GBI Tutorium Nr. 2^5

Tutorium 12

Dominik Muth - dominik.muth@student.kit.edu | 23. Januar 2013

INSTITUT FÜR INFORMATIK

Outline/Gliederung

1. Wiederholung
2. Turingmaschinen
3. Alan Turing
4. Komplexität
5. Fragen

Überblick

1. Wiederholung
2. Turingmaschinen
3. Alan Turing
4. Komplexität
5. Fragen

Wiederholung - Quiz

- Bei Bäumen ist die Wurzel nicht immer klar
- $\neg(A \Rightarrow B) \Leftrightarrow \neg(\neg A \vee B)$
- Mit rechtslinearen Grammatiken kann man all Probleme lösen, welche man auch durch kontextfreie Grammatiken lösen kann.
- Sei $f = A \rightarrow B$ und $g = B \rightarrow C$
  $\Rightarrow f \circ g = A \rightarrow C$

Wiederholung - Quiz

- Bei Bäumen ist die Wurzel nicht immer klar $\surd$
- $\neg(A \Rightarrow B) \Leftrightarrow \neg(\neg A \vee B)$
- Mit rechtslinearen Grammatiken kann man all Probleme lösen, welche man auch durch kontextfreie Grammatiken lösen kann.
- Sei $f = A \rightarrow B$ und $g = B \rightarrow C$
  $\Rightarrow f \circ g = A \rightarrow C$

Wiederholung - Quiz

- Bei Bäumen ist die Wurzel nicht immer klar $\surd$
- $\neg(A \Rightarrow B) \Leftrightarrow \neg(\neg A \vee B)$ X
- Mit rechtslinearen Grammatiken kann man all Probleme lösen, welche man auch durch kontextfreie Grammatiken lösen kann.
- Sei $f = A \rightarrow B$ und $g = B \rightarrow C$
  $\Rightarrow f \circ g = A \rightarrow C$

Wiederholung - Quiz

- Bei Bäumen ist die Wurzel nicht immer klar $\surd$
- $\neg(A \Rightarrow B) \Leftrightarrow \neg(\neg A \vee B)$ X
- Mit rechtslinearen Grammatiken kann man all Probleme lösen, welche man auch durch kontextfreie Grammatiken lösen kann. X
- Sei $f = A \rightarrow B$ und $g = B \rightarrow C$
  $\Rightarrow f \circ g = A \rightarrow C$

Wiederholung - Quiz

- Bei Bäumen ist die Wurzel nicht immer klar $\surd$
- $\neg(A \Rightarrow B) \Leftrightarrow \neg(\neg A \vee B)$ X
- Mit rechtslinearen Grammatiken kann man all Probleme lösen, welche man auch durch kontextfreie Grammatiken lösen kann. X
- Sei $f = A \rightarrow B$ und $g = B \rightarrow C$
  $\Rightarrow f \circ g = A \rightarrow C$ X

Wiederholung - Allgemein

- Wie ist das Kartesische Produkt definiert?
- Was war eine Schleifeninvariante?
- Was waren Homomorphismen?
- Kodiere das Wort "Bald ist das Semester zu ende"mit der Huffman Kodierung

Wiederholung - Allgemein

- Wie ist das Kartesische Produkt definiert?
- Was war eine Schleifeninvariante?
- Was waren Homomorphismen?
- Kodiere das Wort "Bald ist das Semester zu ende"mit der Huffman Kodierung

Wiederholung - Allgemein

- Wie ist das Kartesische Produkt definiert?
- Was war eine Schleifeninvariante?
- Was waren Homomorphismen?
- Kodiere das Wort "Bald ist das Semester zu ende"mit der Huffman Kodierung

Wiederholung - Allgemein

- Wie ist das Kartesische Produkt definiert?
- Was war eine Schleifeninvariante?
- Was waren Homomorphismen?
- Kodiere das Wort ”Bald ist das Semester zu ende”mit der Huffman Kodierung

Überblick

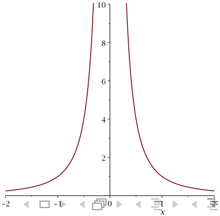
Partielle Funktionen

Definition: Partielle Funktion

Eine partielle Funktion ist eine rechtseindeutige Relation, die nicht zwingend linkstotal ist.
Wir schreiben

$$f : M \dashrightarrow M'$$

Anschaulich:
Funktionen, die an manchen Stellen "Definitionslücken" haben dürfen.
Beispiel:
$\frac{1}{x^2}$ ist eine partielle Funktion ($x = 0$ hat keinen Funktionswert)



Turingmaschine

Definition: Turingmaschine

Eine Turingmaschine T ist definiert durch

$$T = (Z, Z_0, X, f, g, m)$$

Dabei ist

- **Z**: die Zustandsmenge

Turingmaschine

Definition: Turingmaschine

Eine Turingmaschine T ist definiert durch

$$T = (Z, Z_0, X, f, g, m)$$

Dabei ist

- **Z**: die Zustandsmenge
- $\mathbf{z_0}$: der Anfangszustand

Turingmaschine

Definition: Turingmaschine

Eine Turingmaschine T ist definiert durch

$$T = (Z, Z_0, X, f, g, m)$$

Dabei ist

- **Z**: die Zustandsmenge
- $\mathbf{z_0}$: der Anfangszustand
- **X**: das Bandalphabet

Turingmaschine

Definition: Turingmaschine

Eine Turingmaschine T ist definiert durch

$$T = (Z, Z_0, X, f, g, m)$$

Dabei ist

- **Z**: die Zustandsmenge
- $\mathbf{z_0}$: der Anfangszustand
- **X**: das Bandalphabet
- $\mathbf{f} : \mathbf{Z} \times \mathbf{X} \dashrightarrow \mathbf{Z}$: die partielle Zustandsüberführungsfunktion

Turingmaschine

Definition: Turingmaschine

Eine Turingmaschine T ist definiert durch

$$T = (Z, Z_0, X, f, g, m)$$

Dabei ist

- **Z**: die Zustandsmenge
- $\mathbf{z_0}$: der Anfangszustand
- **X**: das Bandalphabet
- $\mathbf{f} : \mathbf{Z} \times \mathbf{X} \dashrightarrow \mathbf{Z}$: die partielle Zustandsüberführungsfunktion
- $\mathbf{g} : \mathbf{Z} \times \mathbf{X} \dashrightarrow \mathbf{g}$: die partielle Ausgabefunktion

Turingmaschine

Definition: Turingmaschine

Eine Turingmaschine T ist definiert durch

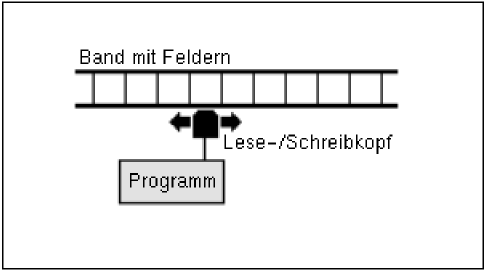
$$T = (Z, Z_0, X, f, g, m)$$

Dabei ist

- **Z**: die Zustandsmenge
- $\mathbf{z_0}$: der Anfangszustand
- **X**: das Bandalphabet
- $\mathbf{f} : \mathbf{Z} \times \mathbf{X} \dashrightarrow \mathbf{Z}$: die partielle Zustandsüberführungsfunktion
- $\mathbf{g} : \mathbf{Z} \times \mathbf{X} \dashrightarrow \mathbf{g}$: die partielle Ausgabefunktion
- $\mathbf{m} : \mathbf{Z} \times \mathbf{X} \dashrightarrow \{-\mathbf{1}, \mathbf{0}, \mathbf{1}\}$: die partielle Bewegungsfunktion

Turingmaschine: Verständnis

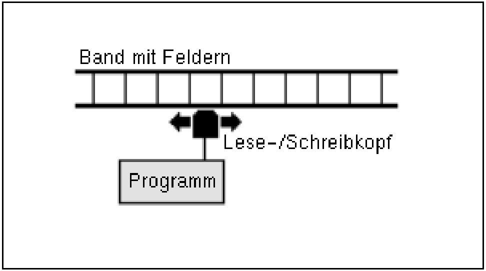
Woher kennen wir ähnliche Funktionen wie f und g?



Turingmaschine: Verständnis

Woher kennen wir ähnliche Funktionen wie *f* und *g*?

Von Automaten und Akzeptoren.

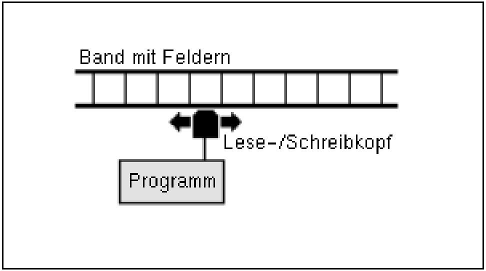


Turingmaschine: Verständnis

Woher kennen wir ähnliche Funktionen wie *f* und *g*?

Von Automaten und Akzeptoren.

Wo war dort der Unterschied?



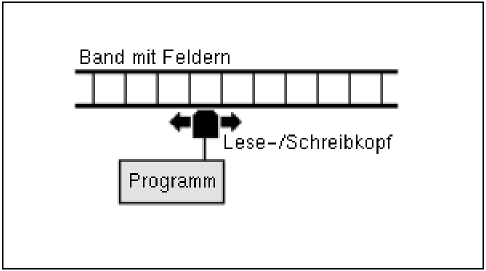
Turingmaschine: Verständnis

Woher kennen wir ähnliche Funktionen wie f und g?

Von Automaten und Akzeptoren.

Wo war dort der Unterschied?

Bei Automaten und Akzeptoren waren die Funktionen nicht partiell.



Turingmaschine: Verständnis

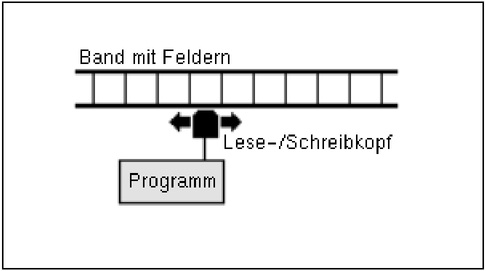
Woher kennen wir ähnliche Funktionen wie f und g?

Von Automaten und Akzeptoren.

Wo war dort der Unterschied?

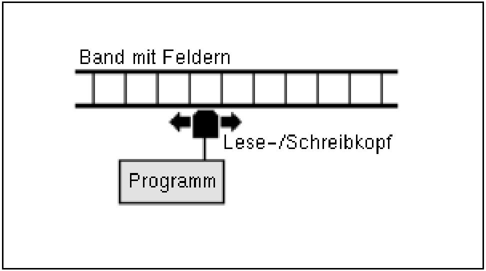
Bei Automaten und Akzeptoren waren die Funktionen nicht partiell.

Was bewirken partielle Zustandsübergangsfunktionen?



Turingmaschine: Verständnis

Woher kennen wir ähnliche Funktionen wie f und g?
Von Automaten und Akzeptoren.
Wo war dort der Unterschied?
Bei Automaten und Akzeptoren waren die Funktionen nicht partiell.
Was bewirken partielle Zustandsübergangsfunktionen?
Die partiellen Funktionen bewirken, dass der Automat zu manchen *Konfigurationen* stehen bleibt.



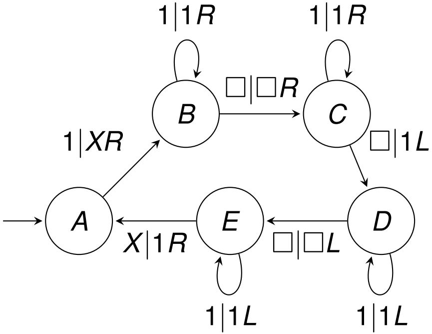
Konfiguration einer Turingmaschine

Definition: Konfiguration

Eine Turingmaschine befindet sich zu jedem Zeitpunkt in einem "Gesamtzustand", den wir eine Konfiguration nennen. Sie ist beschrieben durch

- den aktuellen Zustand $z \in Z$ der Steuereinheit
- die aktuelle Beschriftung $b \in X^*$ des gesamten Bandes
- die aktuelle Position $p \in \mathbb{Z}$ des Kopfes

Turingmaschine: Beispiel



	A	B	C	D	E
□		C,□,R	D,1,L	E,□,L	
1	B,X,R	B,1,R	C,1,R	D,1,L	E,1,L
X					A,1,R

Turingmaschine: Beispiel

Was macht die Turingmaschine?

Was passiert mit dem Wort . . . □11□ . . . , das auf dem Band steht?

		A ↓						
1.	□	1	1	□	□	□	□	□

Turingmaschine: Beispiel

Was macht die Turingmaschine?

Was passiert mit dem Wort . . . □11□ . . . , das auf dem Band steht?

		A						
1.		↓						
	□	1	1	□	□	□	□	□
			B					
2.			↓					
	□	X	1	□	□	□	□	□

Turingmaschine: Beispiel

Was macht die Turingmaschine?

Was passiert mit dem Wort . . . □11□ . . . , das auf dem Band steht?

		A						
1.		↓						
	□	1	1	□	□	□	□	□
			B					
2.			↓					
	□	X	1	□	□	□	□	□
				B				
3.				↓				
	□	X	1	□	□	□	□	□

Turingmaschine: Beispiel

Was macht die Turingmaschine?

Was passiert mit dem Wort . . . □11□ . . . , das auf dem Band steht?

		A						
1.		↓						
	□	1	1	□	□	□	□	□
			B					
2.			↓					
	□	X	1	□	□	□	□	□
				B				
3.				↓				
	□	X	1	□	□	□	□	□
					C			
4.					↓			
	□	X	1	□	□	□	□	□

				D				
5.				$\downarrow$				
	$\square$	X	1	$\square$	1	$\square$	$\square$	$\square$

				D				
5.				↓				
	□	X	1	□	1	□	□	□
			E					
6.			↓					
	□	X	1	□	1	□	□	□

				D				
5.				↓				
	□	X	1	□	1	□	□	□
			E					
6.			↓					
	□	X	1	□	1	□	□	□
		E						
7.		↓						
	□	X	1	□	1	□	□	□

				D				
5.				↓				
	□	X	1	□	1	□	□	□
			E					
6.			↓					
	□	X	1	□	1	□	□	□
		E						
7.		↓						
	□	X	1	□	1	□	□	□
			A					
8.			↓					
	□	1	1	□	1	□	□	□

				D				
5.				↓				
	□	X	1	□	1	□	□	□
			E					
6.			↓					
	□	X	1	□	1	□	□	□
		E						
7.		↓						
	□	X	1	□	1	□	□	□
			A					
8.			↓					
	□	1	1	□	1	□	□	□
				B				
9.				↓				
	□	1	X	□	1	□	□	□

5. D ↓

□ X 1 □ 1 □ □ □

6. E ↓

□ X 1 □ 1 □ □ □

7. E ↓

□ X 1 □ 1 □ □ □

8. A ↓

□ 1 1 □ 1 □ □ □

9. B ↓

□ 1 X □ 1 □ □ □

10. C ↓

□ 1 X □ 1 □ □ □

						C		
11.						$\downarrow$		
	$\square$	1	X	$\square$	1	$\square$	$\square$	$\square$

					C			
11.					↓			
	□	1	X	□	1	□	□	□
				D				
12.				↓				
	□	1	X	□	1	1	□	□

11.						C ↓		
	□	1	X	□	1	□	□	□
12.					D ↓			
	□	1	X	□	1	1	□	□
13.				D ↓				
	□	1	X	□	1	1	□	□

11.						C ↓		
	□	1	X	□	1	□	□	□
12.					D ↓			
	□	1	X	□	1	1	□	□
13.				D ↓				
	□	1	X	□	1	1	□	□
14.			E ↓					
	□	1	X	□	1	1	□	□

						C		
11.						↓		
	□	1	X	□	1	□	□	□
					D			
12.					↓			
	□	1	X	□	1	1	□	□
				D				
13.				↓				
	□	1	X	□	1	1	□	□
			E					
14.			↓					
	□	1	X	□	1	1	□	□
				A				
15.				↓				
	□	1	1	□	1	1	□	□

Turingmaschine: Beispiele

Nicht jede Turingmaschine kommt wie die vorherige zum Halten. Es gibt auch unendliche Berechnungen (wie in Java).

Turingmaschine als Akzeptor

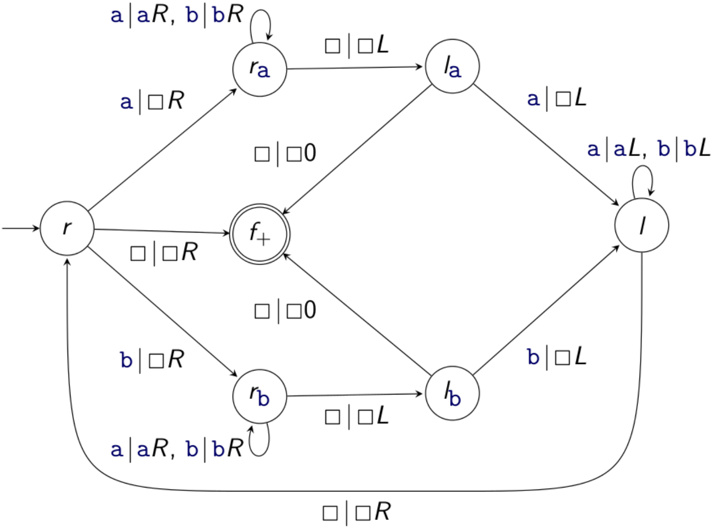
Ist eine Turingmaschine ein Akzeptor, so ist ein Eingabewort akzeptiert, wenn der Endzustand ein akzeptierter Zustand ist.

Definition: Eigenschaften von Sprachen

Eine Sprache L ist

- eine aufzählbare Sprache, wenn es eine Turingmaschine gibt, die L akzeptiert oder
- eine entscheidbare Sprache, wenn es eine Turingmaschine gibt, die L akzeptiert und immer hält.

Turingmaschine als Akzeptor: Beispiel



Klausuraufgabe

Wintersemester 11/12

Konstruieren Sie eine Turingmaschine, die für die Eingabe w die Funktion $f : \{a, b\}^* \rightarrow \mathbb{G}_3, f(w) = N_b(w) mod\ 3$ berechnet und das Ergebnis (nur von Blanksymbolen umgeben) an beliebiger Stelle auf das Band schreibt.

Die Laufzeit der Turingmaschine soll durch $\mathcal{O}(n)$ beschränkt sein und die Turingmaschine soll höchstens 6 Zustände enthalten. Es ist möglich mit weniger Zuständen auszukommen.

Überblick

Alan Turing

- Verantwortlich für viele wichtige Entwicklungen in der theoretischen Informatik
- Mitentwickler der *Turing-Bombe*, die im zweiten Weltkrieg zur Entschlüsselung der Enigma half
- nebenbei auch guter Marathonläufer (nahm an Olympiavorwettkämpfen teil)
- wurde wegen Homosexualität 1952 einer psychiatrischen Zwangsbehandlung unterzogen
- musste dabei weibliche Hormone nehmen
- Depression führten zu Selbstmord

Überblick

Hinweis

Hinweis

Die Komplexität untersuchen wir nur an Turingmaschinenen, die für jede Eingaben zum Halten kommen.

Komplexität

Definition: Zeitkomplexität

Die Zeitkomplexität $Time(n)$ einer Turingmaschine ist die maximale Anzahl an Schritten, die eine Turingmaschine bei Eingabe eines Wortes der Länge n benötigen kann.
(Worst-Case)

Definition: Platzkomplexität

Die Platzkomplexität $Space(n)$ einer Turingmaschine ist die maximale Anzahl an Feldern, die eine Turingmaschine bei Eingabe eines Wortes der Länge n benötigen kann.
(Worst-Case)
Benötigt wird ein Feld, sobald es vom Schreibkopf besucht wird.

Beispiel: Zeitkomplexität der Palindromturingmaschine

1. n Schritte vom ersten zum letzten Symbol
2. n Schritte wieder zurück zum ersten Symbol
3. Gleiche Prozedur mit "innerem Wort": $1 + \mathit{Time}\,(n-2)$-Schritte

Insgesamt:

$$T(n) \leq n + n + 1 + T(n-2)$$
$$T(n) - T(n-2) \leq 2n + 1 \in \mathcal{O}(n) \Rightarrow T(n) \in \mathcal{O}\left(n^2\right)$$

Beispiel: Platzkomplexität der Palindromturingmaschine

1. $n + 1$ Felder vom ersten zum letzten Symbol
2. 0 Felder auf dem Weg zurück zum ersten Symbol
3. Gleiche Prozedur mit “innerem Wort”: 0 Felder

Insgesamt:

$$n + 1$$

Komplexitätsklassen

Definition der Komplexitätsklassen

- **P** ist die Menge aller Entscheidungsprobleme, die von Turingmaschinen entschieden werden können, deren Zeitkomplexität polynomiell ist.
- **PSPACE** ist die Menge aller Entscheidungsprobleme, die von Turingmaschinen entschieden werden können, deren Raumkomplexität polynomiell ist.

Daraus kann man leicht folgern:

$$P \subset PSPACE$$

Denn in t Schritten sind maximal $t + 1$ Felder erreichbar.

Klausuraufgabe

Sommersemester 2010

Gegeben sei folgende Turingmaschine T:

- Anfangszustand ist z_0
- Die Arbeitsweis ist wie folgt festgelegt:

	z_0	z_1	z_2	z_3	z_4
$\square$		$z_4,\square,L$			
a	z_0,a,R	z_2,b,L	z_0,a,R	z_4,b,R	
b	z_1,a,R	z_1,b,R	z_2,b,L	z_3,b,L	

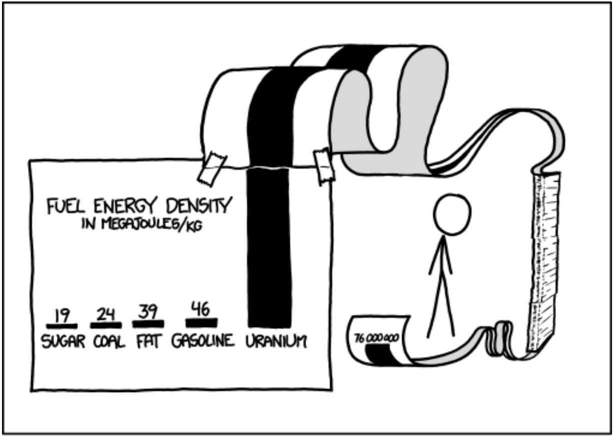
1. Zeichnen Sie T
2. berechnen Sie folgende Eingaben: aab, aba, baa

Überblick

Fragen

- Fragen zum Stoff?
- Fragen zum nächsten Übungsblatt?
- Generelle Fragen?
- Feedback?

EOF



SCIENCE TIP: LOG SCALES ARE FOR QUITTERS WHO CAN'T FIND ENOUGH PAPER TO MAKE THEIR POINT *PROPERLY*.

source : http : //imgs.xkcd.com/comics/log_scale.png