet. Da entweder τ_1 oder τ_2 terminieren, wird auch τ terminieren und liefert so entsprechend, ob $w \in M$ oder $w \in \overline{M}$.

□

[2] Der amerikanische Logiker Alanzo Church (1903-1995) formulierte 1936 die nach ihm benannte These:
„Jede im intuitiven Sinne berechenbare Funktion ist turingberechenbar“.

Def./Satz E „ENT, AUF“:

Sei ENT die Klasse aller entscheidbaren und AUF die Klasse aller aufzählbaren Sprachen.
Dann gilt:

1. Jede endliche Sprache ist aufzählbar und entscheidbar.
2. ENT ist abgeschlossen gegen Vereinigung, Durchschnitt, Komplement, *-Bildung und Konkatenation ($\cup$, $\cap$, $\overline{}$, *, $\bullet$)
3. AUF ist ist abgeschlossen gegen Vereinigung, Durchschnitt, *-Bildung und Konkatenation ($\cup$, $\cap$, *, $\bullet$) aber nicht gegen Komplementbildung (siehe Definition).

Def. F „Halteproblemsprache H“:

Sei $X_0 = \{0, 1\}$.
Die Menge $H := \{\, bw_\tau 00u \mid$ τ ist eine TM
$\wedge$ X_0 das zugehörige Eingabealphabet
$\wedge$ bw_τ ist die binäre Standardcodierung von τ
$\wedge$ $u \in X_0^*$
$\wedge$ $Res_\tau(u)$ existiert
$\}$
heißt Halteproblemsprache.
u ist das Wort, für das die TM laufen soll. Diese liegt ebenfalls auf dem Band vor, binär codiert. Dazwischen steht zweimal das Zeichen Null.

Einschub „binäre Standardcodierung“:

Die TM τ mit $\tau = (S, X, \Gamma, \delta, s_0, b)$ ist vorgegeben.
Nun läßt sich τ wie folgt betrachten:

- N sei die Anzahl der Zustände $\Rightarrow$ o.B.d.A. sei $S = \{0, 1, 2, ..., n\text{-}1\}$
- $\Gamma = \{\gamma_1, \gamma_2, ..., \gamma_k\}$
- $X = \{\gamma_{i_1}, \gamma_{i_2}, ..., \gamma_{i_r}\} \subseteq \Gamma$
- δ besteht aus $n \cdot k$ Zeilen zu je 5 Wörtern (s,x,s‘,x‘,Bewegung)
- o.B.d.A.: $s_0=0$, $b=\gamma_k$, $i_1<i_2<...<i_r$

Sei nun $\Gamma_2 = \Gamma \cup \{„,“, „;“\} \cup \{0, 1, 2, ..., n\}$
Folglich ist nun $S \subseteq \Gamma_2$
Nun kann man zu jeder TM τ eindeutig ein kodiertes Wort w_τ zuorden:

$$\Gamma_2^* \ni w_\tau = n;\gamma_{i_1},\gamma_{i_2},...,\gamma_{i_r};\delta,\gamma_1,\delta(0,\gamma_1);\delta,\gamma_2,\delta(0,\gamma_2);...0;...;n\text{-}1,\gamma_k,\delta(\delta(n\text{-}1),\gamma_k)$$

Dies ist die Standardcodierung der TM τ.
Das Wort w_τ ist leicht in ein Wort $bw_\tau \in \{0,1\}^* = X_0^*$ zu überführen. Dies nennt man die binäre Standardcodierung der TM τ.

Lemma G „$H \in AUF \backslash ENT$“:

H ist aufzählbar aber nicht entscheidbar.

Beweis:

Die Nichtentscheidbarkeit ist bereits bekannt. Um die Aufzählbarkeit zu zeigen, muß man eine TM τ‘ konstruieren, deren Haltebereich H entspricht. (siehe Satz B)
Diese TM τ‘ arbeite wie folgt:
Sie prüft zuerst zu jedem $v \in X_0^*$, ob v die Form „$bw_\tau 00u$“ hat. Wenn nicht, dann geht sie in eine Endlosschleife.
Andernfalls ändert sie das Wort „$bw_\tau 00u$“ um in „bw_τ#¢u\$“. Nun simuliert τ‘ schrittweise die Arbeit von τ, indem die δ-Tabelle von τ schrittweise in dem Band von τ(¢u\$) umgesetzt wird.
Falls τ nicht terminiert, dann terminiert auch τ‘ nicht.
Sonst trifft τ‘ irgendwann auf „H“ (also dem Halten der TM τ).
Im Anschluß löscht τ‘ alles bis auf ein Symbol $t \in X_0$.
Folglich ist H Haltebereich der TM τ‘.

□

Def./Satz H „universelle TM“:

Es gibt eine TM τ, die als Eingabe eine beliebige TM τ‘ und ein Wort w akzeptiert und nun die Arbeitsweise von τ‘ auf dem Wort w simuliert. $h(\tau',w) = h_{\tau'}(w)$. τ heißt universelle TM und ist effektiv konstruierbar.

Def. I „Chomsky-Hierarchie“:

Man stellt bei der Betrachtung der bisher bekannten Sprachen fest, daß sie sich hinsichtlich Ihrer Mächtigkeit unterscheiden. Die Beziehungen von Grammatiken zueinander wurden von A. Noam Chomsky genauer untersucht. Erweitert man diese Chomsky-Hierarchie durch die bekannten Modelle erkennender Automaten, so erhält man folgende Übersicht:

Chomsky-Hierarchie:

```
         abzählbare Sprachen
                  |
       CH-0=AUF=TM=RM=µ-rek
                  |
                 ENT
                  |
       CS-1/CS =ERW =NLBA
                  |
           CS-2/CF=NPDA
                /   \
               /     \
           DPDA      LIN
               \     /
                \   /
       CS-3/RL/LL=REG=DA=NA
```

```
              B
hierbei meint |   , daß A echt in B enthalten ist, also A⊂B gilt.
              A
```

CH-0: Sprachen des Typs Chomsky-0. Dies ist die Menge der Sprachen, die durch Grammatiken ohne Einschränkungen erzeugt werden.
AUF: Menge der aufzählbaren Sprachen.
TM: Menge der Sprachen, welche durch TM erkannt werden können.
RM: Menge aller Sprachen, die durch Registermaschinen erkannt werden können.
µ-rek: Dies ist der Wertebereich aller µ-rekursiven Funktionen. Dies sind alle Funktionen, welche durch endlich viele Anwendungen der Erzeugungsverfahren aus den Grundfunktionen hervorgehen.
ENT : Die Menge aller entscheidbaren Sprachen.
CS-1/CS: Sprachen des Typs Chomsky-1. Dies ist die Menge der Sprachen, die durch kontextsensitive Grammatiken erzeugt werden.
ERW: Sprachen vom Erweiterungstyp.
NLBA: Dies sind die Sprachen, welche von nichtdeterministisch linear beschränkten Automaten erkannt werden.
CS-2/CF: Sprachen des Typs Chomsky-2. Dies ist die Menge der Sprachen, die durch kontextfreien Grammatiken erzeugt werden.
NPDA: Menge aller Sprachen, welche von nichtdeterministischen Push-Down-Automaten erkannt werden.
DPDA: Menge der Sprachen, die von deterministischen Push-Down-Automaten erkannt werden.
LIN: Menge aller linearen Sprachen.
CS-3/RL/LL: Sprachen des Typs Chomsky-3. Dies ist die Menge der Sprachen, die durch rechtslineare bzw. linkslineare Grammatiken erzeugt werden.
REG: Menge aller regulären Ausdrücke.
DA: Menge aller Sprachen, die von deterministischen Automaten erkannt werden.
NA: Menge der Sprachen, welche von nichtdeterministischen Automaten erkannt werden.

Beispiele für Sprachen:

$X_0^*\backslash H \in$ abzählbar\CH-0 .... *abzählbar, aber nicht aufzählbar*
$H \in$ CH-0\ENT .... *aufzählbar, aber nicht entscheidbar*
$\{ww \mid w \in X^*\} \in$ CH-1\CH-2............ *kontextsensitiv, aber nicht kontextfrei*
$\{ww^R vv^R \mid w,v \in X^*\} \in$ CH-2\(DPDA∪LIN)............ *kontextfrei, aber weder linear noch durch DPDAs akzeptierbar*
$\{w \mid |w|_0 = |w|_1\} = \{w \mid \#_0 w = \#_1 w\} \in$ DPDA\LIN............ *ist durch DPDAs akzeptierbar, aber nicht linear*
$\{ww^R \mid w \in X^*\} \in$ LIN\ DPDA............ *ist linear, aber nicht durch DPDAs akzeptierbar*
$\{wcw^R \mid w \in X^*\}, \{o^n 1^n \mid n \geq 0\} \in$ (DPDA∪LIN)\CH-3.... *sind sowohl linear als auch durch DPDAs akzeptierbar, aber kein regulärer Ausdruck*

7.1 Bussy Beaver

Def. K „Bussy-Beaver-TM“:

a) eine TM β heißt Bussy-Beaver-Turingmaschine (BBT) mit n Zuständen, wenn $\beta = (S, \{|\}, \{|, b\}, \delta, 1, b)$ mit $S = \{1, 2, 3, ..., n\}$.
Hierfür gibt es folgenden Kurzbezeichnung: „BBT(n)“.

b) Die Abbildung $BB:\mathbb{N}\rightarrow\mathbb{N}$ mit $BB(n) = \max\{ anz(\beta) \mid \beta \text{ ist } BBT(n) \text{ und } \beta \text{ hält an } \}$ heißt Bussy-Beaver-Funktion (BB).
$anz(\beta)$ meint die Anzahl der Striche, die β auf das leere Band schreibt.

Beispiel einer BBT:

n = 2:

s	x	δ(s,x)
1	b	2|R
1	|	2|L
2	b	1|L
2	|	2|H

Man erkennt hier, daß $BB(2) \geq 4$

Beschäftigt man sich weiter mit diesem Thema, so stellt man folgendes fest:

n	1	2	3	4	5	6
BB(n)	1	4	6	13	≥4098	$\geq 6{,}4 \cdot 10^{462}$

Es stellt sich nun die Frage: „Wieviele BBT(n) gibt es?“

Betrachtet man die Regeln einer BBT, so erkennt man folgende Gesetzmäßigkeiten:

	s	x	δ(s,x)
erste Zeile:	1	b	2|R
Haltezeile:	s	a	s|H
restliche Zeilen:	s	a	s’a’B

mit $B\in\{L,R\}$, $a,a'\in\{|,b\}$ und $s,s'\in\{1,2,3,...,n\}$

Folglich gilt für die Anzahl der BBT durch die Kombinationsmöglichkeiten der Zustände:

$BBT(n) = (n\cdot 2\cdot 2)^{2n-2}\cdot (2n-1) = (4n)^{2n-2}\cdot (2n-1)$

$(4n)^{2n-2}$ sind die Anzahl der Regelmöglichkeiten der restlichen Zeilen. 2n Kombinationsmöglichkeiten bleiben für die Restzustände, da aber bei der Startzeile keine Kombinationsmöglichkeiten vorliegt (sie ist bei allen BBT vorgegeben), wird eins abgezogen.

Daraus resultiert folgende Tabelle:

n:	2	3	4	5
	192	1245	$117\cdot 10^6$	$230\cdot 10^9$

Satz L „BB$\notin\mu$-rek“:

BB ist nicht partiell-rekursiv, also nicht berechenbar.
(Anschaulich: BB wächst schneller als jede berechenbare Funktion)

Beweis:

Angenommen BB wäre berechenbar, dann gibt es zu jeder berechenbaren Funktion f ein $n_0\in\mathbb{N}$, so daß für alle $n\geq n_0$ gilt: $BB(n) > f(n)$.

Wenn BB(n) berechenbar ist, dann gibt es eine TM τ mit $|^n\rightarrow|^{BB(n)}$.

Sei f eine beliebige berechenbare Funktion und sei F(x) definiert durch:

$$F(x) = \sum_{i=0}^{x}(f(i)+i^2)$$

Es ist offensichtlich, da f berechenbar ist, daß F berechenbar ist.
Sei nun τ_F eine TM, die F berechnet, d.h. sie produziert angesetzt auf $|^x$ die Inschrift $|^x b|^{F(x)}$. τ_F habe n Zustände.
Nun betrachte TM τ‘, die mit x Zuständen zuerst $|^x$ auf das Band schreibt, und anschließend τ_F zweimal hintereinander startet. Das Ergebnis auf dem Band ist nun: $|^x b|^{F(x)} b|^{F(F(x))}$.
τ‘ hat x+2n Zustände. τ‘ ist eine BBT(x+2n) mit x+F(x)+F(x) Strichen.

- Also gilt: $BB(x+2n) \geq x+F(x)+F(F(x))$
- Desweiteren gibt es eine Konstanten c ab der für alle $x \geq c$ gilt: $x^2 > x+2n$.
  Außerdem ist offensichtlich: $F(x) > x^2$.
  Es gilt also ab $x \geq c$: $F(x) > x+2n$.
- Ferner gilt: $F(x) > F(y)$ wenn $x > y$.
  Es gilt also ab $x \geq c$: $F(F(x)) > F(x+2n)$.

Insgesamt folgt also ab $x \geq c$: $BB(x+2n) \geq x+F(x)+F(F(x)) > F(F(x)) > F(x+2n) > f(x+2n)$.
Ab dem Index c ist BB (als berechenbar angenommen) größer als jede berechenbare Funktion f. Dieser Schluß ist falsch, also stimmt die Voraussetzung nicht. Demnach kann BB nicht berechenbar sein.

$\square$

Quellenangabe

- Vorlesung vom 19.01.2001, 26.01.2001 und 02.02.2001
  Prof. Dr. Andreas Schwill

- Schüler Duden „Informatik“
  Dudenverlag

- „Einführung in die Automatentheorie, Formale Sprachen, und Komplexitätstheorie“
  John E. Hopcroft, Jeffrey D. Ullman
  Oldenbourg Verlag München Wien

- K. Erk, L. Priese
  "Theoretische Informatik"
  Springer Verlag

- viele weitere Seiten im Internet
  von diversen Universitäten