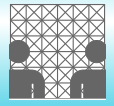


F2 — Automaten und formale Sprachen

Matthias Jantzen
(nach und mit Folienvorlagen von Berndt Farwer)

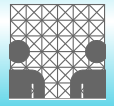
Fachbereich Informatik
AB „Theoretische Grundlagen der Informatik“ (TGI)
Universität Hamburg
jantzen@informatik.uni-hamburg.de

Themen

- Für die heutige Vorlesung geplant:
  - Anwendungen von Abschlusseigenschaften
  - Entscheidungprobleme für endliche Automaten

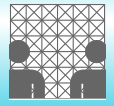
Grenzen von $\mathcal{R}eg$ (Forts.)

- Eine oftmals sehr einfache Methode, von Mengen zu zeigen, dass sie nicht regulär sind, ist die Verwendung der Abschlusseigenschaften von $\mathcal{R}eg$.



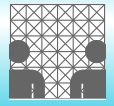
Grenzen von $\mathcal{R}eg$ (Forts.)

- Eine oftmals sehr einfache Methode, von Mengen zu zeigen, dass sie nicht regulär sind, ist die Verwendung der Abschlusseigenschaften von $\mathcal{R}eg$.
  - Zunächst nimmt man an, die fragliche Menge sei regulär.



Grenzen von $\mathcal{R}eg$ (Forts.)

- Eine oftmals sehr einfache Methode, von Mengen zu zeigen, dass sie nicht regulär sind, ist die Verwendung der Abschlusseigenschaften von $\mathcal{R}eg$.
  - Zunächst nimmt man an, die fragliche Menge sei regulär.
  - Dann verwendet man Transformationen mit Abschluss-Operatoren, um solche Sprachen zu erhalten, von denen man schon weiß, oder für die man vielleicht leichter zeigen kann, dass diese nicht regulär sind.

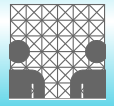


Grenzen von $\mathcal{R}eg$ (Forts.)

- Eine oftmals sehr einfache Methode, von Mengen zu zeigen, dass sie nicht regulär sind, ist die Verwendung der Abschlusseigenschaften von $\mathcal{R}eg$.
  - Zunächst nimmt man an, die fragliche Menge sei regulär.
  - Dann verwendet man Transformationen mit Abschluss-Operatoren, um solche Sprachen zu erhalten, von denen man schon weiß, oder für die man vielleicht leichter zeigen kann, dass diese nicht regulär sind.
  - Somit ist ein Widerspruch erzielt und die ursprüngliche Menge kann nicht regulär gewesen sein.

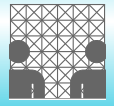
Grenzen von $\mathcal{R}eg$ (Forts.)

- Eine oftmals sehr einfache Methode, von Mengen zu zeigen, dass sie nicht regulär sind, ist die Verwendung der Abschlusseigenschaften von $\mathcal{R}eg$.
  - Zunächst nimmt man an, die fragliche Menge sei regulär.
  - Dann verwendet man Transformationen mit Abschluss-Operatoren, um solche Sprachen zu erhalten, von denen man schon weiß, oder für die man vielleicht leichter zeigen kann, dass diese nicht regulär sind.
  - Somit ist ein Widerspruch erzielt und die ursprüngliche Menge kann nicht regulär gewesen sein.
- Varianten dieses Vorgehens ...



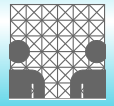
Beispielanwendung: DUP

- Wir zeigen, dass die Menge
  $C := \{w \in \{a,b\}^* \mid |w|_a = |w|_b\}$ nicht regulär ist:



Beispielanwendung: DUP

- Wir zeigen, dass die Menge $C := \{w \in \{a,b\}^* \mid |w|_a = |w|_b\}$ nicht regulär ist:
  - Setze $D := C \cap \{a\}^*\{b\}^*$. Mit $C \in \mathcal{REG}$ wäre auch $D \in \mathcal{REG}$, es ist aber $D = DUP = \{a^n b^n \mid n \in I\!N\}$ bekanntlich nicht regulär.



Beispielanwendung: DUP

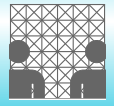
- Wir zeigen, dass die Menge $C := \{w \in \{a,b\}^* \mid |w|_a = |w|_b\}$ nicht regulär ist:
  - Setze $D := C \cap \{a\}^*\{b\}^*$. Mit $C \in \mathcal{REG}$ wäre auch $D \in \mathcal{REG}$, es ist aber $D = DUP = \{a^n b^n \mid n \in \mathbb{N}\}$ bekanntlich nicht regulär.
- Das uvw-Theorem ist nicht direkt benutzbar, um zu beweisen, dass die Sprache $L := \{c^k a^l b^m \mid k,l,m \in \mathbb{N} : k = 0 \text{ oder } l = m\}$ nicht regulär ist.

Beispielanwendung: DUP

- Wir zeigen, dass die Menge $C := \{w \in \{a,b\}^* \mid |w|_a = |w|_b\}$ nicht regulär ist:
  - Setze $D := C \cap \{a\}^*\{b\}^*$. Mit $C \in \mathcal{REG}$ wäre auch $D \in \mathcal{REG}$, es ist aber $D = DUP = \{a^n b^n \mid n \in I\!N\}$ bekanntlich nicht regulär.
- Das uvw-Theorem ist nicht direkt benutzbar, um zu beweisen, dass die Sprache $L := \{c^k a^l b^m \mid k,l,m \in I\!N : k = 0 \text{ oder } l = m\}$ nicht regulär ist.
  - Wir definieren nun die Menge $E := L \cap \{c\}\{a\}^*\{b\}^*$, die regulär wäre, wenn L dies ist.

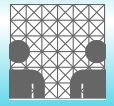
Entscheidbarkeit

- Ein **Entscheidungsproblem** ist eine Fragestellung, die mit **JA** oder **NEIN** beantwortet werden muss.



Entscheidbarkeit

- Ein **Entscheidungsproblem** ist eine Fragestellung, die mit **JA** oder **NEIN** beantwortet werden muss.
- Wir nennen ein Entscheidungsproblem **entscheidbar**, gdw. es einen deterministischen Algorithmus gibt, der die Fragestellung in endlich vielen elementaren Schritten beantwortet.



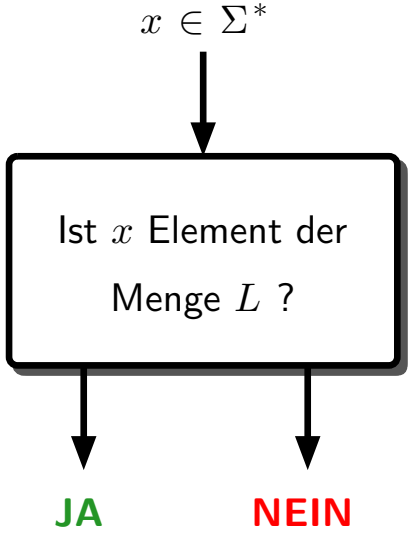
Entscheidbarkeit

- Ein **Entscheidungsproblem** ist eine Fragestellung, die mit **JA** oder **NEIN** beantwortet werden muss.
- Wir nennen ein Entscheidungsproblem **entscheidbar**, gdw. es einen deterministischen Algorithmus gibt, der die Fragestellung in endlich vielen elementaren Schritten beantwortet.
- D.h. sowohl eine **positive** als auch eine **negative** Antwort muss in endlicher Zeit erreicht werden!

Entscheidbarkeit

- Ein **Entscheidungsproblem** ist eine Fragestellung, die mit **JA** oder **NEIN** beantwortet werden muss.
- Wir nennen ein Entscheidungsproblem **entscheidbar**, gdw. es einen deterministischen Algorithmus gibt, der die Fragestellung in endlich vielen elementaren Schritten beantwortet.
- D.h. sowohl eine **positive** als auch eine **negative** Antwort muss in endlicher Zeit erreicht werden!

Entscheidbarkeit/Beweisbarkeit



spezielles Wortproblem

- . . . was sind typische Entscheidungsprobleme?

spezielles Wortproblem

- . . . was sind typische Entscheidungsprobleme?
- **Definition:** (Wortproblem für reguläre Menge L)

  Eingabe: Ein Wort $w \in \Sigma^*$

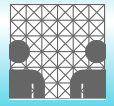
  Frage: Gilt $w \in L$?

spezielles Wortproblem

- . . . was sind typische Entscheidungsprobleme?
- **Definition:** (Wortproblem für reguläre Menge L)

  Eingabe: Ein Wort $w \in \Sigma^*$

  Frage: Gilt $w \in L$?
- . . . ist entscheidbar.

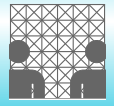


spezielles Wortproblem

- . . . was sind typische Entscheidungsprobleme?
- **Definition:** (Wortproblem für reguläre Menge L)

  Eingabe: Ein Wort $w \in \Sigma^*$

  Frage: Gilt $w \in L$?
- . . . ist entscheidbar.
- Algorithmus: der DFA für die Sprache L.



spezielles Wortproblem

- . . . was sind typische Entscheidungsprobleme?
- **Definition:** (Wortproblem für reguläre Menge L)

  Eingabe: Ein Wort $w \in \Sigma^*$

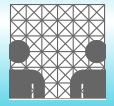
  Frage: Gilt $w \in L$?
- . . . ist entscheidbar.
- Algorithmus: der DFA für die Sprache L.
- Komplexität: Länge des zu prüfenden Wortes w.

allgemeines Wortproblem

- **Definition:** (Allg. Wortproblem für λ-freie NFA's)

  Eingabe: Ein NFA $A = (Z, \Sigma, K, z_0, Z_{\mathsf{end}})$ und ein Wort $w \in \Sigma^*$

  Frage: Gilt $w \in L(A)$?



allgemeines Wortproblem

- **Definition:** (Allg. Wortproblem für λ-freie NFA's)

  Eingabe: Ein NFA $A = (Z, \Sigma, K, z_0, Z_{\text{end}})$ und ein Wort $w \in \Sigma^*$

  Frage: Gilt $w \in L(A)$?

- ...ist natürlich auch entscheidbar.

allgemeines Wortproblem

- **Definition:** (Allg. Wortproblem für λ-freie NFA's)

  Eingabe: Ein NFA $A = (Z, \Sigma, K, z_0, Z_{\text{end}})$ und ein Wort $w \in \Sigma^*$

  Frage: Gilt $w \in L(A)$?

- . . . ist natürlich auch entscheidbar.
- 1. Möglichkeit für einen Algorithmus: Konstruiere aus NFA einen DFA $\Rightarrow$ spezielles Wortproblem.

allgemeines Wortproblem

- **Definition:** (Allg. Wortproblem für λ-freie NFA's)

  Eingabe: Ein NFA $A = (Z, \Sigma, K, z_0, Z_{\text{end}})$ und ein Wort $w \in \Sigma^*$

  Frage: Gilt $w \in L(A)$?

- . . . ist natürlich auch entscheidbar.
- 1. Möglichkeit für einen Algorithmus: Konstruiere aus NFA einen DFA $\Rightarrow$ spezielles Wortproblem.
- Komplexität: exponentiell (Potenzautomatenkonstr.)

allgemeines Wortproblem

- **Definition:** (Allg. Wortproblem für λ-freie NFA's)

  Eingabe: Ein NFA $A = (Z, \Sigma, K, z_0, Z_{\text{end}})$ und ein Wort $w \in \Sigma^*$

  Frage: Gilt $w \in L(A)$?

- . . . ist natürlich auch entscheidbar.
- 1. Möglichkeit für einen Algorithmus: Konstruiere aus NFA einen DFA $\Rightarrow$ spezielles Wortproblem.
- Komplexität: exponentiell (Potenzautomatenkonstr.)
- 2. Möglichkeit: Konstruktion des Potenzautomaten nur für die möglichen Pfade in dem λ-freien NFA entlang des Wortes $w := w_1 w_2 w_3 \ldots w_n$.

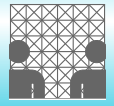
allgemeines Wortproblem

- **Definition:** (Allg. Wortproblem für λ-freie NFA's)

  Eingabe: Ein NFA $A = (Z, \Sigma, K, z_0, Z_{\text{end}})$ und ein Wort $w \in \Sigma^*$

  Frage: Gilt $w \in L(A)$?

- . . . ist natürlich auch entscheidbar.
- 1. Möglichkeit für einen Algorithmus: Konstruiere aus NFA einen DFA $\Rightarrow$ spezielles Wortproblem.
- Komplexität: exponentiell (Potenzautomatenkonstr.)
- 2. Möglichkeit: Konstruktion des Potenzautomaten nur für die möglichen Pfade in dem λ-freien NFA entlang des Wortes $w := w_1 w_2 w_3 \ldots w_n$.
- Komplexität: polynomiell, nämlich $n\,|Z|^2$



Leerheitsproblem

- **Definition:** (Leerheitsproblem für endliche Automaten)

  Eingabe: Ein DFA oder ein NFA A

  Frage: Ist $L(A) = \emptyset$?

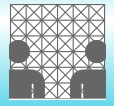
Leerheitsproblem

- **Definition:** (Leerheitsproblem für endliche Automaten)

  Eingabe: Ein DFA oder ein NFA A

  Frage: Ist $L(A) = \emptyset$?

- . . . ist entscheidbar.



Leerheitsproblem

- **Definition:** (Leerheitsproblem für endliche Automaten)

  Eingabe: Ein DFA oder ein NFA A

  Frage: Ist $L(A) = \emptyset$?

- . . . ist entscheidbar.
- Algorithmus: ähnlich dem Algorithmus zum Auffinden der initialen Zusammenhangskomponente.

Leerheitsproblem

- **Definition:** (Leerheitsproblem für endliche Automaten)

  Eingabe: Ein DFA oder ein NFA A

  Frage: Ist $L(A) = \emptyset$?

- . . . ist entscheidbar.
- Algorithmus: ähnlich dem Algorithmus zum Auffinden der initialen Zusammenhangskomponente.
- Komplexität: linear in der Größe des Automaten.

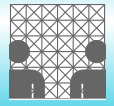
Leerheitsproblem

- **Definition:** (Leerheitsproblem für endliche Automaten)

  Eingabe: Ein DFA oder ein NFA A

  Frage: Ist $L(A) = \emptyset$?

- . . . ist entscheidbar.
- Algorithmus: ähnlich dem Algorithmus zum Auffinden der initialen Zusammenhangskomponente.
- Komplexität: linear in der Größe des Automaten.
- Zur Größe des Automaten gehört: Anzahl der Zustände, Anzahl der Kanten

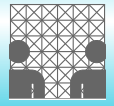


Äquivalenzproblem

- **Definition:** (Äquivalenzproblem für DFAs)

  Eingabe: Zwei DFAs
  $A = (Z_A, \Sigma_A, \delta_A, z_{A,0}, Z_{A,\text{end}})$ und
  $B = (Z_B, \Sigma_B, \delta_B, z_{B,0}, Z_{B,\text{end}})$

  Frage: Gilt $L(A) = L(B)$?



Äquivalenzproblem

- **Definition:** (Äquivalenzproblem für DFAs)

  Eingabe: Zwei DFAs
  $A = (Z_A, \Sigma_A, \delta_A, z_{A,0}, Z_{A,\text{end}})$ und
  $B = (Z_B, \Sigma_B, \delta_B, z_{B,0}, Z_{B,\text{end}})$

  Frage: Gilt $L(A) = L(B)$?

- . . . ist entscheidbar.

Äquivalenzproblem

- **Definition:** (Äquivalenzproblem für DFAs)

  Eingabe: Zwei DFAs
  $A = (Z_A, \Sigma_A, \delta_A, z_{A,0}, Z_{A,\text{end}})$ und
  $B = (Z_B, \Sigma_B, \delta_B, z_{B,0}, Z_{B,\text{end}})$

  Frage: Gilt $L(A) = L(B)$?

- . . . ist entscheidbar.
- Algorithmus: $L(A) = L(B)$ gilt gdw. sowohl $L(A) \cap \overline{L(B)} = \emptyset$ als auch $L(B) \cap \overline{L(A)} = \emptyset$ ist.

Äquivalenzproblem

- **Definition:** (Äquivalenzproblem für DFAs)

  Eingabe: Zwei DFAs
  $A = (Z_A, \Sigma_A, \delta_A, z_{A,0}, Z_{A,\text{end}})$ und
  $B = (Z_B, \Sigma_B, \delta_B, z_{B,0}, Z_{B,\text{end}})$

  Frage: Gilt $L(A) = L(B)$?

- . . . ist entscheidbar.
- Algorithmus: $L(A) = L(B)$ gilt gdw. sowohl $L(A) \cap \overline{L(B)} = \emptyset$ als auch $L(B) \cap \overline{L(A)} = \emptyset$ ist.
  - vervollständige die DFAs

Äquivalenzproblem

- **Definition:** (Äquivalenzproblem für DFAs)

  Eingabe: Zwei DFAs
  $A = (Z_A, \Sigma_A, \delta_A, z_{A,0}, Z_{A,\text{end}})$ und
  $B = (Z_B, \Sigma_B, \delta_B, z_{B,0}, Z_{B,\text{end}})$

  Frage: Gilt $L(A) = L(B)$?

- . . . ist entscheidbar.
- Algorithmus: $L(A) = L(B)$ gilt gdw. sowohl $L(A) \cap \overline{L(B)} = \emptyset$ als auch $L(B) \cap \overline{L(A)} = \emptyset$ ist.
  - vervollständige die DFAs
  - konstruiere DFAs für das Komplement

Äquivalenzproblem

- **Definition:** (Äquivalenzproblem für DFAs)

  Eingabe: Zwei DFAs
  $A = (Z_A, \Sigma_A, \delta_A, z_{A,0}, Z_{A,\text{end}})$ und
  $B = (Z_B, \Sigma_B, \delta_B, z_{B,0}, Z_{B,\text{end}})$

  Frage: Gilt $L(A) = L(B)$?

- . . . ist entscheidbar.
- Algorithmus: $L(A) = L(B)$ gilt gdw. sowohl $L(A) \cap \overline{L(B)} = \emptyset$ als auch $L(B) \cap \overline{L(A)} = \emptyset$ ist.
  - vervollständige die DFAs
  - konstruiere DFAs für das Komplement
  - konstruiere Produktautomaten

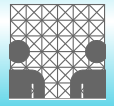
Äquivalenzproblem

- **Definition:** (Äquivalenzproblem für DFAs)

  Eingabe: Zwei DFAs
  $A = (Z_A, \Sigma_A, \delta_A, z_{A,0}, Z_{A,\text{end}})$ und
  $B = (Z_B, \Sigma_B, \delta_B, z_{B,0}, Z_{B,\text{end}})$

  Frage: Gilt $L(A) = L(B)$?

- . . . ist entscheidbar.
- Algorithmus: $L(A) = L(B)$ gilt gdw. sowohl $L(A) \cap \overline{L(B)} = \emptyset$ als auch $L(B) \cap \overline{L(A)} = \emptyset$ ist.
  - vervollständige die DFAs
  - konstruiere DFAs für das Komplement
  - konstruiere Produktautomaten
  - $\Rightarrow$ Leerheitsproblem



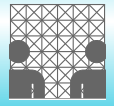
Äquivalenzproblem

- **Definition:** (Äquivalenzproblem für DFAs)

  Eingabe: Zwei DFAs
  $A = (Z_A, \Sigma_A, \delta_A, z_{A,0}, Z_{A,\text{end}})$ und
  $B = (Z_B, \Sigma_B, \delta_B, z_{B,0}, Z_{B,\text{end}})$

  Frage: Gilt $L(A) = L(B)$?

- . . . ist entscheidbar.
- Algorithmus: $L(A) = L(B)$ gilt gdw. sowohl $L(A) \cap \overline{L(B)} = \emptyset$ als auch $L(B) \cap \overline{L(A)} = \emptyset$ ist.
  - vervollständige die DFAs
  - konstruiere DFAs für das Komplement
  - konstruiere Produktautomaten
  - $\Rightarrow$ Leerheitsproblem
- Komplexität: $|\Sigma_A \cap \Sigma_B| \cdot |Z_A| \cdot |Z_B|$ Schritte.

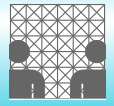


Universalitätsproblem

- **Definition:** (Universalitätsproblem für DFAs)

  Eingabe: Ein DFA $A = (Z, \Sigma, \delta, z_0, Z_{\mathsf{end}})$

  Frage: Ist $L(A) = \Sigma^*$?



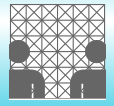
Universalitätsproblem

- **Definition:** (Universalitätsproblem für DFAs)

  Eingabe: Ein DFA $A = (Z, \Sigma, \delta, z_0, Z_{\mathsf{end}})$

  Frage: Ist $L(A) = \Sigma^*$?

- . . . ist entscheidbar.



Universalitätsproblem

- **Definition:** (Universalitätsproblem für DFAs)

  Eingabe: Ein DFA $A = (Z, \Sigma, \delta, z_0, Z_{\text{end}})$

  Frage: Ist $L(A) = \Sigma^*$?

- . . . ist entscheidbar.
- Algorithmus: Ist jeder Zustand Endzustand und verlassen jeden Zustand $|\Sigma|$ Kanten?

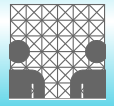
Universalitätsproblem

- **Definition:** (Universalitätsproblem für DFAs)

  Eingabe: Ein DFA $A = (Z, \Sigma, \delta, z_0, Z_{\text{end}})$

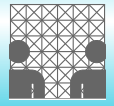
  Frage: Ist $L(A) = \Sigma^*$?

- . . . ist entscheidbar.
- Algorithmus: Ist jeder Zustand Endzustand und verlassen jeden Zustand $|\Sigma|$ Kanten?
- Komplexität: linearer Aufwand



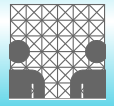
Ausblick

- Für die Definition von Programmiersprachen reichen reguläre Mengen **nicht** aus, deshalb suchen wir nach einer größeren Sprachfamilie ...



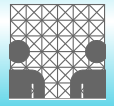
Ausblick

- Für die Definition von Programmiersprachen reichen reguläre Mengen **nicht** aus, deshalb suchen wir nach einer größeren Sprachfamilie ...
- Grammatiken



Ausblick

- Für die Definition von Programmiersprachen reichen reguläre Mengen **nicht** aus, deshalb suchen wir nach einer größeren Sprachfamilie ...
- Grammatiken
- kontextfreie Sprachen



Ausblick

- Für die Definition von Programmiersprachen reichen reguläre Mengen **nicht** aus, deshalb suchen wir nach einer größeren Sprachfamilie ...
- Grammatiken
- kontextfreie Sprachen
- Abschlusseigenschaften