Automatentheorie und Formale Sprachen

– LV 4110 –

Turingmaschinen und Kontextsensitive Sprachen

- Kennenlernen der Zusammenhänge zwischen unterschiedlichen Sprach-Typen
- Untersuchung der Sprache $L(G) = \{a^n b^n c^n \mid n = 0, 1, 2, \ldots\}$
- Definition eines Maschinenmodells zur allgemeinen Beschreibung von Algorithmen
- Kennenlernen der Arbeitsweise einer Turingmaschine
- Anwendung der Technik des Zusammensetzens elementarer TM-Operationen
- Kennenlernen der Turingmaschine als Akzeptor
- Identifizierung des Halteproblems bei einer beschränkten Bandlänge

V. Turingmaschinen und Kontextsensitive Sprachen

1. Kontextsensitive Sprachen
2. Die Sprache vom Typ 0
3. Turingmaschinen
   - 3.1 Modell einer Turingmaschine
   - 3.2 Arbeitsweise einer Turingmaschine
   - 3.3 Beispiele für elementare Operationen einer TM
   - 3.4 Turingmaschinen als Akzeptoren
   - 3.5 Turingmaschinen zur Funktionsberechnung
   - 3.6 Linear beschränkte Automaten (LBA) und Typ-1-Sprachen

V. Turingmaschinen und Kontextsensitive Sprachen

1. Kontextsensitive Sprachen

2. Die Sprache vom Typ 0

3. Turingmaschinen

   3.1 Modell einer Turingmaschine

   3.2 Arbeitsweise einer Turingmaschine

   3.3 Beispiele für elementare Operationen einer TM

   3.4 Turingmaschinen als Akzeptoren

   3.5 Turingmaschinen zur Funktionsberechnung

   3.6 Linear beschränkte Automaten (LBA) und Typ-1-Sprachen

Wir erinnern uns:

Chomsky-Grammatiken vom **Typ 1** haben die Form:

$$\alpha A \beta \rightarrow \alpha \gamma \beta \quad \text{mit} \quad \gamma \neq \varepsilon$$

$$A \in N; \qquad \alpha, \beta, \gamma \in (N \cup T)^*$$

mit der Ausnahme, dass

$$S \rightarrow \varepsilon$$

dazugehören darf, wenn **S** in keiner Regel auf der rechten Seite auftritt.

Gedankenexperiment:

Wählt man: $\alpha, \beta = \varepsilon$ $\Rightarrow$ $A \rightarrow \gamma$ (entspr. Typ 2)

Folge: Die Regeln der **kontextsensitiven** Grammatik gehen in solche einer **kontextfreien** Grammatik über.

Da es zu einer **kontextfreien** Grammatik außerdem immer eine **äquivalente ε-freie** Grammatik gibt, folgt weiter, dass die Menge der **kontextsensitiven Sprachen** die Menge der **kontextfreien Sprachen** umfaßt und damit eine echte **Obermenge** darstellt.

Definitionen für eine monotone Chomsky-Grammatik:

Eine Chomsky-Grammatik $G = (\mathbf{N}, \mathbf{T}, \mathbf{P}, S)$ heißt **monoton**, wenn sie – abgesehen von der Regel $\mathbf{S} \rightarrow \varepsilon$ – nur Regeln der Form

$$\varphi \rightarrow \gamma \qquad \text{mit} \qquad |\varphi| \leq |\gamma|\,; \qquad \varphi, \gamma \in (\mathbf{N} \cup \mathbf{T})^*$$

hat.

Satz:

Jede kontextsensitive Grammatik ist auch monoton und zu jeder monotonen gibt es eine **äquivalente**, kontextsensitive Grammatik (vgl. Beispiel).

<u>Beispiel</u>: $\Rightarrow$ Monotone Grammatik für $L = \{a^n b^n c^n \mid n \in \mathbb{N}_0\}$

Wir betrachten die Grammatik G = (**T**, **N**, **P**, S) mit dem Startsymbol S sowie **T** = { a, b, c }, **N** = { A, C, S} und den Produktionen **P**:

P = {	S $\Rightarrow$ aAbc | abc ,	(1, 1')
	A $\Rightarrow$ aAbC | abC ,	(2, 2')
	Cb $\Rightarrow$ bC ,	(3)
	Cc $\Rightarrow$ cc }	(4)

- G ist monoton.
- G ist <u>nicht</u> **kontextsensitiv**, weil (3) nicht die Form $\alpha A \beta \rightarrow \alpha \gamma \beta$.

$S \Rightarrow aAbc \mid abc\ (1)$, $A \Rightarrow aAbC \mid abC\ (2)$, $Cb \Rightarrow bC\ (3)$, $Cc \Rightarrow cc\ (4)$

Ableitung von $w = a^3b^3c^3$:

$S \Rightarrow_{(1)} a\underline{A}bc \Rightarrow_{(2)} aa\underline{A}bCbc \Rightarrow_{(2')} aaab\underline{Cb}Cbc \Rightarrow_{(3)} aaabbC\underline{Cb}c$

$\Rightarrow_{(3)} aaabb\underline{Cb}Cc \Rightarrow_{(3)} aaabbbC\underline{Cc} \Rightarrow_{(4)} aaabbb\underline{Cc}c$

$\Rightarrow_{(4)} aaabbbccc = a^3b^3c^3$

- G ist zwar monoton, aber <u>nicht</u> kontextsensitiv.
- Jetzt eine zu G **äquivalente kontextsensitive** Grammatik G'.
- Dazu muss Regel (3) geändert werden.

$S \Rightarrow aA\mathbf{B}c \mid abc\ (1)$, $A \Rightarrow aA\mathbf{B}C \mid abC\ (2)$, $Cc \Rightarrow cc\ (4)$

$CB \Rightarrow HB\ \ (3)$, $HB \Rightarrow HC\ \ (3')$, $HC \Rightarrow BC\ \ (3'')$, $B \Rightarrow b\ \ (3''')$

Ableitung von $w = a^3b^3c^3$:

$$S \Rightarrow_{(1)} a\underline{A}Bc \Rightarrow_{(2)} aa\underline{A}BCBc \Rightarrow_{(2')} aaabCB\underline{CB}c \Rightarrow_{(3)} aaabCB\underline{HB}c$$

$$\Rightarrow_{(3')} aaabCB\underline{HC}c \Rightarrow_{(3'')} aaab\underline{CB}BCc \Rightarrow_{(3)} aaab\underline{HB}BCc$$

$$\Rightarrow_{(3')} aaab\underline{HC}BCc \Rightarrow_{(3'')} aaabB\underline{CB}Cc \Rightarrow_{(3)} aaabB\underline{HB}Cc$$

$$\Rightarrow_{(3')} aaabB\underline{HC}Cc \Rightarrow_{(3'')} aaab\underline{B}BCCc \Rightarrow_{(3''')} aaabb\underline{B}CCc$$

$$\Rightarrow_{(3''')} aaabbbC\underline{Cc} \Rightarrow_{(4)} aaabbb\underline{Cc}c \Rightarrow_{(4)} aaabbbccc = a^3b^3c^3$$

- G ist jetzt monoton <u>und</u> kontextsensitiv, d. h. L(G) vom **Typ 1**.

V. Turingmaschinen und Kontextsensitive Sprachen

1. Kontextsensitive Sprachen
2. **Die Sprache vom Typ 0**
3. Turingmaschinen
   - 3.1 Modell einer Turingmaschine
   - 3.2 Arbeitsweise einer Turingmaschine
   - 3.3 Beispiele für elementare Operationen einer TM
   - 3.4 Turingmaschinen als Akzeptoren
   - 3.5 Turingmaschinen zur Funktionsberechnung
   - 3.6 Linear beschränkte Automaten (LBA) und Typ-1-Sprachen

Allgemeine Sprachen

<u>Wir erinnern uns</u>:
Chomsky-Grammatiken vom **Typ 0** haben die Form:

$$\alpha \rightarrow \beta \qquad \text{mit} \qquad \alpha, \beta \in (\mathbf{N} \cup \mathbf{T})^*$$

ohne sonstige Einschränkungen.

<u>Anmerkung</u>:

- ∀ Sprachen vom Typ > 0 sind auch vom Typ = 0.
- Dass auch Sprachen existieren, die vom Typ = 0 sind, aber nicht vom Typ > 0, ist **bisher** nur über einen rein mathematischen Existenzbeweis nachvollziehbar (→ ein explizites Beispiel **bislang** nicht bekannt).
- Bei jeder Konstruktion einer Grammatik vom Typ = 0 hat sich bisher gezeigt, dass die zugehörige Sprache auch von einer Typ-1-Grammatik erzeugt werden kann.

Die Familie L1 der kontextsensitiven Sprachen ist eine echte Obermenge der kontextfreien Sprachen L2.

Die Familie L0 der allgemeinen Sprachen ist eine echte Obermenge der kontext-sensitiven bzw. monotonen Sprachen L1.

V. Turingmaschinen und Kontextsensitive Sprachen

1. Kontextsensitive Sprachen
2. Die Sprache vom Typ 0
3. **Turingmaschinen**
   - 3.1 Modell einer Turingmaschine
   - 3.2 Arbeitsweise einer Turingmaschine
   - 3.3 Beispiele für elementare Operationen einer TM
   - 3.4 Turingmaschinen als Akzeptoren
   - 3.5 Turingmaschinen zur Funktionsberechnung
   - 3.6 Linear beschränkte Automaten (LBA) und Typ-1-Sprachen

- Wir wollen nun **kontextsensitive** (Typ 1) und **allgemeine** (Typ 0) Sprachen betrachten.
- Wir suchen nach einem **Maschinenmodell**, das diese **beiden** Sprachen beschreiben kann.

$\Rightarrow$ Es muss offensichtlich allgemeiner sein, als der **KA**, dessen wesentliche Beschränkung die Zugriffsmöglichkeit auf den Kellerspeicher darstellt, d. h. jeweils nur auf das **oberste** Zeichen des Kellers zugreifbar.

Alan M. Turing (engl. Mathematiker und Kryptoanalytiker) beschrieb 1936 ein nach ihm benanntes Maschinenmodell – sog. **Turingmaschine** – im Zusammenhang mit der **Berechenbarkeit von Funktionen** und der Frage, was man unter einem **Algorithmus** zu verstehen hat.

(1912 – 1954)

Alan Mathison Turing

- britischer Mathematiker und Kryptoanalytiker (Bletchley Park, 1943)
- einflussreichster Theoretiker der Computerentwicklung (Colossus)
- legte die theoretischen Grundlagen der frühen Informatik (Berechen- und Entscheidbarkeit)
- maßgeblich an der Entzifferung von Enigma-verschlüsselten Funksprüchen beteiligt

Von 1945 bis 1948 im National Physical Laboratory, Teddington, tätig am Design der **A**utomatic **C**omputing **E**ngine (ACE)

Übersicht:

Verschlüsselungsgerät der Deutschen Wehrmacht im Zweiten Weltkrieg (1939 – 1945)

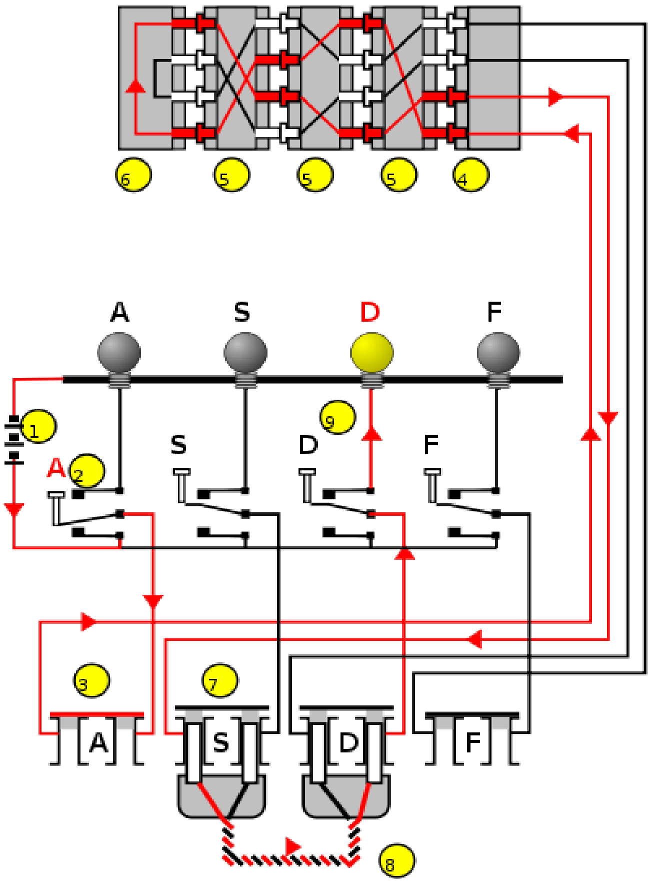
Two Design Principles for secure ciphers:

- Confusion

  The ciphertext statistics should depend on the plaintext statistics in a manner **too complicated** to be exploited by the enemy cryptanalyst.

- Diffusion

  Each digit of the plaintext (and/or secret key) should influence **many digits** of the ciphertext.



Komponenten und Funktionsweise:

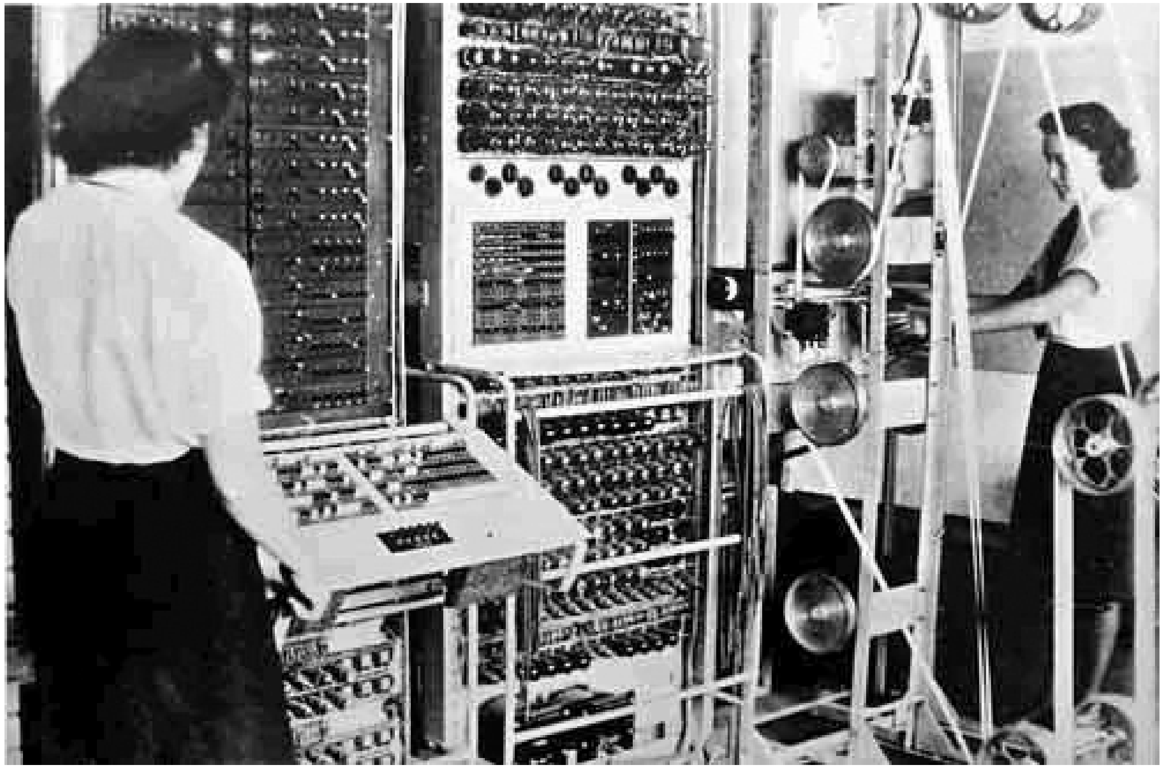
1	Batterie
2	Tastatur
3,7	Steckerbrett
8	Steckkabel
5	Walzensatz (dreht sich)
4	Eintrittswalze (fest)
6	Umkehrwalze (fest)
9	Lampenfeld

Übersicht:

Electronic **N**umerical **I**ntegrator **a**nd **C**omputer (**ENIAC**), USA – 1946

- erste rein elektronische Universal-rechner der U.S. Army
- entwickelt ab 1942 bis 1946 an der University of Pennsylvania
- benutzt Elektronenröhren zur Repräsentation von Zahlen und elektrische Pulse für deren Übertragung

Funktionen: add, sub, mult, div, sqrt



Übersicht:

COLOSSUS, UK – 1943

- Röhrenrechner (Computer der ersten Gen.)
- Militärischer Einsatz im Zweiten Weltkrieg
- Erzeugung von Zufallszahlen
- Zum Chiffrieren Bildung von XOR

Entwurf in Bletchley-Park (1943) nach der Idee von Alan M. Turing

Z3 (Konrad Zuse, 1941, Berlin)

- **erster funktionsfähiger Digitalrechner** (Universalrechner) weltweit
- in **elektromagnetischer Relaistechnik** aufgebaut 600 Relais (RW) und 1400 Relais (SW)
- verwendete eine **binäre Gleitkommaarithmetik**
- erst 1998 fand man heraus, dass die Z3 der Definition eines **turingmächtigen Computers** genügt
- sie wurde 1944 bei einem Bombenangriff zerstört

Ein **funktionsfähiger Nachbau** (Deutsches Museum in München)

Eigenschaften der ersten Computer

Computer-Name	Land	Jahr	GkA[1)]	binär	elektro-nisch	program-mierbar	Turing-fähig
Zuse Z3	DE	1941	ja	ja	nein	ja, (Lochst)	ja
Colossus	UK	1943	nein	ja	ja	teilweise	nein
ENIAC	USA	1946	nein	nein	ja	teilweise	ja

1) **G**leit**k**omma-**A**rithmetik

V. Turingmaschinen und Kontextsensitive Sprachen

1. Kontextsensitive Sprachen
2. Die Sprache vom Typ 0
3. Turingmaschinen

   3.1 Modell einer Turingmaschine

   3.2 Arbeitsweise einer Turingmaschine

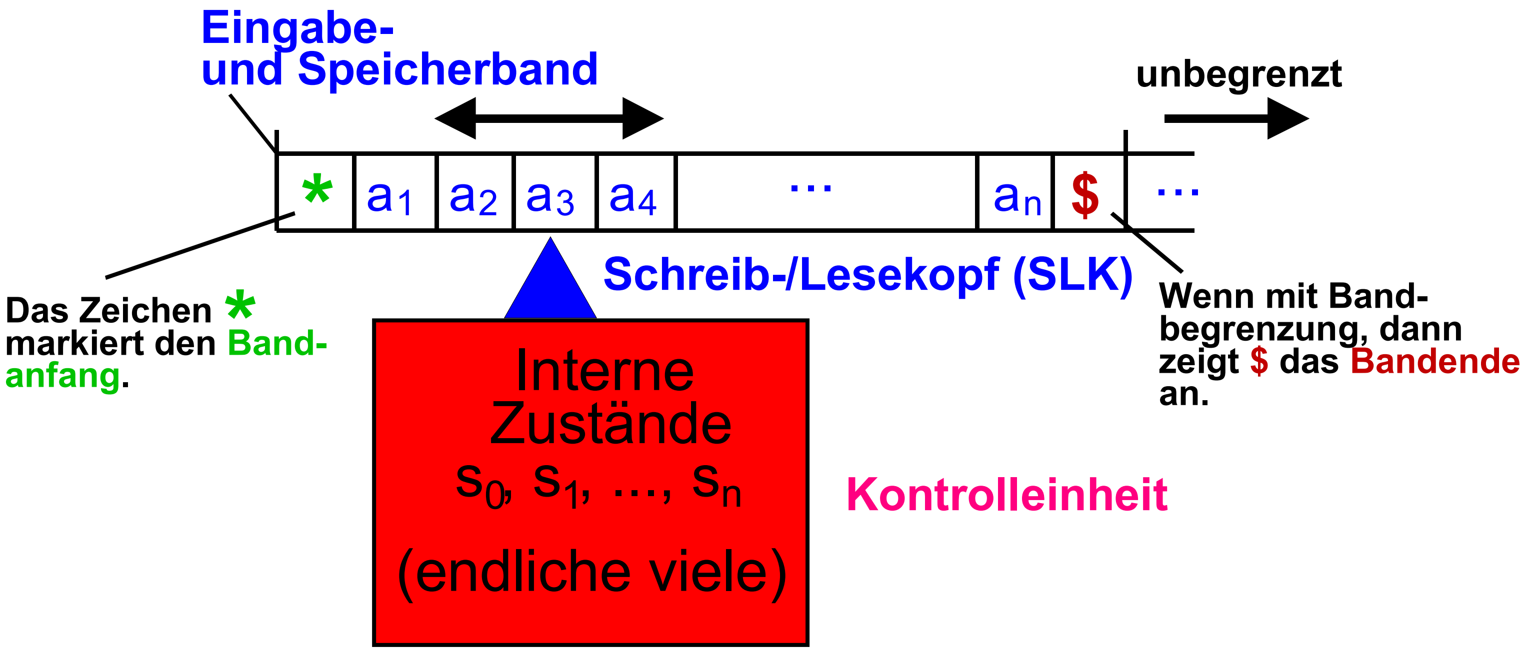
   3.3 Beispiele für elementare Operationen einer TM

   3.4 Turingmaschinen als Akzeptoren

   3.5 Turingmaschinen zur Funktionsberechnung

   3.6 Linear beschränkte Automaten (LBA) und Typ-1-Sprachen

Turingmaschine



- Turingmaschine besteht aus einer **Kontrolleinheit** mit einem **Schreib-/Lesekopf (SLK)**
- sowie aus einem **einseitig unbegrenzten** Speicherband (→ Eingabe und Ausgabe).

Funktion des SLK:

- Mit dem **SLK** kann die Maschine **genau ein** Zeichen lesen bzw. überschreiben.
- Der **SLK** kann um eine Position nach links oder nach rechts gerückt werden.

⇒

Die wesentliche Eigenschaft einer **TM** ist, dass **jedes** Feld des Eingabe- und Speicherbandes **gelesen** und **verändert** werden kann.

Damit entspricht das Eingabe- und Speicherband dem Hauptspeicher eines modernen Rechners.

V. Turingmaschinen und Kontextsensitive Sprachen

1. Kontextsensitive Sprachen
2. Die Sprache vom Typ 0
3. Turingmaschinen
   - 3.1 Modell einer Turingmaschine
   - **3.2 Arbeitsweise einer Turingmaschine**
   - 3.3 Beispiele für elementare Operationen einer TM
   - 3.4 Turingmaschinen als Akzeptoren
   - 3.5 Turingmaschinen zur Funktionsberechnung
   - 3.6 Linear beschränkte Automaten (LBA) und Typ-1-Sprachen

Arbeitsweise der TM:

Der ***typische elementare Arbeitsschritt*** einer Turingmaschine besteht darin, das Zeichen eines Arbeitsfeldes zu lesen, in Abhängigkeit vom eingelesenen Zeichen und dem gegenwärtigen Maschinenzustand das gleiche oder ein anderes Zeichen in das Arbeitsfeld zu schreiben und anschließend gegebenenfalls auf ein Nachbarfeld zu positionieren.

Definition:

TM = (**S**, s_0, **F**, Σ, **B**, δ) bezeichnet eine (deterministische) Turingmaschine, wenn für die einzelnen Komponenten gilt:

- **S** endliche Menge der ***internen*** Zustände der Maschine
- s_0 interner Anfangszustand, $s_0 \in$ **S**
- **F** Menge der internen Endzustände, **F** $\in$ **S**
- Σ endliche Menge der Eingabezeichen
- **B** endliche **Menge der Bandzeichen** (inkl. Σ und eines Leerzeichens *, das nicht als Eingabezeichen eines Wortes auftritt)
- δ (*deterministische*) Überführungsfunktion δ: **S** x **B** $\rightarrow$ **S** x **B** x **X**, wobei **X** = {l, r, h} die möglichen Bewegungen des SLK darstellt

Erklärung zur Überführungsfunktion einer TM:

$$\delta\text{: } \mathbf{S} \times \mathbf{B} \rightarrow \mathbf{S} \times \mathbf{B} \times \mathbf{X}$$

Die Zuordnung (s, b) → (s‘, b‘, **X**) mit **X** ∈ {l, r, h} bedeutet, dass – falls sich die Maschine im aktuellen Zustand s befindet und das Feld unter dem SLK mit dem Zeichen b beschriftet ist – sie in den Zustand s‘ übergeht, b durch b‘ ersetzt wird und der SLK entweder um eine Position nach links (**X** = l) bzw. nach rechts (**X** = r) geht oder an der aktuellen Position (**X** = h) verharrt.

⇒ Die Tabelle von δ wird auch als ***Maschinentafel*** oder als ***Programm*** der Turingmaschine bezeichnet.

Turingmaschine

Zustands- bzw. Maschinentafel

	b_1	b_2	b		...		b_m
S_0							
S_1							
...							
s			(s', b', X)				
...							
S_n							

- SLK zu Anfang ganz links auf dem ersten Feld des Bands → i. a. *
- TM hält an, falls δ für das aktuelle Paar (s, b) nicht definiert ist

Beispiel: $\Rightarrow$ Turing-Maschine **TM** = (**S**, S_0, **F**, Σ, **B**, δ) zum Löschen des Speichers

Für

$$\Sigma = \{0, 1\}\ ,\ \mathbf{B} = \{0, 1, *\}\ ,\ \mathbf{S} = \{s_0, s_1\}\ \text{ und }\ \mathbf{F} = \{s_1\}$$

ist eine **TM** gesucht, deren SLK nach rechts zum nächsten Leerzeichen (*) läuft und dabei alle Felder löscht, d. h. 0 oder 1 jeweils durch ein Leerzeichen * ersetzt.

Maschinentafel:

δ	**0**	**1**	*****
$\mathbf{S_0}$	(S_0, *, r)	(S_0, *, r)	(S_1, *, h)

Funktionsweise:

t = 0:

*	**0**	**1**	*	*	…

Bandanfang —↑   ↑ S_0 ↑ = SLK

t = 1:

*	*	**1**	*	*	…

↑ S_0

t = 2:

*	*	*	*	*	…

Endkonfiguration   ↑ S_1

V. Turingmaschinen und Kontextsensitive Sprachen

- Ein Zeichen nach rechts (**r**):

  $\delta(s_0, x) = (s_f, x, r)$ für beliebiges Bandzeichen x.

- Linkes Wortende suchen (**L**):

  $\delta(s_0, x) = (s_0, x, l)$ für $x \neq *$; $\delta(s_0, *) = (s_f, *, h)$

- Aktuelles Zeichen a auf erstes Freizeichen rechts verschieben (**VR**): (Für jedes Eingabezeichen $a \in \Sigma$ wird dazu ein innerer Zustand s_a benötigt).

$\delta(s_0, a) = (s_a, *, r)$; $\delta(s_a, x) = (s_a, x, r)$ für $x \neq *$; $\delta(s_a, *) = (s_f, a, h)$

- Aktuelles Zeichen a auf erstes Freizeichen links kopieren (**KL**):

$\delta(s_0, a) = (s_a, a, l)$; $\delta(s_a, x) = (s_a, x, l)$ für $x \neq *$; $\delta(s_a, *) = (s_f, a, h)$

- Bedingtes Anhalten in Abhängigkeit vom Feldinhalt * (**B(*)**):

$\delta(s_0, x) = (s_{f1}, x, h)$ für $x \neq *$; $\delta(s_0, *) = (s_{f2}, *, h)$

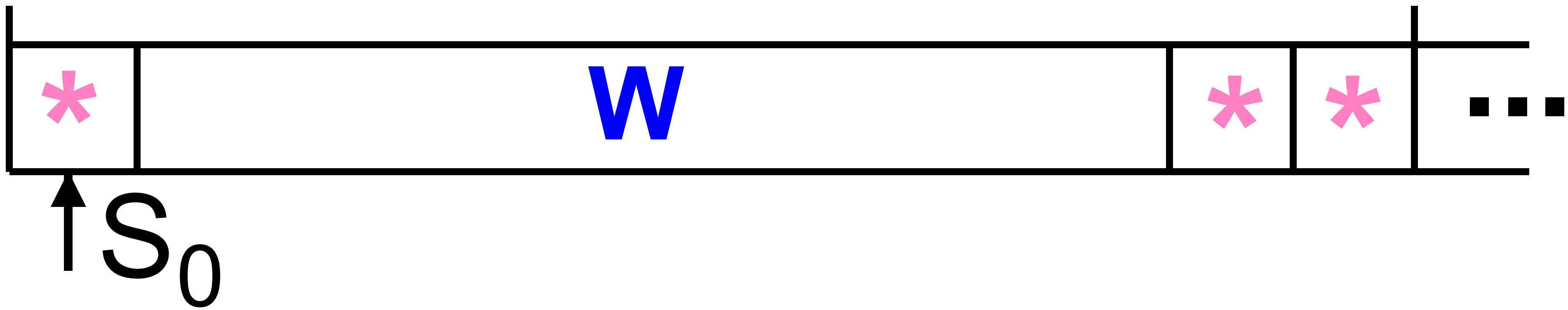
- Zusammengesetzte Turingmaschine zum Kopieren eines Wortes:
  (SLK stehe im Ausgangszustand s_0 auf dem Freizeichen links des Eingabewortes.)

```
r  ----→   VR-VR-KL-VL-r-B(*)  ----→  ende
     |                         |
     ---------------←-----------
```

V. Turingmaschinen und Kontextsensitive Sprachen

Ausgangslage (Initialkonfiguration):

In der Ausgangssituation möge ein Wort w aus Σ*, eingegrenzt durch zwei Leerzeichen, auf dem Band stehen. Die Maschine sei im Anfangszustand s_0 und der SLK stehe auf dem Leerzeichen am Anfang (des Bandes).



Definition (Sprache einer TM = (**S**, s_0, **F**, Σ, **B**, δ)):

Das Wort w wird von der Turingmaschine TM ***akzeptiert***, wenn nach einer Folge von Zwischenschritten eine Endsituation entsteht, bei der sich die TM in einem **internen Endzustand** s $\in$ **F** befindet **und** der SLK wieder auf dem Leerzeichen am Anfang des Bandes steht. Auf den in der Endsituation vorhandenen Bandinhalt kommt es **nicht** an. Unter der *Sprache **L(TM)** einer Turingmaschine **TM*** versteht man alle Worte, die von der TM akzeptiert werden.

Endzustand:

Beispiel: $\Rightarrow$ Turing-Maschine **TM** = (**S**, S_0, **F**, Σ, **B**, δ) zum Erkennen der Sprache $L = \{ a^n b^n c^n \mid n = 1,2,\ldots\}$

hier:
n = 3

*	**a**	**a**	**a**	**b**	**b**	**b**	**c**	**c**	**c**	*	**...**

↑ S_0

Grundidee:

↑ = **SLK**

- am weitesten links stehendes **a** markieren $\Rightarrow$ durch **A** ersetzen
- das erste **b** suchen und durch **B** ersetzen
- wenn dies erfolgreich, dann erstes **c** suchen und durch **C** ersetzen
- danach läuft **SLK** zurück und Vorgang beginnt von neuem

$\Rightarrow$ Wird kein **a** mehr gefunden, so darf auch kein **b** und kein **c** mehr auf dem Band stehen.

hier:
n = 3

*	a	a	a	b	b	b	c	c	c	*	...

↑ S_0

Umsetzung:

- S_0: Anfangszustand ⇒ **SLK** auf *
- S_1: **SLK** erwartet ein **a**, dann ersetze **a** ⇒ **A**, oder **SLK** liest **B** ⇒ kein **a** mehr vorhanden
- S_2: **SLK** geht nach rechts, um erstes **b** zu finden, dann ersetze **b** ⇒ **B**
- S_3: analog zu S_2, lediglich mit **c** ⇒ **C**

hier:
n = 3

*	**a**	**a**	**a**	**b**	**b**	**b**	**c**	**c**	**c**	*	**...**

↑ S_0

Umsetzung (Fortsetzung):

- S_4: **SLK** wieder ganz nach links auf das erste **A** (von rechts);
  SLK wird dann auf dem nächsten rechten Feld positioniert
- S_5: prüft, ob kein **b** und kein **c** mehr auf dem Band vorhanden
- S_6: bringt **SLK** wieder in die Ausgangslage
- S_7: einziger Endzustand $\Rightarrow$ **SLK** wieder auf *

$\Rightarrow$ folglich:

hier:
n = 3

*	**a**	**a**	**a**	**b**	**b**	**b**	**c**	**c**	**c**	*	**...**

↑ S_0

Ergebnis:

Turing-Maschine **TM** = (**S**, S_0, **F**, Σ, **B**, δ) mit

S_0 = Anfangszustand

Σ = { **a**, **b**, **c** };

B = { **a**, **b**, **c**, * , **A**, **B**, **C** };

S = { S_0, S_1, ... , S_7 };

F = { S_7 }

Maschinentafel der **TM**:

	a	b	c	A	B	C	*
S_0	-	-	-	-	-	-	(S_1,*, r)
S_1	(S_2,A, r)	-	-	-	(S_5,B, r)	-	-
S_2	(S_2,a, r)	(S_3,B, r)	-	-	(S_2,B, r)	-	-
S_3	-	(S_3,b, r)	(S_4,C, l)	-	-	(S_3,C, r)	-
S_4	(S_4,a, l)	(S_4,b, l)	-	(S_1,A, r)	(S_4,B, l)	(S_4,C, l)	-
S_5	-	-	-	-	(S_5,B, r)	(S_5,C, r)	(S_6,*, l)
S_6	-	-	-	(S_6,A, l)	(S_6,B, l)	(S_6,C, l)	(S_7,*, h)

Konfigurationsfolge für w = abc:

t = 0	* ↑ S_0	a	b	c	*	*	*	...
t = 1	*	a ↑ S_1	b	c	*	*	*	...
t = 2	*	A	b ↑ S_2	c	*	*	*	...
t = 3	*	A	B	c ↑ S_3	*	*	*	...
t = 4	*	A	B ↑ S_4	C	*	*	*	...
t = 5	*	A ↑ S_4	B	C	*	*	*	...

t = 6	*	A	B ↑ S_1	C	*	*	*	...
t = 7	*	A	B	C ↑ S_5	*	*	*	...
t = 8	*	A	B	C	* ↑ S_5	*	*	...
t = 9	*	A	B	C ↑ S_6	*	*	*	...
t = 10	*	A	B ↑ S_6	C	*	*	*	...
t = 11	*	A ↑ S_6	B	C	*	*	*	...

t = 12	* $\uparrow S_6$	**A**	**B**	**C**	*	*	*	...
t = 13	* $\uparrow S_7$	**A**	**B**	**C**	*	*	*	...

$S_7 \in \mathbf{F}$

$\Rightarrow$ <u>w = abc $\in$ L(**TM**)</u>

V. Turingmaschinen und Kontextsensitive Sprachen

Eine Turingmaschine zur Berechnung der Funktion:

$$f(TM) := \begin{cases} 1, & \text{falls } w = 1^n0^n \text{ für } n > 0 \\ 0, & \text{sonst} \end{cases}$$

Anfangskonfiguration (* = Leerzeichen, S_0 = Anfangszustand):

* ↑ S_0	Eingabewort w	*	*	*	...

Endkonfiguration (S_f = Endzustand):

*	*	*	0 / 1 ↑ S_f	*	*	*	*	...

TM = (S, S_0, F, Σ, B, δ)

mit **S** = { S_0, S_1, ..., S_6 }, **F** = { S_6 }, Σ = { 0, 1 } und **B** = { 0, 1, * }:

Maschinentafel:

δ	**0**	**1**	*
S_0	-	-	(S_1, *, r)
S_1	(S_5, 0, r)	(S_2, *, r)	(S_5, 1, r)
S_2	(S_2, 0, r)	(S_2, 1, r)	(S_3, *, l)
S_3	(S_4, *, l)	(S_5, 0, r)	(S_5, 0, r)
S_4	(S_4, 0, l)	(S_4, 1, l)	(S_1, *, r)
S_5	(S_6, *, l)	(S_6, *, l)	(S_6, *, l)

Funktionsberechnung für w = **1100**:

t = 0	* ↑ S_0	1	1	0	0	*	*	...
t = 1	*	1 ↑ S_1	1	0	0	*	*	...
t = 2	*	*	1 ↑ S_2	0	0	*	*	...
t = 3	*	*	1	0 ↑ S_2	0	*	*	...
t = 4	*	*	1	0	0 ↑ S_2	*	*	...
t = 5	*	*	1	0	0	* ↑ S_2	*	...

Turingmaschine

t = 6	*	*	1	0	0 $\uparrow S_3$	*	*	…
t = 7	*	*	1	0 $\uparrow S_4$	*	*	*	…
t = 8	*	*	1 $\uparrow S_4$	0	*	*	*	…
t = 9	*	* $\uparrow S_4$	1	0	*	*	*	…
t = 10	*	*	1 $\uparrow S_1$	0	*	*	*	…
t = 11	*	*	*	0 $\uparrow S_2$	*	*	*	…

t = 12	*	*	*	0	* $\uparrow S_2$	*	*	...
t = 13	*	*	*	0 $\uparrow S_3$	*	*	*	...
t = 14	*	*	* $\uparrow S_4$	*	*	*	*	...
t = 15	*	*	*	* $\uparrow S_1$	*	*	*	...
t = 16	*	*	*	**1**	* $\uparrow S_5$	*	*	...
t = 17	*	*	*	**1** $\uparrow S_6$	*	*	*	...

$\Rightarrow$ Funktionswert f(w = 1100) = 1

Nichtdeterministische Turingmaschinen:

Bei diesen gilt für die Überführungsfunktion

$$\delta: \mathbf{S} \times \mathbf{B} \text{ -> } P(\mathbf{S} \times \mathbf{B} \times \mathbf{X}).$$

Turingmaschinen mit mehreren Bändern:

Bei diesen wird auf mehreren gleichartigen Bändern gearbeitet. Die Überführungsfunktion hat dabei die Form

$$\delta: \mathbf{S} \times \mathbf{B}^k \text{ -> } \mathbf{S} \times \mathbf{B}^k \times \mathbf{X}^k$$

Solche Maschinen arbeiten **effizienter** als eine Turingmaschine mit nur einem Band.

Sätze:

1. Zu jeder als Akzeptor entworfenen nichtdeterministischen Turingmaschine gibt es eine deterministische, die die **gleiche** Sprache akzeptiert.
2. Zu jeder als Akzeptor entworfenen Turingmaschine mit mehreren Bändern gibt es eine mit **nur einem** Band, die die **gleiche** Sprache akzeptiert.
3. Zu jeder Sprache **L(G)** einer Grammatik vom **Typ 0** gibt es eine Turingmaschine **TM** mit **L(G) = L(TM)** und umgekehrt.
4. Es ist **nicht entscheidbar**, ob eine beliebige TM bei der Abarbeitung eines beliebigen Wortes anhält oder nicht (sog. **Halteproblem** bei TM → **Terminierung eines Algorithmus**).

V. Turingmaschinen und Kontextsensitive Sprachen

1. Kontextsensitive Sprachen
2. Die Sprache vom Typ 0
3. Turingmaschinen
   - 3.1 Modell einer Turingmaschine
   - 3.2 Arbeitsweise einer Turingmaschine
   - 3.3 Beispiele für elementare Operationen einer TM
   - 3.4 Turingmaschinen als Akzeptoren
   - 3.5 Turingmaschinen zur Funktionsberechnung
   - **3.6 Linear beschränkte Automaten (LBA) und Typ-1-Sprachen**

Bei **Linear beschränkte Automaten** (LBA) handelt sich um als Akzeptoren entworfene Turingmaschinen, die hinsichtlich ihrer Arbeitsweise einer bestimmten Beschränkung unterliegen: ***Die zur Verfügung stehende Länge des Bandes ist linear abhängig von der Länge des zu untersuchenden Eingabewortes.***

<u>Satz:</u>

Zu jeder Sprache L(G) einer Grammatik vom Typ 1 (monotone Grammatiken) gibt es einen (nichtdeterministischen) LBA mit

L(G) = L(LBA)

und umgekehrt.

Bemerkungen:

Ob ***deterministische*** linear beschränkte Automaten die gleiche Mächtigkeit als Akzeptoren haben wie die ***nichtdeterministischen***, ist eine Frage, die im Gegensatz zu den anderen behandelten Automatentypen bisher **noch nicht** beantwortet werden konnte.

Das ***Halteproblem*** für **LBA** ist entscheidbar, das heißt:

> Für jedes Wort und jeden LBA ist entscheidbar, ob der LBA das Wort akzeptiert oder nicht.