

# Theoretische Grundlagen der Informatik

## Übung am 18.11.2010

INSTITUT FÜR THEORETISCHE INFORMATIK



- 3. Übungsblatt wird im Laufe des Tages auf die Homepage gestellt

# Aufgabe 1

Wir betrachten den endlichen Automaten über dem Alphabet  $\Sigma = \{0, 1\}$  mit den Zuständen  $\{A, B, C, D, E, F, G\}$ . Startzustand ist  $A$ , Endzustände sind  $D$  und  $E$ . Die Übergangsfunktion ist gegeben durch folgende Tabelle:

$\delta$	A	B	C	D	E	F	G
0	C	F	A	C	F	F	F
1	B	E	D	B	E	E	D

- Zeichnen Sie den Automaten.
- Entfernen Sie daraus alle nicht erreichbaren Zustände
- Konstruieren Sie daraus den zugehörigen Minimalautomaten

## Aufgabe 1 a)

Zeichnen Sie den Automaten.

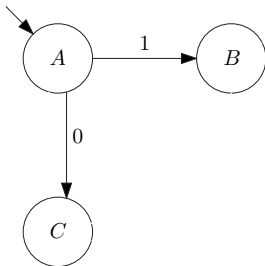
$\delta$	A	B	C	D	E	F	G
0	C	F	A	C	F	F	F
1	B	E	D	B	E	E	D



## Aufgabe 1 a)

Zeichnen Sie den Automaten.

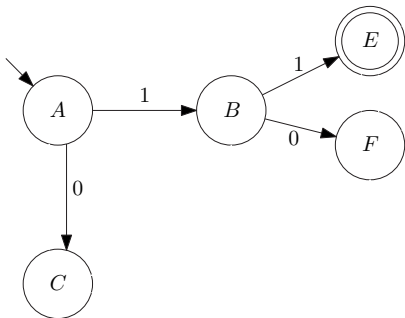
$\delta$	A	B	C	D	E	F	G
0	C	F	A	C	F	F	F
1	B	E	D	B	E	E	D



## Aufgabe 1 a)

Zeichnen Sie den Automaten.

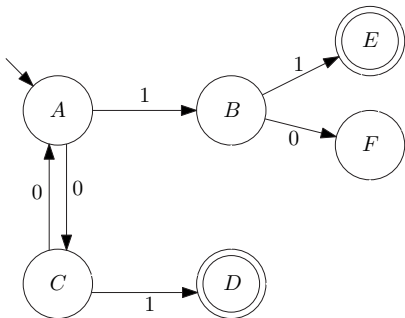
$\delta$	A	B	C	D	E	F	G
0	C	F	A	C	F	F	F
1	B	E	D	B	E	E	D



## Aufgabe 1 a)

Zeichnen Sie den Automaten.

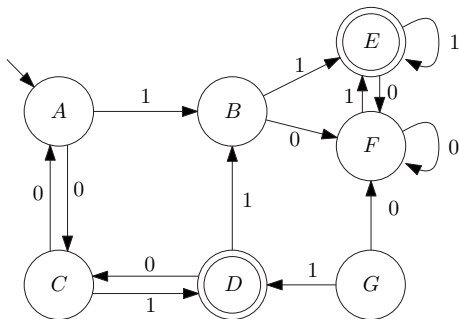
$\delta$	A	B	C	D	E	F	G
0	C	F	A	C	F	F	F
1	B	E	D	B	E	E	D



# Aufgabe 1 a)

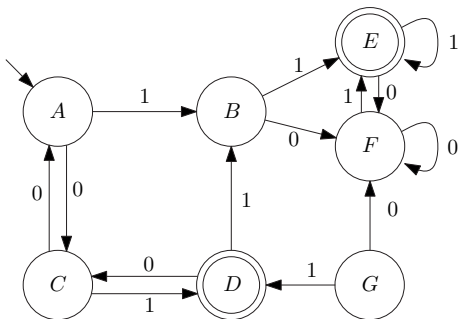
Zeichnen Sie den Automaten.

$\delta$	A	B	C	D	E	F	G
0	C	F	A	C	F	F	F
1	B	E	D	B	E	E	D



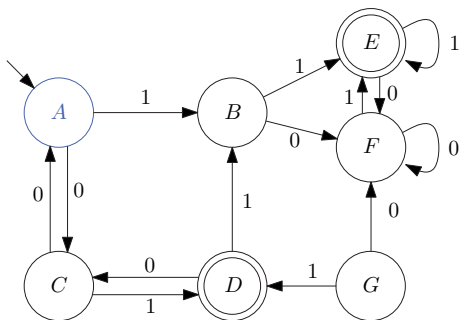
## Aufgabe 1 b)

Entfernen Sie daraus alle nicht erreichbaren Zustände.



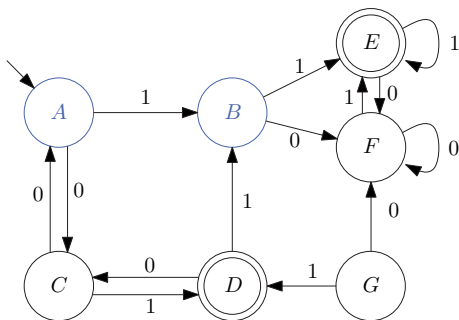
## Aufgabe 1 b)

Entfernen Sie daraus alle nicht erreichbaren Zustände.



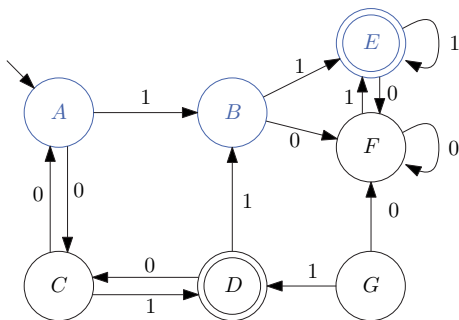
## Aufgabe 1 b)

Entfernen Sie daraus alle nicht erreichbaren Zustände.



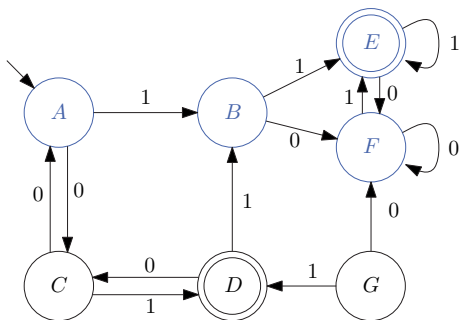
## Aufgabe 1 b)

Entfernen Sie daraus alle nicht erreichbaren Zustände.



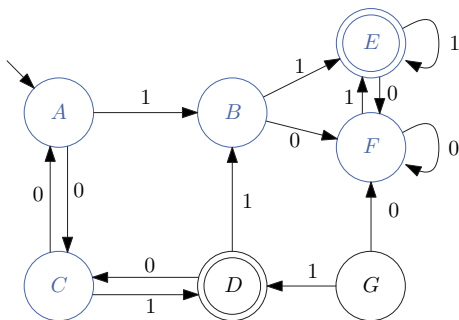
## Aufgabe 1 b)

Entfernen Sie daraus alle nicht erreichbaren Zustände.



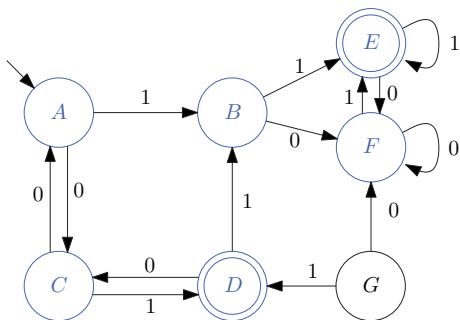
## Aufgabe 1 b)

Entfernen Sie daraus alle nicht erreichbaren Zustände.



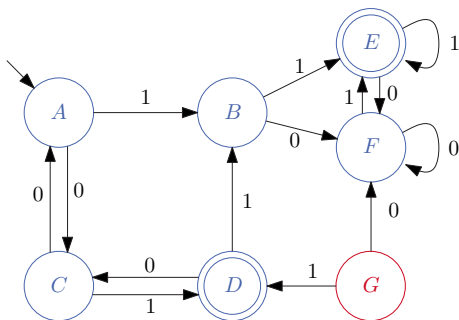
## Aufgabe 1 b)

Entfernen Sie daraus alle nicht erreichbaren Zustände.



## Aufgabe 1 b)

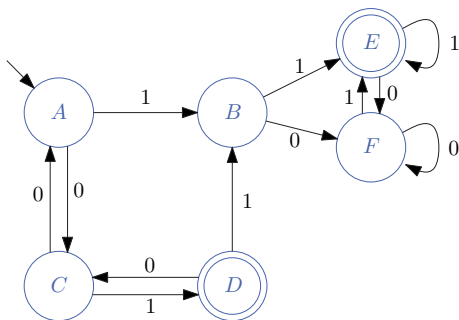
Entfernen Sie daraus alle nicht erreichbaren Zustände.



Einzigster nicht erreichbarer Zustand ist G, dieser kann gelöscht werden.

## Aufgabe 1 b)

Entfernen Sie daraus alle nicht erreichbaren Zustände.



Einziger nicht erreichbarer Zustand ist G, dieser kann gelöscht werden.

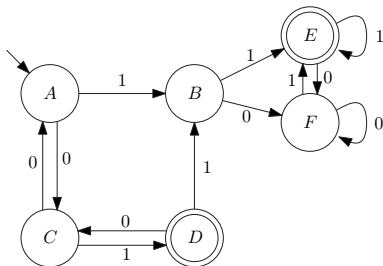
# Vorgehen zur Konstruktion des Minimalautomaten

Ein **Zeuge** für die Nichtäquivalenz von zwei Zuständen  $p$  und  $q$  ist ein Wort  $w \in \Sigma^*$  mit  $\delta(p, w) \in F$  und  $\delta(q, w) \notin F$  bzw.  $\delta(p, w) \notin F$  und  $\delta(q, w) \in F$

**Vorgehensweise (Kurzfassung):** Betrachte zunächst alle Zustände als eine Klasse. Beginnend mit  $l = 0$  teste alle Wörter der Länge  $l$ , ob sie Zeuge der Nichtäquivalenz zweier Zustände sind, die sich momentan in einer Klasse befinden. Das Verfahren bricht ab, wenn für eine bestimmte Länge keine Zustände mehr getrennt werden.

## Aufgabe 1 c)

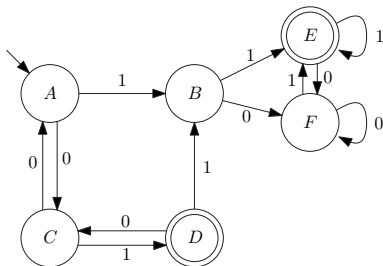
Konstruieren Sie daraus den zugehörigen Minimalautomaten.



- $\varepsilon$  trennt  $\{A, B, C, F\}$  von  $\{D, E\}$
- 0 trennt nichts
- 1 trennt  $\{A\}$  von  $\{B, C, F\}$  und  $\{D\}$  von  $\{E\}$
- 00 trennt nichts
- 01 trennt  $\{B, F\}$  von  $\{C\}$
- 10 trennt nichts
- 11 trennt nichts
- 000, 001, 010 011, 100, 101, 110 und 111 trennen nichts

## Aufgabe 1 c)

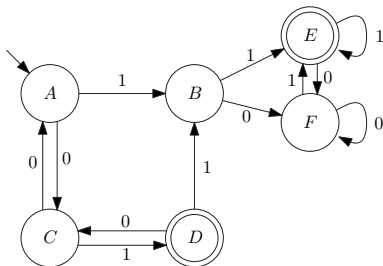
Konstruieren Sie daraus den zugehörigen Minimalautomaten.



- $\varepsilon$  trennt  $\{A, B, C, F\}$  von  $\{D, E\}$
- 0 trennt nichts
- 1 trennt  $\{A\}$  von  $\{B, C, F\}$  und  $\{D\}$  von  $\{E\}$
- 00 trennt nichts
- 01 trennt  $\{B, F\}$  von  $\{C\}$
- 10 trennt nichts
- 11 trennt nichts
- 000, 001, 010 011, 100, 101, 110 und 111 trennen nichts

## Aufgabe 1 c)

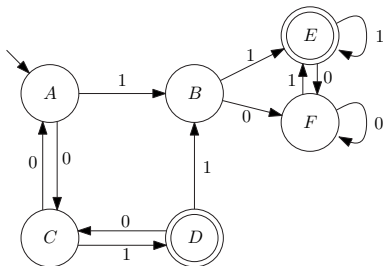
Konstruieren Sie daraus den zugehörigen Minimalautomaten.



- $\varepsilon$  trennt  $\{A, B, C, F\}$  von  $\{D, E\}$
- 0 trennt nichts
- 1 trennt  $\{A\}$  von  $\{B, C, F\}$  und  $\{D\}$  von  $\{E\}$
- 00 trennt nichts
- 01 trennt  $\{B, F\}$  von  $\{C\}$
- 10 trennt nichts
- 11 trennt nichts
- 000, 001, 010 011, 100, 101, 110 und 111 trennen nichts

## Aufgabe 1 c)

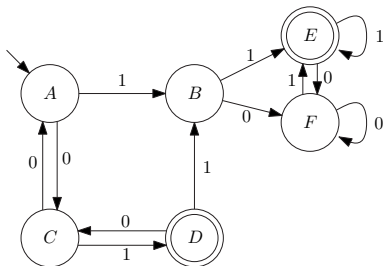
Konstruieren Sie daraus den zugehörigen Minimalautomaten.



- $\varepsilon$  trennt  $\{A, B, C, F\}$  von  $\{D, E\}$
- 0 trennt nichts
- 1 trennt  $\{A\}$  von  $\{B, C, F\}$  und  $\{D\}$  von  $\{E\}$
- 00 trennt nichts
- 01 trennt  $\{B, F\}$  von  $\{C\}$
- 10 trennt nichts
- 11 trennt nichts
- 000, 001, 010 011, 100, 101, 110 und 111 trennen nichts

## Aufgabe 1 c)

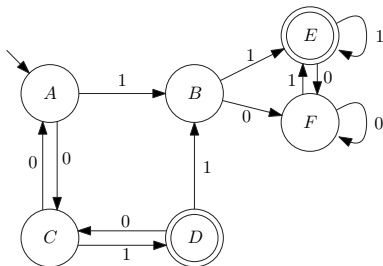
Konstruieren Sie daraus den zugehörigen Minimalautomaten.



- $\varepsilon$  trennt  $\{A, B, C, F\}$  von  $\{D, E\}$
- 0 trennt nichts
- 1 trennt  $\{A\}$  von  $\{B, C, F\}$  und  $\{D\}$  von  $\{E\}$
- 00 trennt nichts
- 01 trennt  $\{B, F\}$  von  $\{C\}$
- 10 trennt nichts
- 11 trennt nichts
- 000, 001, 010 011, 100, 101, 110 und 111 trennen nichts

## Aufgabe 1 c)

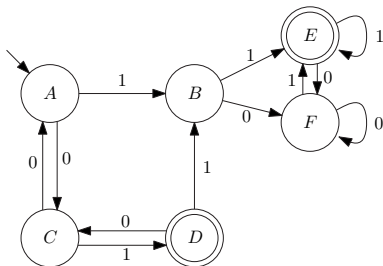
Konstruieren Sie daraus den zugehörigen Minimalautomaten.



- $\varepsilon$  trennt  $\{A, B, C, F\}$  von  $\{D, E\}$
- 0 trennt nichts
- 1 trennt  $\{A\}$  von  $\{B, C, F\}$  und  $\{D\}$  von  $\{E\}$
- 00 trennt nichts
- 01 trennt  $\{B, F\}$  von  $\{C\}$
- 10 trennt nichts
- 11 trennt nichts
- 000, 001, 010 011, 100, 101, 110 und 111 trennen nichts

## Aufgabe 1 c)

Konstruieren Sie daraus den zugehörigen Minimalautomaten.

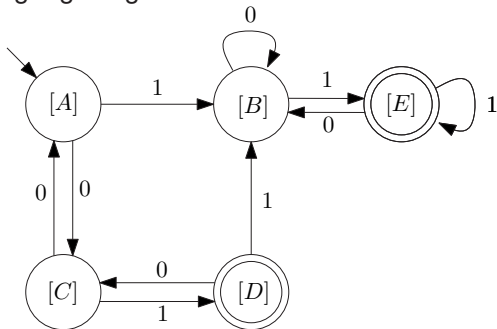


- $\varepsilon$  trennt  $\{A, B, C, F\}$  von  $\{D, E\}$
- 0 trennt nichts
- 1 trennt  $\{A\}$  von  $\{B, C, F\}$  und  $\{D\}$  von  $\{E\}$
- 00 trennt nichts
- 01 trennt  $\{B, F\}$  von  $\{C\}$
- 10 trennt nichts
- 11 trennt nichts
- 000, 001, 010 011, 100, 101, 110 und 111 trennen nichts

## Aufgabe 1 c)

Konstruieren Sie daraus den zugehörigen Minimalautomaten.

- Äquivalenzklassen:  $[A] = \{A\}$ ,  $[B] = \{B, F\}$ ,  $[C] = \{C\}$ ,  $[D] = \{D\}$  und  $[E] = \{E\}$
- Startzustand:  $[A]$
- Endzustände:  $[D]$  und  $[E]$
- Zustandsübergangsdiagramm:



## Aufgabe 2

Zeigen Sie mit Hilfe des Pumping-Lemmas die Nicht-Regularität folgender Sprachen:

- $L_1 = \{ww^R \mid w \in \{a, b\}^*\}$ , wobei  $w^R$  das ‚Spiegelwort‘ zu  $w$  ist (Sprache der Palindrome gerader Länge).
- Sei  $\Sigma = \{0, 1\}$  und  $L_2$  die Sprache aller Wörter aus  $\Sigma^*$ , die mehr Nullen als Einsen enthalten.
- $L_3 = \{a^i b^j c^k \mid i < j < k\}$ .

### **Pumping Lemma für reguläre Sprachen:**

Sei  $L$  eine reguläre Sprache. Dann existiert eine Zahl  $n \in \mathbb{N}$ , so dass für jedes Wort  $w \in L$  mit  $|w| > n$  eine Darstellung  $w = uvx$  mit  $|uv| \leq n$ ,  $v \neq \varepsilon$ , existiert, bei der auch  $uv^i x \in L$  ist für alle  $i \in \mathbb{N}_0$ .

Struktur des Beweises für alle Teilaufgaben gleich:

- Beweis durch Widerspruch
- Annahme:  $L_{\{1,2,3\}}$  ist regulär
- Sei  $n$  wie im Pumping Lemma, d.h., so dass für jedes Wort  $w \in L_{\{1,2,3\}}$  mit  $|w| > n$  eine Darstellung  $w = uvx$  mit  $|uv| \leq n$ ,  $v \neq \varepsilon$  existiert, bei der auch  $uv^i x \in L_{\{1,2,3\}}$  ist für alle  $i \in \mathbb{N}_0$
- Wähle dann  $w \in L_{\{1,2,3\}}$  abhängig von  $n$  geschickt, so dass es keine Unterteilung gemäß des Pumping-Lemmas gibt

### Pumping Lemma für reguläre Sprachen:

Sei  $L$  eine reguläre Sprache. Dann existiert eine Zahl  $n \in \mathbb{N}$ , so dass für jedes Wort  $w \in L$  mit  $|w| > n$  eine Darstellung  $w = uvx$  mit  $|uv| \leq n$ ,  $v \neq \varepsilon$ , existiert, bei der auch  $uv^i x \in L$  ist für alle  $i \in \mathbb{N}_0$ .

Teilaufgabe a):  $L_1 = \{ww^R \mid w \in \{a, b\}^*\}$ , wobei  $w^R$  das ‚Spiegelwort‘ zu  $w$  ist (Sprache der Palindrome gerader Länge).

- Wähle  $w = a^n b^{2n} a^n \in L_1$
- Dann ist  $|w| > n$  und für jede Unterteilung  $w = uvx$  mit  $|uv| \leq n$ ,  $v \neq \varepsilon$  ist  $uv^i x = a^{n+m} b^{2n} a^n$  mit  $m > 0$ , d.h.  $uv^i x \notin L_1$
- Das ist ein Widerspruch zum Pumping-Lemma

### **Pumping Lemma für reguläre Sprachen:**

Sei  $L$  eine reguläre Sprache. Dann existiert eine Zahl  $n \in \mathbb{N}$ , so dass für jedes Wort  $w \in L$  mit  $|w| > n$  eine Darstellung  $w = uvx$  mit  $|uv| \leq n$ ,  $v \neq \varepsilon$ , existiert, bei der auch  $uv^i x \in L$  ist für alle  $i \in \mathbb{N}_0$ .

Teilaufgabe b):  $L_2$  sei die Sprache aller Wörter aus  $\{0, 1\}^*$ , die mehr Nullen als Einsen enthalten.

- Wähle  $w = 1^n 0^{n+1} \in L_2$
- Dann ist  $|w| > n$  und für jede Unterteilung  $w = uvx$  mit  $|uv| \leq n$ ,  $v \neq \varepsilon$  ist  $uv^i x = 1^{n+m} 0^{n+1}$  mit  $m > 0$ , d.h.  $uv^i x \notin L_2$
- Das ist ein Widerspruch zum Pumping-Lemma

### Pumping Lemma für reguläre Sprachen:

Sei  $L$  eine reguläre Sprache. Dann existiert eine Zahl  $n \in \mathbb{N}$ , so dass für jedes Wort  $w \in L$  mit  $|w| > n$  eine Darstellung  $w = uvx$  mit  $|uv| \leq n$ ,  $v \neq \varepsilon$ , existiert, bei der auch  $uv^i x \in L$  ist für alle  $i \in \mathbb{N}_0$ .

Teilaufgabe c):  $L_3 = \{a^i b^j c^k \mid i < j < k\}$

- Wähle  $w = a^n b^{n+1} c^{n+2} \in L_3$
- Dann ist  $|w| > n$  und für jede Unterteilung  $w = uvx$  mit  $|uv| \leq n$ ,  $v \neq \varepsilon$  ist  $uv^i x = a^{n+m} b^{n+1} c^{n+2}$  mit  $m > 0$ , d.h.  $uv^i x \notin L_3$
- Das ist ein Widerspruch zum Pumping-Lemma

## Aufgabe 3

Seien  $L$ ,  $L_1$  und  $L_2$  reguläre Sprachen. Beweisen Sie:

- $L^c := \Sigma^* \setminus L$  ist regulär
- $L_3 := L_1 \cap L_2$  ist regulär
- $L_4 := \{w \in \Sigma^* \mid w \text{ ist in } L_1, \text{ aber nicht in } L_2\}$  ist regulär

## Aufgabe 3 a)

Sei  $L$  eine reguläre Sprache. Beweisen Sie:

- $L^c := \Sigma^* \setminus L$  ist regulär

## Aufgabe 3 a)

Sei  $L$  eine reguläre Sprache. Beweisen Sie:

- $L^c := \Sigma^* \setminus L$  ist regulär

### Beweis:

- Sei  $A = (Q, \Sigma, \delta, s, F)$  DEA, der  $L$  akzeptiert
- Konstruiere daraus DEA  $\bar{A} = (Q, \Sigma, \delta, s, Q \setminus F)$ , d.h. bei  $A$  werden einfach Endzustände und Nicht-Endzustände vertauscht
- Dann akzeptiert  $\bar{A}$  genau die Worte über  $\Sigma$ , die nicht in  $L$  liegen, also  $L^c$
- Also ist  $L^c$  regulär

## Aufgabe 3 b)

Seien  $L_1$  und  $L_2$  reguläre Sprachen. Beweisen Sie:

- $L_3 := L_1 \cap L_2$  ist regulär

## Aufgabe 3 b)

Seien  $L_1$  und  $L_2$  reguläre Sprachen. Beweisen Sie:

- $L_3 := L_1 \cap L_2$  ist regulär

### Beweis:

- Eine Möglichkeit: Nimm Automaten  $A_1$  und  $A_2$ , die  $L_1$  und  $L_2$  akzeptieren und baue daraus Automaten für  $L_1 \cap L_2$
- Aber: Es geht viel einfacher, denn
- $L_1 \cap L_2 = (L_1^c \cup L_2^c)^c$
- Mit Teilaufgabe a) sind  $L_1^c$  und  $L_2^c$  regulär und nach Definition regulärer Sprachen ist damit auch  $L_1^c \cup L_2^c$  regulär
- Damit ist auch  $L_3$  regulär

## Aufgabe 3 c)

Seien  $L_1$  und  $L_2$  reguläre Sprachen. Beweisen Sie:

- $L_4 := \{w \in \Sigma^* \mid w \text{ ist in } L_1, \text{ aber nicht in } L_2\}$  ist regulär

## Aufgabe 3 c)

Seien  $L_1$  und  $L_2$  reguläre Sprachen. Beweisen Sie:

- $L_4 := \{w \in \Sigma^* \mid w \text{ ist in } L_1, \text{ aber nicht in } L_2\}$  ist regulär

### Beweis:

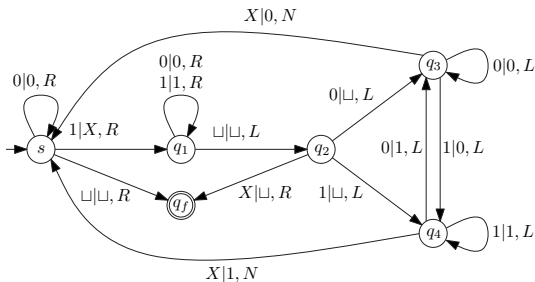
- Lösung ähnlich zu Teilaufgabe b) möglich, denn:
- $L_4 = L_1 \cap L_2^c$
- Mit Teilaufgaben a) und b) folgt damit direkt, dass auch  $L_4$  regulär ist

## Aufgabe 4

Oft wird die Situation, in der sich eine Turingmaschine  $\mathcal{M} = (Q, \Sigma, \Gamma, \delta, s, F)$  befindet, durch die Angabe der *Konfiguration* in der Form  $w(q)av$  codiert, wobei  $w, v \in \Gamma^*$ ,  $a \in \Gamma$  und  $q \in Q$  sind. Dies bedeutet, dass sich  $\mathcal{M}$  gerade im Zustand  $q$  befindet; der Lesekopf steht auf dem Zeichen  $a$ ; links davon steht das Wort  $w$  und rechts davon das Wort  $v$  auf dem Rechenband.

## Aufgabe 4

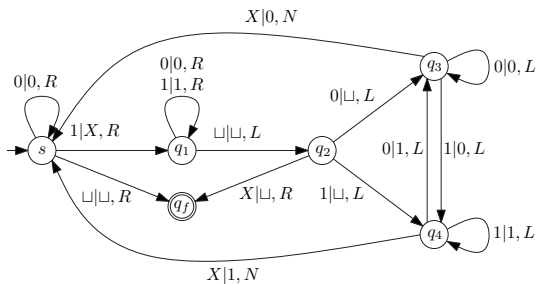
Gegeben sei nun die Turingmaschine  $\mathcal{M}$  durch das folgende Zustandsdiagramm:



- Streng genommen hätte man noch  $\Sigma = \{0, 1\}$  und  $\Gamma = \{0, 1, \sqcup, X\}$  angeben müssen
- Falls jemand bei Aufgabe b) und c) angenommen hat, dass sich auch andere Zeichen in der Eingabe befinden dürfen, bekommt er auch die Punkte

## Aufgabe 4 a)

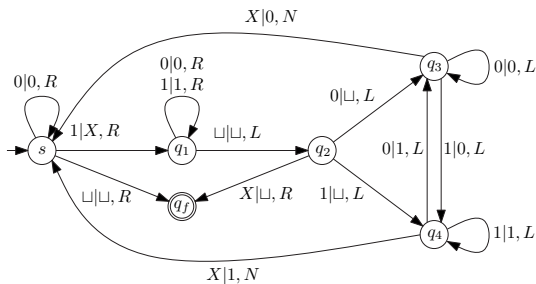
Welche Konfigurationen durchläuft die Maschine bei Eingabe des Wortes 01010?



- |                  |                  |                 |                          |
|------------------|------------------|-----------------|--------------------------|
| ■ (s)01010       | ■ 0X010( $q_1$ ) | ■ 0(s)010       | ■ 00X( $q_2$ )0          |
| ■ 0(s)1010       | ■ 0X01( $q_2$ )0 | ■ 00(s)10       | ■ 00( $q_3$ )X           |
| ■ 0X( $q_1$ )010 | ■ 0X0( $q_3$ )1  | ■ 00X( $q_1$ )0 | ■ 00(s)0                 |
| ■ 0X0( $q_1$ )10 | ■ 0X( $q_4$ )00  | ■ 00X0( $q_1$ ) | ■ 000(s)                 |
| ■ 0X01( $q_1$ )0 | ■ 0( $q_3$ )X10  | ■ 00X0( $q_1$ ) | ■ 000 $\sqcup$ ( $q_f$ ) |

## Aufgabe 4 a)

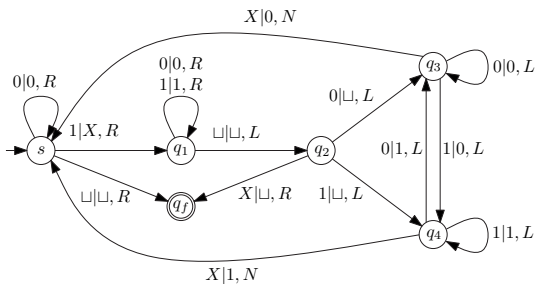
Welche Konfigurationen durchläuft die Maschine bei Eingabe des Wortes 01010?



- |                  |                  |                 |                          |
|------------------|------------------|-----------------|--------------------------|
| ■ (s)01010       | ■ 0X010( $q_1$ ) | ■ 0(s)010       | ■ 00X( $q_2$ )0          |
| ■ 0(s)1010       | ■ 0X01( $q_2$ )0 | ■ 00(s)10       | ■ 00( $q_3$ )X           |
| ■ 0X( $q_1$ )010 | ■ 0X0( $q_3$ )1  | ■ 00X( $q_1$ )0 | ■ 00(s)0                 |
| ■ 0X0( $q_1$ )10 | ■ 0X( $q_4$ )00  | ■ 00X0( $q_1$ ) | ■ 000(s)                 |
| ■ 0X01( $q_1$ )0 | ■ 0( $q_3$ )X10  | ■ 00X0( $q_1$ ) | ■ 000 $\sqcup$ ( $q_f$ ) |

## Aufgabe 4 a)

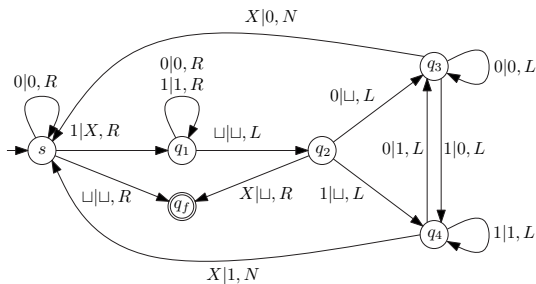
Welche Konfigurationen durchläuft die Maschine bei Eingabe des Wortes 01010?



- |                |                |               |                    |
|----------------|----------------|---------------|--------------------|
| ■ $(s)01010$   | ■ $0X010(q_1)$ | ■ $0(s)010$   | ■ $00X(q_2)0$      |
| ■ $0(s)1010$   | ■ $0X01(q_2)0$ | ■ $00(s)10$   | ■ $00(q_3)X$       |
| ■ $0X(q_1)010$ | ■ $0X0(q_3)1$  | ■ $00X(q_1)0$ | ■ $00(s)0$         |
| ■ $0X0(q_1)10$ | ■ $0X(q_4)00$  | ■ $00X0(q_1)$ | ■ $000(s)$         |
| ■ $0X01(q_1)0$ | ■ $0(q_3)X10$  | ■ $00X0(q_1)$ | ■ $000\sqcup(q_f)$ |

## Aufgabe 4 a)

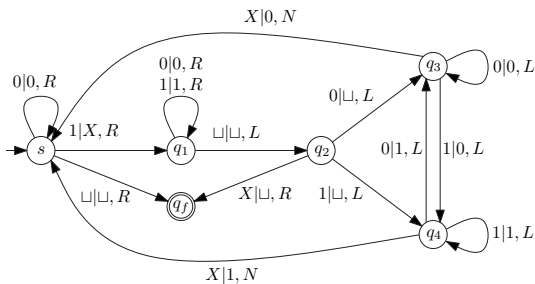
Welche Konfigurationen durchläuft die Maschine bei Eingabe des Wortes 01010?



- |                |                |               |                    |
|----------------|----------------|---------------|--------------------|
| ■ $(s)01010$   | ■ $0X010(q_1)$ | ■ $0(s)010$   | ■ $00X(q_2)0$      |
| ■ $0(s)1010$   | ■ $0X01(q_2)0$ | ■ $00(s)10$   | ■ $00(q_3)X$       |
| ■ $0X(q_1)010$ | ■ $0X0(q_3)1$  | ■ $00X(q_1)0$ | ■ $00(s)0$         |
| ■ $0X0(q_1)10$ | ■ $0X(q_4)00$  | ■ $00X0(q_1)$ | ■ $000(s)$         |
| ■ $0X01(q_1)0$ | ■ $0(q_3)X10$  | ■ $00X0(q_1)$ | ■ $000\sqcup(q_f)$ |

## Aufgabe 4 a)

Welche Konfigurationen durchläuft die Maschine bei Eingabe des Wortes 01010?

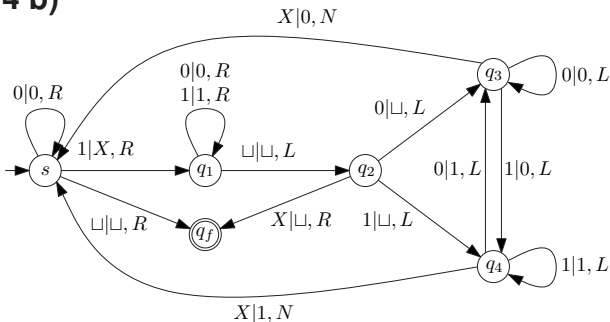


- |                |                |               |                    |
|----------------|----------------|---------------|--------------------|
| ■ $(s)01010$   | ■ $0X010(q_1)$ | ■ $0(s)010$   | ■ $00X(q_2)0$      |
| ■ $0(s)1010$   | ■ $0X01(q_2)0$ | ■ $00(s)10$   | ■ $00(q_3)X$       |
| ■ $0X(q_1)010$ | ■ $0X0(q_3)1$  | ■ $00X(q_1)0$ | ■ $00(s)0$         |
| ■ $0X0(q_1)10$ | ■ $0X(q_4)00$  | ■ $00X0(q_1)$ | ■ $000(s)$         |
| ■ $0X01(q_1)0$ | ■ $0(q_3)X10$  | ■ $00X0(q_1)$ | ■ $000\sqcup(q_f)$ |

## Aufgabe 4 b)

Welche Aufgabe haben die Zustände  $q_2$ ,  $q_3$  und  $q_4$ ?  
(Hinweis: Überlegen Sie dazu, was passiert, wenn  $\mathcal{M}$  sich im Zustand  $q_2$  befindet und der Lesekopf auf dem letzten Zeichen der Eingabe steht, bis zu dem Zeitpunkt, zu dem das erste Mal  $X$  gelesen wird.)

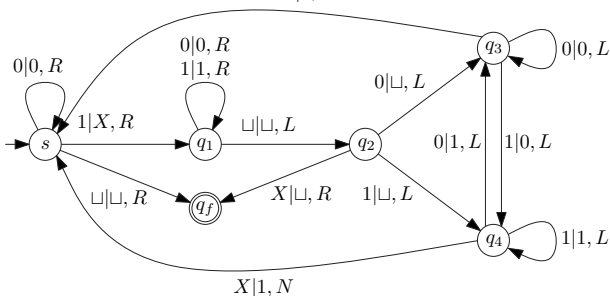
## Aufgabe 4 b)



- Der Schreibkopf bewegt sich immer nach links, bis  $X$  gelesen wird, dabei werden Zeichen überschrieben
- $q_2$  löscht das letzte Zeichen
- Wenn in Zustand  $q_3(q_4)$ , dann war letztes gelesenes Zeichen eine  $0(1)$ , nächstes Zeichen wird mit  $0(1)$  überschrieben
- Insgesamt: Das Suffix des Wortes, das mit  $X$  beginnt, wird um eine Stelle nach links geschoben, dabei wird das  $X$  überschrieben

## Aufgabe 4 c)

Wie verändert  $\mathcal{M}$  die Eingabe?  $X|0, N$



- In Zustand  $s$  läuft Kopf nach rechts, bis die erste 1 gefunden wird, diese wird mit  $X$  überschrieben
- Zustand  $q_1$  bewegt den Kopf zum letzten Zeichen des Wortes
- Zustände  $q_2, q_3$  und  $q_4$  schieben den hinteren Teil des Wortes eine Stelle nach links, dabei wird das  $X$  überschrieben
- Insgesamt: Die Turingmaschine löscht alle Einsen aus einem Wort in  $\{0, 1\}^*$

## Aufgabe 5

Entwerfen Sie eine deterministische Turing-Maschine  $\mathcal{M} := (Q, \Sigma, \Gamma, \delta, s, F)$ , die genau die Sprache  $L := \{ww^R \in \{a, b\}^*\}$ , wobei  $w^R$  das „Spiegelwort“ zu  $w$  ist, akzeptiert. Dies ist die Sprache der Palindrome gerade Länge, die Sie bereits aus Aufgabe 2 kennen.

## Aufgabe 5

Beschreiben Sie zunächst kurz in Worten, wie Ihre Turing-Maschine arbeitet.

## Aufgabe 5

Beschreiben Sie zunächst kurz in Worten, wie Ihre Turing-Maschine arbeitet.

### Lösung:

- Idee: Prüfe immer, ob erstes Zeichen und letztes Zeichen identisch und lösche diese dann. Falls dies mit leerem Band endet, akzeptiere
- Prüfe zunächst, ob Band leer ist, falls ja, akzeptiere. Wiederhole dann iterativ folgende Schritte:
  - Lösche erstes Zeichen
  - Bewege Lesekopf zum letzten Zeichen (falls Band leer, war die Anzahl der Zeichen nicht gerade  $\Rightarrow$  verwerfe)
  - Falls 1. Zeichen  $a(b)$  war und letztes Zeichen  $b(a)$  ist, ist Eingabe kein Palindrom  $\Rightarrow$  verwerfe
  - Lösche letztes Zeichen
  - Falls Band leer  $\Rightarrow$  akzeptiere
  - Bewege Kopf zum ersten Zeichen

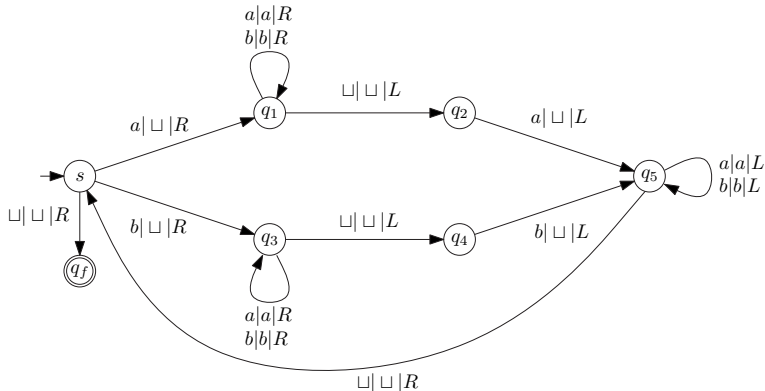
## Aufgabe 5

Spezifizieren Sie  $\mathcal{M}$  dann explizit durch Angabe eines Zustandsdiagramms oder einer formalen Spezifikation.

## Aufgabe 5

Spezifizieren Sie  $\mathcal{M}$  dann explizit durch Angabe eines Zustandsdiagramms oder einer formalen Spezifikation.

Eine Möglichkeit für eine Turingmaschine:



## Aufgabe 6

Zeige:

- Eine Sprache  $L$  ist genau dann entscheidbar, wenn  $L$  und  $L^c$  semientscheidbar sind.
- Die Menge der semi-entscheidbaren Sprachen ist unter Vereinigung und Schnitt abgeschlossen.
- Die Menge der semi-entscheidbaren Sprachen ist unter Komplementbildung nicht abgeschlossen.
- Ist die Menge der ungeraden Zahlen entscheidbar? Geben Sie eine kurze Begründung.  
(Hinweis: Dabei kann angenommen werden, dass natürliche Zahlen binär kodiert vorliegen.)

## Aufgabe 6 a)

Zeige: Eine Sprache  $L$  ist genau dann entscheidbar, wenn  $L$  und  $L^c$  semientscheidbar sind.

## Aufgabe 6 a)

Zeige: Eine Sprache  $L$  ist genau dann entscheidbar, wenn  $L$  und  $L^c$  semientscheidbar sind.

### Lösung:

- Falls  $L$  entscheidbar ist, dann gibt es eine TM  $M$ , die immer hält und genau die Worte aus  $L$  akzeptiert
- Damit ist  $L$  offensichtlich semi-entscheidbar und durch Vertauschen von Akzeptieren und Verwerfen bei  $M$  ist offensichtlich auch  $L^c$  semi-entscheidbar
- Noch zu zeigen: Falls  $L$  und  $L^c$  semientscheidbar sind, dann ist  $L$  entscheidbar
- Wir konstruieren stets haltende 2-Band-Turingmaschine  $M$ , die  $L$  akzeptiert
- Seien  $M_1$  und  $M_2$  Turingmaschinen, die für Eingaben aus  $L$  bzw.  $L^c$  in endlicher Zeit stoppen und nur diese Eingaben akzeptieren

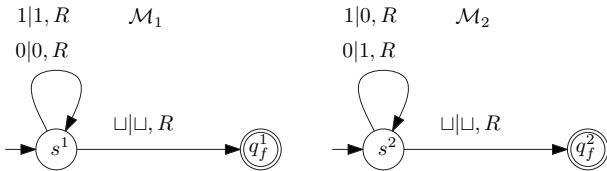
## Aufgabe 6 a)

Zeige: Eine Sprache  $L$  ist genau dann entscheidbar, wenn  $L$  und  $L^c$  semientscheidbar sind.

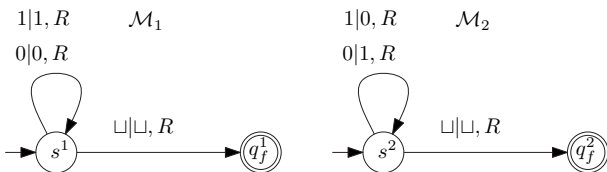
### Lösung:

- $M$  kopiert zunächst die Eingabe auf das zweite Band
- Auf Band 1 wird  $M_1$  simuliert, auf Band 2  $M_2$
- Zustandsmenge von  $M$  ist  $Q_1 \times Q_2$ , wobei  $Q_1$  Zustandsmenge von  $M_1$  und  $Q_2$  Zustandsmenge von  $M_2$  ist
- Simuliere immer abwechselnd einen Berechnungsschritt von  $M_1$  und  $M_2$  und wechsle dann das Band
- Es wird akzeptiert, wenn  $M_1$  (erstes Band) akzeptieren würde und verworfen, wenn  $M_2$  (zweites Band) akzeptieren würde
- Da die Eingabe entweder in  $L$  oder in  $L^c$  liegt, stoppt  $M$  immer und akzeptiert genau  $L$

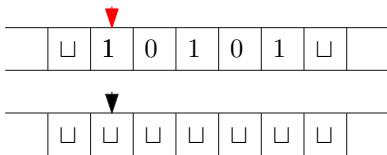
# Beispiel zur Konstruktion aus Teilaufgabe a)



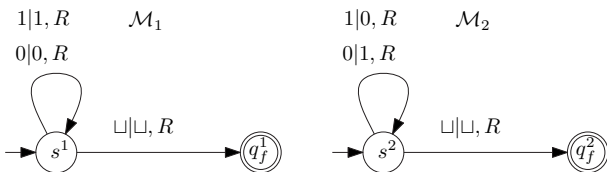
# Beispiel zur Konstruktion aus Teilaufgabe a)



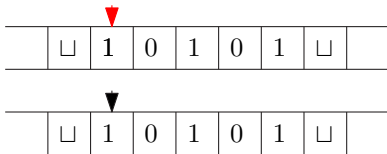
Anfangszustand:



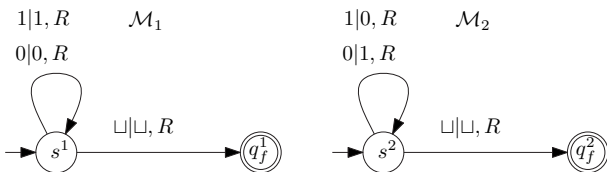
# Beispiel zur Konstruktion aus Teilaufgabe a)



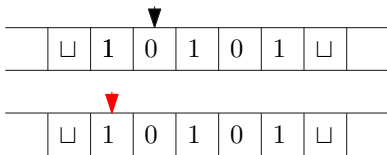
Kopiere Eingabe auf zweites Band:



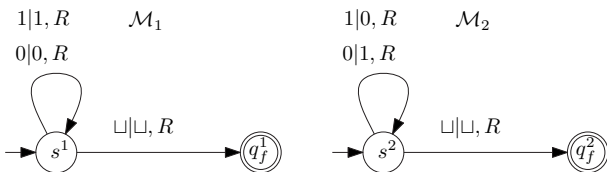
# Beispiel zur Konstruktion aus Teilaufgabe a)



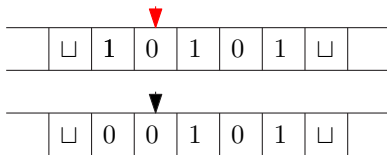
Arbeitsweise der 2-Band-TM:



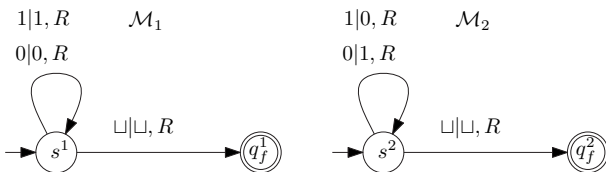
# Beispiel zur Konstruktion aus Teilaufgabe a)



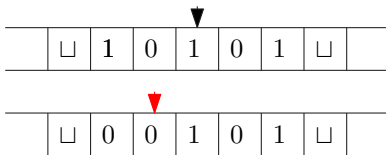
Arbeitsweise der 2-Band-TM:



# Beispiel zur Konstruktion aus Teilaufgabe a)



Arbeitsweise der 2-Band-TM:





## Aufgabe 6 b)

Zeige: Die Menge der semi-entscheidbaren Sprachen ist unter Vereinigung und Schnitt abgeschlossen.

## Aufgabe 6 b)

Zeige: Die Menge der semi-entscheidbaren Sprachen ist unter Vereinigung und Schnitt abgeschlossen.

### Lösung:

- Seien  $M_1$  und  $M_2$  Turingmaschinen, die für Eingaben aus  $L_1$  bzw.  $L_2$  in endlicher Zeit stoppen und nur diese Eingaben akzeptieren
- Aus  $M_1$  und  $M_2$  konstruieren wir 2-Band-Turingmaschine  $M$ , die für Eingaben aus  $L_1 \cup L_2$  bzw.  $L_1 \cap L_2$  in endlicher Zeit stoppt und nur diese Eingaben akzeptiert
- Die Konstruktion ist analog zu Teilaufgabe a), auf Band 1 wird  $M_1$  simuliert, auf Band 2  $M_2$
- $L_1 \cap L_2$ : Akzeptiere, sobald beide Maschinen die Eingabe akzeptieren würden
- $L_1 \cup L_2$ : Akzeptiere, sobald eine der beiden Maschinen die Eingabe akzeptieren würde

## Aufgabe 6 c)

Zeige: Die Menge der semi-entscheidbaren Sprachen ist unter Komplementbildung nicht abgeschlossen.

## Aufgabe 6 c)

Zeige: Die Menge der semi-entscheidbaren Sprachen ist unter Komplementbildung nicht abgeschlossen.

### Lösung:

- Annahme: Die Menge der semi-entscheidbaren Sprachen ist unter Komplementbildung abgeschlossen
- Aus Teilaufgabe a) folgt dann, dass jede semi-entscheidbare Sprache auch entscheidbar ist
- Dies steht im Widerspruch dazu, dass in der Vorlesung gezeigt wurde, dass die universelle Sprache  $L_U$  zwar semi-entscheidbar, aber nicht entscheidbar ist

## Aufgabe 6 d)

Ist die Menge der ungeraden Zahlen entscheidbar? Geben Sie eine kurze Begründung.

(Hinweis: Dabei kann angenommen werden, dass natürliche Zahlen binär kodiert vorliegen.)

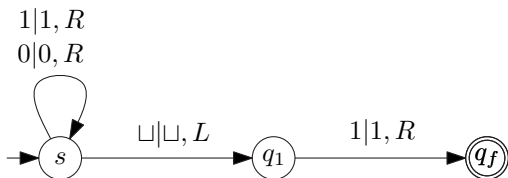
## Aufgabe 6 d)

Ist die Menge der ungeraden Zahlen entscheidbar? Geben Sie eine kurze Begründung.

(Hinweis: Dabei kann angenommen werden, dass natürliche Zahlen binär kodiert vorliegen.)

### Lösung:

Ja, die Menge der ungeraden Zahlen ist entscheidbar, denn man kann leicht eine Turingmaschine angeben, die immer hält und genau die ungeraden Zahlen akzeptiert, z.B.:



## Zusatzaufgabe: Alternative Definition einer NTM

- $\mathcal{M} = (Q, \Sigma, \sqcup, \Gamma, q_0, \delta, q_J, q_N)$
- Funktionsweise wie DTM
- Definition von  $\delta$  in Analogie zu NEAs als Relation auf  $(Q \times \Gamma) \times (Q \times \Gamma \times \{L, R, N\})$

$$\delta : Q \times \Gamma \longrightarrow 2^{Q \times \Gamma \times \{L, R, N\}}$$

- NTM führt im Zustand  $q$  beim Lesen von  $a$  einen der Übergänge in  $\delta(q, a)$  aus.
- i.A. viele Rechenwege für eine Eingabe  $w$  möglich

# Rechenwege einer ANTM für Eingabe $w$ als Baum

Abarbeitung von  $w = ab\dots$

$(q_0)$

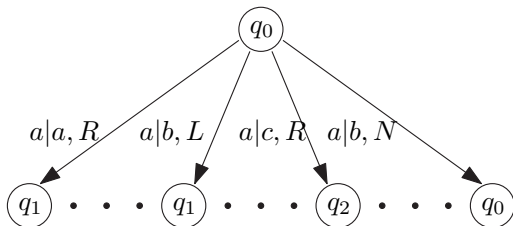
$(q_1, a, R) \in \delta(q_0, a)$

$(q_1, b, L) \in \delta(q_0, a)$

$(q_2, c, R) \in \delta(q_0, a)$

$(q_0, b, N) \in \delta(q_0, a)$

Abarbeitung von  $w = ab\dots$



$$(q_1, a, R) \in \delta(q_0, a)$$

$$(q_1, b, L) \in \delta(q_0, a)$$

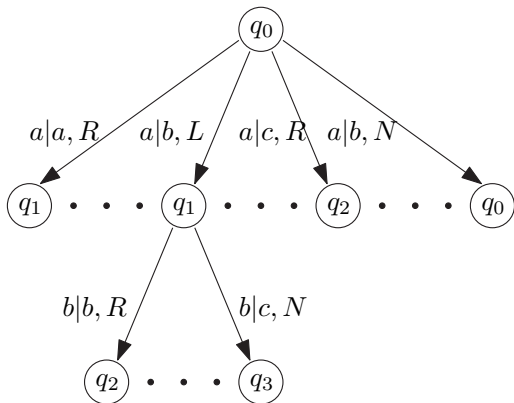
$$(q_2, c, R) \in \delta(q_0, a)$$

$$(q_0, b, N) \in \delta(q_0, a)$$

$$(q_2, b, R) \in \delta(q_1, b)$$

$$(q_2, c, N) \in \delta(q_1, b)$$

Abarbeitung von  $w = ab\dots$



$$(q_1, a, R) \in \delta(q_0, a)$$

$$(q_1, b, L) \in \delta(q_0, a)$$

$$(q_2, c, R) \in \delta(q_0, a)$$

$$(q_0, b, N) \in \delta(q_0, a)$$

$$(q_2, b, R) \in \delta(q_1, b)$$

$$(q_2, c, N) \in \delta(q_1, b)$$

## Alternative Definition einer NTM (Fortsetzung)

- NTM stoppt, falls  $\delta(q, a) = \emptyset$
- NTM akzeptiert  $w$  falls es einen akzeptierenden Rechenweg für  $w$  gibt
- d.h. ein Blatt des Baumes ist akzeptierender Zustand
- wir nennen eine solche NTM hier kurz ANTM
- Zeitkomplexität für  $w \in \Sigma^*$ :  $t(w) =$  Länge der kürzesten akzeptierenden Berechnung für  $w$ , falls  $w \in L$ , 0 sonst
- $T_{\mathcal{M}}(n) = \max\{t(w) : |w| = n\}$

**Definition:**

$\mathcal{NP}$  ist definiert als die Menge der Sprachen, die von ANTM's mit polynomialer Zeitkomplexitätsfunktion akzeptiert werden.

**Zeigen Sie:**  $\mathcal{ANP} = \mathcal{NP}$

# Zeigen Sie: $\mathcal{ANP} = \mathcal{NP}$

- $\subseteq$ :  $L \in \mathcal{ANP}$
- $\Rightarrow$  Es gibt eine ANTM  $\mathcal{A}$ , die  $L$  akzeptiert und ein Polynom  $p$ , so dass für  $w \in L$ :  $T_{\mathcal{A}}(|w|) \leq p(|w|)$
- $\Rightarrow$  Für  $w \in L$  existiert ein akzeptierender Rechenweg der Länge  $\leq p(|w|)$
- Konstruiere NTM  $\mathcal{M}$ , die den Rechenweg rät, und anschließend  $\mathcal{A}$  mit Eingabe  $w$  simuliert.
- Falls Zeitkomplexität von  $\mathcal{M}$  polynomial beschränkt
- $\Rightarrow L \in \mathcal{NP}$

**Zeigen Sie:**  $\mathcal{ANP} = \mathcal{NP}$

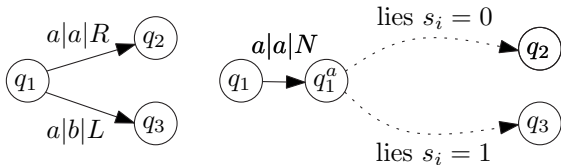
### Kodierung des Rechenwegs:

- Jeder Knoten im „Berechnungsbaum“ hat höchstens  $|Q| \cdot |\Gamma| \cdot 3$  Blätter
- Folge von Binärzahlen  $s_1, \dots, s_k$  in  $\{0, \dots, |Q| \cdot |\Gamma| \cdot 3\}$ ,  $k \leq p(|w|)$
- $s_i = 3$  soll bedeuten, dass im  $i$ -ten Berechnungsschritt die dritte Kante genommen werden soll, falls diese existiert
- $|s_i| \leq \log_2(|Q| \cdot |\Gamma| \cdot 3) = \textit{konst}$
- Kodierung der Länge des Rechenweges  $\leq p(|w|) \cdot \textit{konst}$ , also polynomial

**Zeigen Sie:**  $\mathcal{ANP} = \mathcal{NP}$

### Konstruktion von $\mathcal{M}$ :

- Eingabe  $w$
- Orakel-/Ratephase: Schreibe  $z \in \{0, \dots, |Q| \cdot |\Gamma| \cdot 3\}$  auf das Band und gehe nach links oder rufe Verifikationsphase auf.
- Verifikationsphase: Nach dem Lesen des  $i$ -ten "regulären" Zeichens, lies die  $i$ -te Rechenanweisung des Orakels
- Simuliere  $\mathcal{A}$  mit der  $i$ -ten Rechenanweisung (wir gehen davon aus, dass es eine Ordnung auf den Rechenanweisungen gibt)



- $\mathcal{M}$  ist polynomial beschränkt realisierbar.

## Zeigen Sie: $\mathcal{ANP} = \mathcal{NP}$

- $\supseteq$ :
- $L \in \mathcal{NP}$
  - $\Rightarrow$  Es gibt NTM  $\mathcal{M}$ , die  $w \in L$  in Zeit  $\leq p(|w|)$  für ein Polynom  $p$  akzeptiert
  - $\Rightarrow$  für  $w \in L$  existiert ein Orakelwort  $z$ , so dass  $zw$  in der Verifikationsphase in Zeit  $\leq p(|w|)$  akzeptiert wird.  
Konstruiere ANTM  $\mathcal{A}$ , die nichtdeterministisch ein Orakelwort der Länge  $\leq p(|w|)$  auf das Band schreibt und dann die Verifikationsphase von  $\mathcal{M}$  aufruft

**Zeige:**  $\mathcal{ANP} = \mathcal{NP}$

### Nichtdeterministische Phase von $\mathcal{A}$ :

- Initialisierung:

$$\delta(q_0, a) = \{(q_0, a, L)\}, a \in \Sigma$$

- Schreibe Orakel:

$$\delta(q_0, \sqcup) = \{(q_0, a, L) \mid a \in \Gamma\} \cup \{(q_1, \sqcup, R)\}$$

- $(q_1, \sqcup, R)$  ruft Verifikator auf (deterministische Phase)

