

Theoretische Informatik 2

Sommersemester 2018

Prof. Dr. Georg Schnitger
AG Theoretische Informatik

Goethe-Universität Frankfurt am Main

Theoretische Informatik 2

Sommersemester 2018

Prof. Dr. Georg Schnitger
AG Theoretische Informatik

Goethe-Universität Frankfurt am Main

Herzlich willkommen!!!

Kapitel 1: Einführung

(I) Reguläre Sprachen:

- ▶ DFAs, das stärkste “vollständig beherrschbare” Rechnermodell:
Automatische Verifikation und Minimierung
sind effizient möglich. (Anwendungen z. B. im Entwurf von Schaltungen.)
- ▶ Verschiedene äquivalente Perspektiven:
 - ★ DFAs und NFAs,
 - ★ endliche probabilistische Automaten,
 - ★ endliche Zweiwege-Automaten,
 - ★ reguläre Ausdrücke,
 - ★ reguläre Grammatiken.

(I) Kontextfreie Sprachen:

- ▶ Grundlage des Compilerbaus,
- ▶ Kellerautomaten und deterministisch kontextfreie Sprachen:
 - ★ Effiziente Compiler,
- ▶ Nicht-kontextfreie Sprachen: Ogden's Lemma,
- ▶ Entscheidungsprobleme für kontextfreie Sprachen.

(II) Speicherplatz-Komplexität:

- ▶ Der Speicherverbrauch regulärer, kontextfreier und kontextsensitiver Sprachen,
- ▶ Wie mächtig sind
 - ★ Quantoren in der Aussagenlogik?
 - ★ probabilistische Berechnungen und Quantenberechnungen?
 - ★ die Bestimmung von Gewinnstrategien in nicht-trivialen 2-Personen Spielen?

(II) Parallelität:

- ▶ Welche Probleme in P besitzen super-schnelle parallele Algorithmen?
- ▶ Zusammenhang zwischen Speicherplatz und paralleler Rechenzeit.

(III) Algorithmische Fragestellungen der Logik

- ▶ Frege Systeme.
- ▶ Resolution und DPLL-Verfahren,
 - ★ KNF-SAT ist NP-vollständig, aber **SAT-Solver** können trotzdem viele KNF-SAT-Instanzen knacken.
 - ★ Worauf beruhen SAT-Solver? Was sind ihre Stärken und Schwächen?
- ▶ Model Checking: Hält ein Entwurf die gewünschten Spezifikationen ein?
 - ★ Computation Tree Logic (CTL), eine temporale Aussagenlogik für ein **effizientes Model Checking**.
- ▶ Prädikatenlogik:
 - ★ Wie schwierig ist automatisches Beweisen?
 - ★ Die **Gödelschen (Un-)Vollständigkeitssätze**: Was kann die Prädikatenlogik und was nicht?

- **Diskrete Modellierung:**

- ▶ Beweismethoden: direkte und indirekte Beweise, vollständige Induktion,
- ▶ eine erste Behandlung von endlichen Automaten und kontextfreien Sprachen, Aussagen- und Prädikatenlogik.

- **Datenstrukturen und Theoretische Informatik 1:**

- ▶ Laufzeitanalyse: O , o , Ω , ω and Rekursionsgleichungen,
- ▶ Graphalgorithmen,
- ▶ NP-Vollständigkeit,
- ▶ Berechenbarkeit.

- Skript zur Vorlesung „Theoretische Informatik 2“, Goethe-Universität Frankfurt.
- M. Sipser, Introduction to the Theory of Computation, Paperback 3rd edition, Cengage Learning, 2012.
- U. Schöning, Theoretische Informatik - kurzgefasst, Spektrum 2008.
- I. Wegener, Theoretische Informatik: Eine algorithmenorientierte Einführung, B.G. Teubner 1999 (2. Auflage).
- Sanjeev Arora und Boaz Barak, Computational Complexity, a modern approach, Cambridge University Press 2009.
- J. Shallit, A second course in Formal Languages and Automata Theory, Cambridge University Press, 2008.
- J.E. Hopcroft, J.D. Ullman, R. Motwani, Introduction to Automata Theory, Languages and Computation, Addison-Wesley, 2013.

Organisatorisches

```
http://www.thi.informatik.uni-frankfurt.de/lehre/gl2/sose18.de
```

Die Webseite enthält alle wichtigen Informationen zur Veranstaltung:

- Alle Vorlesungsmaterialien ([Skript](#), [Folien](#), Zugang zu [Extra-Materialien](#)) finden Sie auf dieser Seite.
- Auch organisatorische Details zum [Übungsbetrieb](#) werden beschrieben.
- Unter **Aktuelles** finden Sie zum Beispiel:
 - ▶ Anmerkungen zum Übungsbetrieb
 - ▶ und gegebenenfalls Anmerkungen zu aktuellen Übungsaufgaben.
- Im **Logbuch** finden Sie Informationen,
 - ▶ Beamer-Folien, weitere Referenzen zu den einzelnen Vorlesungsstunden.

BITTE, BITTE, BITTE aktiv an den Übungen teilnehmen!

- Wöchentliche Übungszettel auf der Webseite mit Ausgabe am Dienstag. Der erste Übungszettel wird in der zweiten Vorlesungswoche ausgegeben.
- Rückgabe, nach 1-wöchiger Bearbeitungszeit, zu Beginn der Dienstag-Vorlesung. (Rückgabe auch im Briefkasten neben Büro 312 möglich.)
- Wenn die Prüfung bestanden ist:
 - ▶ Bei mindestens 50% aller Übungspunkte eine Verbesserung um einen Notenschnitt,
 - ▶ bei mindestens 70% eine Verbesserung um zwei Notenschritte.
- Bei nicht zu vielen Teilnehmern werden nur mündliche Prüfungen angeboten, sonst eine Klausur.

BITTE, BITTE, BITTE

- 1 Bitte helfen Sie mir durch
 - ▶ Fragen,
 - ▶ Kommentare
 - ▶ und Antworten!
- 2 Die Vorlesung kann nur durch **Interaktion** interessant werden.
 - ▶ Ich muss wissen, wo der Schuh drückt.
- 3 Sie erreichen mich außerhalb der Vorlesung im Büro 303.
 - ▶ Sprechstunde: Dienstags 10-12.
 - ▶ Kommen Sie vorbei.

- **„Bachelor Informatik“** oder **„Bachelor Bioinformatik“**:
Theoretische Informatik 2 (4+2 SWS, 10 CP). Auf Wunsch wie bisher als 3+2 SWS Veranstaltung mit 8 CP bis 2.5 Wochen vor Vorlesungsende.
- **„Neuer Master Informatik“** oder **„Master Wirtschaftsinformatik“**:
 - ▶ Theoretische Informatik 2: Grundlagen (5 CP).
 - ★ Die Veranstaltung findet vom 18.04 bis zum 31.05 statt.
 - ▶ Theoretische Informatik 2: Weiterführende Themen (5 CP)
 - ★ Die Veranstaltung findet vom 06.06 bis zum 19.07 statt.
 - ▶ Als Veranstaltung vom 18.4 bis zum 19.07 (10 CP).
- **„Alter Master Informatik“**:
Theoretische Informatik 2 (4+2 SWS, 10 CP).

Grundbegriffe zu “Worten und Sprachen”

- 1 $\mathbb{N} := \{0, 1, 2, 3, \dots\}$ ist die Menge aller natürlichen Zahlen.
 $\mathbb{N}_{>0} := \{1, 2, 3, \dots\}$ ist die Menge aller positiven natürlichen Zahlen.
 \mathbb{Q} ist die Menge der rationalen und \mathbb{R} die Menge der reellen Zahlen.

- ① $\mathbb{N} := \{0, 1, 2, 3, \dots\}$ ist die Menge aller natürlichen Zahlen.
 $\mathbb{N}_{>0} := \{1, 2, 3, \dots\}$ ist die Menge aller positiven natürlichen Zahlen.
 \mathbb{Q} ist die Menge der rationalen und \mathbb{R} die Menge der reellen Zahlen.
- ② Ein Alphabet Σ ist eine endliche, nicht-leere Menge von Buchstaben.
 - ▶ $\Sigma^n = \{a_1 \cdots a_n \mid a_1, \dots, a_n \in \Sigma\}$ ist die Menge aller Worte der Länge n über Σ .
 - ▶ $\Sigma^0 = \{\varepsilon\}$ besteht nur aus dem **leeren Wort** ε .
 - ▶ $\Sigma^* = \bigcup_{n \in \mathbb{N}} \Sigma^n$ ist die Menge aller Worte über dem Alphabet Σ .
 - ▶ $\Sigma^+ = \bigcup_{n \in \mathbb{N}_{>0}} \Sigma^n$ ist die Menge aller nicht-leeren Worte über Σ .

- 1 $\mathbb{N} := \{0, 1, 2, 3, \dots\}$ ist die Menge aller natürlichen Zahlen.
 $\mathbb{N}_{>0} := \{1, 2, 3, \dots\}$ ist die Menge aller positiven natürlichen Zahlen.
 \mathbb{Q} ist die Menge der rationalen und \mathbb{R} die Menge der reellen Zahlen.
- 2 Ein Alphabet Σ ist eine endliche, nicht-leere Menge von Buchstaben.
 - ▶ $\Sigma^n = \{a_1 \cdots a_n \mid a_1, \dots, a_n \in \Sigma\}$ ist die Menge aller Worte der Länge n über Σ .
 - ▶ $\Sigma^0 = \{\varepsilon\}$ besteht nur aus dem **leeren Wort** ε .
 - ▶ $\Sigma^* = \bigcup_{n \in \mathbb{N}} \Sigma^n$ ist die Menge aller Worte über dem Alphabet Σ .
 - ▶ $\Sigma^+ = \bigcup_{n \in \mathbb{N}_{>0}} \Sigma^n$ ist die Menge aller nicht-leeren Worte über Σ .
- 3 Für $w \in \Sigma^*$ ist $|w|$ die Länge von w , d.h. die Anzahl der Buchstaben in w .
Für $a \in \Sigma$ ist $|w|_a$ die Anzahl der Vorkommen des Buchstabens a in w .

- 1 $\mathbb{N} := \{0, 1, 2, 3, \dots\}$ ist die Menge aller natürlichen Zahlen.
 $\mathbb{N}_{>0} := \{1, 2, 3, \dots\}$ ist die Menge aller positiven natürlichen Zahlen.
 \mathbb{Q} ist die Menge der rationalen und \mathbb{R} die Menge der reellen Zahlen.
- 2 Ein Alphabet Σ ist eine endliche, nicht-leere Menge von Buchstaben.
 - ▶ $\Sigma^n = \{a_1 \cdots a_n \mid a_1, \dots, a_n \in \Sigma\}$ ist die Menge aller Worte der Länge n über Σ .
 - ▶ $\Sigma^0 = \{\varepsilon\}$ besteht nur aus dem **leeren Wort** ε .
 - ▶ $\Sigma^* = \bigcup_{n \in \mathbb{N}} \Sigma^n$ ist die Menge aller Worte über dem Alphabet Σ .
 - ▶ $\Sigma^+ = \bigcup_{n \in \mathbb{N}_{>0}} \Sigma^n$ ist die Menge aller nicht-leeren Worte über Σ .
- 3 Für $w \in \Sigma^*$ ist $|w|$ die Länge von w , d.h. die Anzahl der Buchstaben in w .
Für $a \in \Sigma$ ist $|w|_a$ die Anzahl der Vorkommen des Buchstabens a in w .

Eine **Sprache** L (über Σ) ist eine Teilmenge von Σ^* .

- Die Menge der im **Duden** aufgeführten Worte über dem Alphabet

$$\{a, A, \dots, z, Z, \ddot{a}, \ddot{A}, \dots, \ddot{u}, \ddot{U}, \beta, -, \dots\}.$$

- **Deutsch** besteht aus allen syntaktisch korrekt aufgebauten und semantisch sinnvollen Sätzen mit Worten aus dem Duden.
- **C++** ist die Menge aller syntaktisch richtig aufgebauten C++ Programme. Das Alphabet ist

- Die Menge der im **Duden** aufgeführten Worte über dem Alphabet

$$\{a, A, \dots, z, Z, \ddot{a}, \ddot{A}, \dots, \ddot{u}, \ddot{U}, \beta, \text{ , } -\}.$$

- **Deutsch** besteht aus allen syntaktisch korrekt aufgebauten und semantisch sinnvollen Sätzen mit Worten aus dem Duden.
- **C++** ist die Menge aller syntaktisch richtig aufgebauten C++ Programme. Das Alphabet ist die Menge aller ASCII-Symbole.

- Die Menge der im **Duden** aufgeführten Worte über dem Alphabet

$$\{a, A, \dots, z, Z, \ddot{a}, \ddot{A}, \dots, \ddot{u}, \ddot{U}, \beta, -\}.$$

- **Deutsch** besteht aus allen syntaktisch korrekt aufgebauten und semantisch sinnvollen Sätzen mit Worten aus dem Duden.
- **C++** ist die Menge aller syntaktisch richtig aufgebauten C++ Programme. Das Alphabet ist die Menge aller ASCII-Symbole.
- Die Sprache der **arithmetischen Ausdrücke** mit den Variablen x und y besteht aus allen arithmetischen Ausdrücken über dem Alphabet

- Die Menge der im **Duden** aufgeführten Worte über dem Alphabet

$$\{a, A, \dots, z, Z, \ddot{a}, \ddot{A}, \dots, \ddot{u}, \ddot{U}, \beta, -, \dots\}.$$

- **Deutsch** besteht aus allen syntaktisch korrekt aufgebauten und semantisch sinnvollen Sätzen mit Worten aus dem Duden.
- **C++** ist die Menge aller syntaktisch richtig aufgebauten C++ Programme. Das Alphabet ist die Menge aller ASCII-Symbole.
- Die Sprache der **arithmetischen Ausdrücke** mit den Variablen x und y besteht aus allen arithmetischen Ausdrücken über dem Alphabet

$$\{x, y, +, *, -, /, (,)\}.$$

- Die Menge der im **Duden** aufgeführten Worte über dem Alphabet

$$\{a, A, \dots, z, Z, \ddot{a}, \ddot{A}, \dots, \ddot{u}, \ddot{U}, \beta, -\}.$$

- **Deutsch** besteht aus allen syntaktisch korrekt aufgebauten und semantisch sinnvollen Sätzen mit Worten aus dem Duden.
- **C++** ist die Menge aller syntaktisch richtig aufgebauten C++ Programme. Das Alphabet ist die Menge aller ASCII-Symbole.
- Die Sprache der **arithmetischen Ausdrücke** mit den Variablen x und y besteht aus allen arithmetischen Ausdrücken über dem Alphabet

$$\{x, y, +, *, -, /, (,)\}.$$

- Weitere Beispielsprachen:
 - ▶ Die Menge aller **HTML-Dokumente**,
 - ▶ die Menge aller **XML-Dokumente**.

Operationen auf Sprachen

Sei Σ ein Alphabet, $u = u_1 \cdots u_n$ und $v = v_1 \cdots v_m$ seien Worte über Σ .

Operationen auf Sprachen

Sei Σ ein Alphabet, $u = u_1 \cdots u_n$ und $v = v_1 \cdots v_m$ seien Worte über Σ .

- 1 $uv = u_1 \cdots u_n \cdot v_1 \cdots v_m$ ist die Konkatenation von u und v .

Operationen auf Sprachen

Sei Σ ein Alphabet, $u = u_1 \cdots u_n$ und $v = v_1 \cdots v_m$ seien Worte über Σ .

- 1 $uv = u_1 \cdots u_n \cdot v_1 \cdots v_m$ ist die Konkatenation von u und v .
- 2 Für Sprachen L_1, L_2 über Σ ist

$$L_1 \circ L_2 = \{uv \mid u \in L_1, v \in L_2\}$$

die Konkatenation von L_1 und L_2 . Oft schreiben wir kurz L_1L_2 oder $L_1 \cdot L_2$ statt $L_1 \circ L_2$.

Operationen auf Sprachen

Sei Σ ein Alphabet, $u = u_1 \cdots u_n$ und $v = v_1 \cdots v_m$ seien Worte über Σ .

- 1 $uv = u_1 \cdots u_n \cdot v_1 \cdots v_m$ ist die Konkatenation von u und v .
- 2 Für Sprachen L_1, L_2 über Σ ist

$$L_1 \circ L_2 = \{uv \mid u \in L_1, v \in L_2\}$$

die Konkatenation von L_1 und L_2 . Oft schreiben wir kurz L_1L_2 oder $L_1 \cdot L_2$ statt $L_1 \circ L_2$.

- 3 Für eine Sprache L über Σ ist

$$\begin{aligned} L^n &= \{u_1 \cdots u_n \mid u_1, \dots, u_n \in L\} \\ L^* &= \bigcup_{n \in \mathbb{N}} L^n \quad (\text{mit } L^0 := \{\epsilon\}) \end{aligned}$$

L^* ist die Kleene-Hülle (oder Kleene-Stern) von L .

- 1 Die Menge aller Felder eines Schachbretts ist

- 1 Die Menge aller Felder eines Schachbretts ist

$$\{A, B, C, D, E, F, G, H\} \circ \{1, 2, 3, 4, 5, 6, 7, 8\}.$$

- 1 Die Menge aller Felder eines Schachbretts ist

$$\{A, B, C, D, E, F, G, H\} \circ \{1, 2, 3, 4, 5, 6, 7, 8\}.$$

- 2 Die Menge aller Karten eines Skatblatts ist

- 1 Die Menge aller Felder eines Schachbretts ist

$$\{A, B, C, D, E, F, G, H\} \circ \{1, 2, 3, 4, 5, 6, 7, 8\}.$$

- 2 Die Menge aller Karten eines Skatblatts ist

$$\{\clubsuit, \spadesuit, \heartsuit, \diamondsuit\} \circ \{7, 8, 9, 10, \text{Bube}, \text{Dame}, \text{König}, \text{Ass}\}.$$

- 1 Die Menge aller Felder eines Schachbretts ist

$$\{A, B, C, D, E, F, G, H\} \circ \{1, 2, 3, 4, 5, 6, 7, 8\}.$$

- 2 Die Menge aller Karten eines Skatblatts ist

$$\{\clubsuit, \spadesuit, \heartsuit, \diamondsuit\} \circ \{7, 8, 9, 10, \text{Bube}, \text{Dame}, \text{König}, \text{Ass}\}.$$

- 3 Die Menge der Binärdarstellungen der natürlichen Zahlen größer Null ist

- 1 Die Menge aller Felder eines Schachbretts ist

$$\{A, B, C, D, E, F, G, H\} \circ \{1, 2, 3, 4, 5, 6, 7, 8\}.$$

- 2 Die Menge aller Karten eines Skatblatts ist

$$\{\clubsuit, \spadesuit, \heartsuit, \diamondsuit\} \circ \{7, 8, 9, 10, \text{Bube}, \text{Dame}, \text{König}, \text{Ass}\}.$$

- 3 Die Menge der Binärdarstellungen der natürlichen Zahlen größer Null ist

$$\{1\} \circ \{0, 1\}^*.$$

- 1 Die Menge aller Felder eines Schachbretts ist

$$\{A, B, C, D, E, F, G, H\} \circ \{1, 2, 3, 4, 5, 6, 7, 8\}.$$

- 2 Die Menge aller Karten eines Skatblatts ist

$$\{\clubsuit, \spadesuit, \heartsuit, \diamondsuit\} \circ \{7, 8, 9, 10, \text{Bube}, \text{Dame}, \text{König}, \text{Ass}\}.$$

- 3 Die Menge der Binärdarstellungen der natürlichen Zahlen größer Null ist

$$\{1\} \circ \{0, 1\}^*.$$

- 4 $\bigcup_{i=1}^4 \{ \cdot, - \}^i$ ist die Menge der Kodierungen von Buchstaben im Morsealphabet.

- 1 Die Menge aller Felder eines Schachbretts ist

$$\{A, B, C, D, E, F, G, H\} \circ \{1, 2, 3, 4, 5, 6, 7, 8\}.$$

- 2 Die Menge aller Karten eines Skatblatts ist

$$\{\clubsuit, \spadesuit, \heartsuit, \diamondsuit\} \circ \{7, 8, 9, 10, \text{Bube, Dame, König, Ass}\}.$$

- 3 Die Menge der Binärdarstellungen der natürlichen Zahlen größer Null ist

$$\{1\} \circ \{0, 1\}^*.$$

- 4 $\bigcup_{i=1}^4 \{ \cdot, - \}^i$ ist die Menge der Kodierungen von Buchstaben im Morsealphabet.

- 5 Die Menge der Uhrzeiten eines Tages ist

- 1 Die Menge aller Felder eines Schachbretts ist

$$\{A, B, C, D, E, F, G, H\} \circ \{1, 2, 3, 4, 5, 6, 7, 8\}.$$

- 2 Die Menge aller Karten eines Skatblatts ist

$$\{\clubsuit, \spadesuit, \heartsuit, \diamondsuit\} \circ \{7, 8, 9, 10, \text{Bube, Dame, König, Ass}\}.$$

- 3 Die Menge der Binärdarstellungen der natürlichen Zahlen größer Null ist

$$\{1\} \circ \{0, 1\}^*.$$

- 4 $\bigcup_{i=1}^4 \{ \cdot, - \}^i$ ist die Menge der Kodierungen von Buchstaben im Morsealphabet.

- 5 Die Menge der Uhrzeiten eines Tages ist

$$(\{2\} \circ \{0, 1, 2, 3\} \cup$$

- 1 Die Menge aller Felder eines Schachbretts ist

$$\{A, B, C, D, E, F, G, H\} \circ \{1, 2, 3, 4, 5, 6, 7, 8\}.$$

- 2 Die Menge aller Karten eines Skatblatts ist

$$\{\clubsuit, \spadesuit, \heartsuit, \diamondsuit\} \circ \{7, 8, 9, 10, \text{Bube, Dame, König, Ass}\}.$$

- 3 Die Menge der Binärdarstellungen der natürlichen Zahlen größer Null ist

$$\{1\} \circ \{0, 1\}^*.$$

- 4 $\bigcup_{i=1}^4 \{., -\}^i$ ist die Menge der Kodierungen von Buchstaben im Morsealphabet.

- 5 Die Menge der Uhrzeiten eines Tages ist

$$(\{2\} \circ \{0, 1, 2, 3\} \cup \{\epsilon, 1\} \circ \{0, 1, \dots, 9\}) \circ \{:\}$$

- 1 Die Menge aller Felder eines Schachbretts ist

$$\{A, B, C, D, E, F, G, H\} \circ \{1, 2, 3, 4, 5, 6, 7, 8\}.$$

- 2 Die Menge aller Karten eines Skatblatts ist

$$\{\clubsuit, \spadesuit, \heartsuit, \diamondsuit\} \circ \{7, 8, 9, 10, \text{Bube, Dame, König, Ass}\}.$$

- 3 Die Menge der Binärdarstellungen der natürlichen Zahlen größer Null ist

$$\{1\} \circ \{0, 1\}^*.$$

- 4 $\bigcup_{i=1}^4 \{., -\}^i$ ist die Menge der Kodierungen von Buchstaben im Morsealphabet.

- 5 Die Menge der Uhrzeiten eines Tages ist

$$(\{2\} \circ \{0, 1, 2, 3\} \cup \{\epsilon, 1\} \circ \{0, 1, \dots, 9\}) \circ \{:\} \circ \{0, 1, \dots, 5\} \circ \{0, 1, \dots, 9\}.$$

Kompakte Beschreibung von Sprachen und Mengen

- 1 Die Menge aller Felder eines Schachbretts ist

$$\{A, B, C, D, E, F, G, H\} \circ \{1, 2, 3, 4, 5, 6, 7, 8\}.$$

- 2 Die Menge aller Karten eines Skatblatts ist

$$\{\clubsuit, \spadesuit, \heartsuit, \diamondsuit\} \circ \{7, 8, 9, 10, \text{Bube, Dame, König, Ass}\}.$$

- 3 Die Menge der Binärdarstellungen der natürlichen Zahlen größer Null ist

$$\{1\} \circ \{0, 1\}^*.$$

- 4 $\bigcup_{i=1}^4 \{., -\}^i$ ist die Menge der Kodierungen von Buchstaben im Morsealphabet.

- 5 Die Menge der Uhrzeiten eines Tages ist

$$(\{2\} \circ \{0, 1, 2, 3\} \cup \{\epsilon, 1\} \circ \{0, 1, \dots, 9\}) \circ \{:\} \circ \{0, 1, \dots, 5\} \circ \{0, 1, \dots, 9\}.$$

- 6 Und die amerikanische Entsprechung ist

Kompakte Beschreibung von Sprachen und Mengen

- 1 Die Menge aller Felder eines Schachbretts ist

$$\{A, B, C, D, E, F, G, H\} \circ \{1, 2, 3, 4, 5, 6, 7, 8\}.$$

- 2 Die Menge aller Karten eines Skatblatts ist

$$\{\clubsuit, \spadesuit, \heartsuit, \diamondsuit\} \circ \{7, 8, 9, 10, \text{Bube, Dame, König, Ass}\}.$$

- 3 Die Menge der Binärdarstellungen der natürlichen Zahlen größer Null ist

$$\{1\} \circ \{0, 1\}^*.$$

- 4 $\bigcup_{i=1}^4 \{ \cdot, - \}^i$ ist die Menge der Kodierungen von Buchstaben im Morsealphabet.

- 5 Die Menge der Uhrzeiten eines Tages ist

$$(\{2\} \circ \{0, 1, 2, 3\} \cup \{\epsilon, 1\} \circ \{0, 1, \dots, 9\}) \circ \{:\} \circ \{0, 1, \dots, 5\} \circ \{0, 1, \dots, 9\}.$$

- 6 Und die amerikanische Entsprechung ist

$$(\{1\} \circ \{0, 1, 2\} \cup \{1, \dots, 9\}) \circ \{:\} \circ$$

Kompakte Beschreibung von Sprachen und Mengen

- 1 Die Menge aller Felder eines Schachbretts ist

$$\{A, B, C, D, E, F, G, H\} \circ \{1, 2, 3, 4, 5, 6, 7, 8\}.$$

- 2 Die Menge aller Karten eines Skatblatts ist

$$\{\clubsuit, \spadesuit, \heartsuit, \diamondsuit\} \circ \{7, 8, 9, 10, \text{Bube, Dame, König, Ass}\}.$$

- 3 Die Menge der Binärdarstellungen der natürlichen Zahlen größer Null ist

$$\{1\} \circ \{0, 1\}^*.$$

- 4 $\bigcup_{i=1}^4 \{ \cdot, - \}^i$ ist die Menge der Kodierungen von Buchstaben im Morsealphabet.

- 5 Die Menge der Uhrzeiten eines Tages ist

$$(\{2\} \circ \{0, 1, 2, 3\} \cup \{\epsilon, 1\} \circ \{0, 1, \dots, 9\}) \circ \{:\} \circ \{0, 1, \dots, 5\} \circ \{0, 1, \dots, 9\}.$$

- 6 Und die amerikanische Entsprechung ist

$$(\{1\} \circ \{0, 1, 2\} \cup \{1, \dots, 9\}) \circ \{:\} \circ \{0, 1, \dots, 5\} \circ \{0, 1, \dots, 9\} \circ \{am, pm\}.$$

Aus welchen Worten besteht die Sprache

$$K = (\{a\}^* \circ \{b\}^* \circ \{d\}^* \circ \{c\}^*)^*$$

Aus welchen Worten besteht die Sprache

$$K = (\{a\}^* \circ \{b\}^* \circ \{d\}^* \circ \{c\}^*)^*$$

- Behauptung: $K \subseteq \{a, b, c, d\}^*$

Aus welchen Worten besteht die Sprache

$$K = (\{a\}^* \circ \{b\}^* \circ \{d\}^* \circ \{c\}^*)^*$$

- Behauptung: $K \subseteq \{a, b, c, d\}^*$
 - ▶ K besteht nur aus Worten über dem Alphabet $\{a, b, c, d\}$.
 - ▶ $\{a, b, c, d\}^*$ besteht aus allen Worten über $\{a, b, c, d\}$.

Aus welchen Worten besteht die Sprache

$$K = (\{a\}^* \circ \{b\}^* \circ \{d\}^* \circ \{c\}^*)^*$$

- Behauptung: $K \subseteq \{a, b, c, d\}^*$
 - ▶ K besteht nur aus Worten über dem Alphabet $\{a, b, c, d\}$.
 - ▶ $\{a, b, c, d\}^*$ besteht aus allen Worten über $\{a, b, c, d\}$. ✓
- Behauptung: $\{a, b, c, d\}^* \subseteq K$

Aus welchen Worten besteht die Sprache

$$K = (\{a\}^* \circ \{b\}^* \circ \{d\}^* \circ \{c\}^*)^*$$

- Behauptung: $K \subseteq \{a, b, c, d\}^*$
 - ▶ K besteht nur aus Worten über dem Alphabet $\{a, b, c, d\}$.
 - ▶ $\{a, b, c, d\}^*$ besteht aus allen Worten über $\{a, b, c, d\}$. ✓
- Behauptung: $\{a, b, c, d\}^* \subseteq K$
 - ▶ Der letzte $*$ in der Definition von K ist mächtig!
 - ▶ Sei $w = w_1 \cdots w_n$ ein beliebiges Wort in $\{a, b, c, d\}^*$:

Aus welchen Worten besteht die Sprache

$$K = (\{a\}^* \circ \{b\}^* \circ \{d\}^* \circ \{c\}^*)^*$$

- Behauptung: $K \subseteq \{a, b, c, d\}^*$
 - ▶ K besteht nur aus Worten über dem Alphabet $\{a, b, c, d\}$.
 - ▶ $\{a, b, c, d\}^*$ besteht aus allen Worten über $\{a, b, c, d\}$. ✓
 - Behauptung: $\{a, b, c, d\}^* \subseteq K$
 - ▶ Der letzte $*$ in der Definition von K ist mächtig!
 - ▶ Sei $w = w_1 \cdots w_n$ ein beliebiges Wort in $\{a, b, c, d\}^*$:
 - ★ Jeder Buchstabe gehört zu $(\{a\}^* \circ \{b\}^* \circ \{d\}^* \circ \{c\}^*)$.
 - ★ Also ist $w = w_1 \cdots w_n \in (\{a\}^* \circ \{b\}^* \circ \{d\}^* \circ \{c\}^*)^n$
- Und deshalb ist $w \in (\{a\}^* \circ \{b\}^* \circ \{d\}^* \circ \{c\}^*)^*$

Aus welchen Worten besteht die Sprache

$$K = (\{a\}^* \circ \{b\}^* \circ \{d\}^* \circ \{c\}^*)^*$$

- Behauptung: $K \subseteq \{a, b, c, d\}^*$
 - ▶ K besteht nur aus Worten über dem Alphabet $\{a, b, c, d\}$.
 - ▶ $\{a, b, c, d\}^*$ besteht aus allen Worten über $\{a, b, c, d\}$. ✓
 - Behauptung: $\{a, b, c, d\}^* \subseteq K$
 - ▶ Der letzte $*$ in der Definition von K ist mächtig!
 - ▶ Sei $w = w_1 \cdots w_n$ ein beliebiges Wort in $\{a, b, c, d\}^*$:
 - ★ Jeder Buchstabe gehört zu $(\{a\}^* \circ \{b\}^* \circ \{d\}^* \circ \{c\}^*)$.
 - ★ Also ist $w = w_1 \cdots w_n \in (\{a\}^* \circ \{b\}^* \circ \{d\}^* \circ \{c\}^*)^n$
- Und deshalb ist $w \in (\{a\}^* \circ \{b\}^* \circ \{d\}^* \circ \{c\}^*)^* = K$. ✓

Rekursive Definitionen von Sprachen

Die Sprache L ist die kleinste Menge von Worten über dem Alphabet $\Sigma = \{0, 1\}$ mit den Eigenschaften:

Basisregel : $\epsilon \in L$,

Rekursive Regel : wenn $u \in L$, dann $0u \in L$ und $u1 \in L$.

Beschreibe L .

Rekursive Definitionen von Sprachen

Die Sprache L ist die kleinste Menge von Worten über dem Alphabet $\Sigma = \{0, 1\}$ mit den Eigenschaften:

Basisregel : $\epsilon \in L$,

Rekursive Regel : wenn $u \in L$, dann $0u \in L$ und $u1 \in L$.

Beschreibe L .

(1) Behauptung: $\{0\}^* \{1\}^* \subseteq L$.

Rekursive Definitionen von Sprachen

Die Sprache L ist die kleinste Menge von Worten über dem Alphabet $\Sigma = \{0, 1\}$ mit den Eigenschaften:

Basisregel : $\epsilon \in L$,

Rekursive Regel : wenn $u \in L$, dann $0u \in L$ und $u1 \in L$.

Beschreibe L .

(1) Behauptung: $\{0\}^* \{1\}^* \subseteq L$.

▶ Sei $w \in \{0\}^* \{1\}^*$, also $w = 0^n 1^m$ für $n, m \in \mathbb{N}$.

Rekursive Definitionen von Sprachen

Die Sprache L ist die kleinste Menge von Worten über dem Alphabet $\Sigma = \{0, 1\}$ mit den Eigenschaften:

Basisregel : $\epsilon \in L$,

Rekursive Regel : wenn $u \in L$, dann $0u \in L$ und $u1 \in L$.

Beschreibe L .

(1) Behauptung: $\{0\}^* \{1\}^* \subseteq L$.

- ▶ Sei $w \in \{0\}^* \{1\}^*$, also $w = 0^n 1^m$ für $n, m \in \mathbb{N}$.
- ▶ Es ist $\epsilon \in L$ und deshalb $0\epsilon, 00\epsilon, \dots, 0^n\epsilon = 0^n \in L$.
- ▶ Dann aber auch $0^n1, 0^n11, \dots, 0^n1^m \in L$ und deshalb ist $w \in L$ ✓.

Formalisiere mit vollständiger Induktion.

Rekursive Definitionen von Sprachen

Die Sprache L ist die kleinste Menge von Worten über dem Alphabet $\Sigma = \{0, 1\}$ mit den Eigenschaften:

Basisregel : $\epsilon \in L$,

Rekursive Regel : wenn $u \in L$, dann $0u \in L$ und $u1 \in L$.

Beschreibe L .

(1) Behauptung: $\{0\}^* \{1\}^* \subseteq L$.

- ▶ Sei $w \in \{0\}^* \{1\}^*$, also $w = 0^n 1^m$ für $n, m \in \mathbb{N}$.
- ▶ Es ist $\epsilon \in L$ und deshalb $0\epsilon, 00\epsilon, \dots, 0^n\epsilon = 0^n \in L$.
- ▶ Dann aber auch $0^n1, 0^n11, \dots, 0^n1^m \in L$ und deshalb ist $w \in L$ ✓.

Formalisiere mit vollständiger Induktion.

(2) Behauptung: $L \subseteq \{0\}^* \{1\}^*$.

Rekursive Definitionen von Sprachen

Die Sprache L ist die kleinste Menge von Worten über dem Alphabet $\Sigma = \{0, 1\}$ mit den Eigenschaften:

Basisregel : $\epsilon \in L$,

Rekursive Regel : wenn $u \in L$, dann $0u \in L$ und $u1 \in L$.

Beschreibe L .

(1) Behauptung: $\{0\}^* \{1\}^* \subseteq L$.

- ▶ Sei $w \in \{0\}^* \{1\}^*$, also $w = 0^n 1^m$ für $n, m \in \mathbb{N}$.
- ▶ Es ist $\epsilon \in L$ und deshalb $0\epsilon, 00\epsilon, \dots, 0^n\epsilon = 0^n \in L$.
- ▶ Dann aber auch $0^n1, 0^n11, \dots, 0^n1^m \in L$ und deshalb ist $w \in L$ ✓.

Formalisiere mit vollständiger Induktion.

(2) Behauptung: $L \subseteq \{0\}^* \{1\}^*$.

- ▶ Dies sieht man leicht per Induktion nach dem Aufbau von L . ✓

Somit ist $L = \{0\}^* \{1\}^*$. (Eine Mengengleichheit $A = B$ zeigt man häufig durch den Nachweis der Teilmengenbeziehungen $A \subseteq B$ und $B \subseteq A$.)

Entscheidungsprobleme und Komplexitätsklassen

Entscheidungsprobleme

Das Entscheidungsproblem für eine Sprache L :

Für ein vorgegebenes Wort w , entscheide ob $w \in L$.

Entscheidungsprobleme

Das Entscheidungsproblem für eine Sprache L :

Für ein vorgegebenes Wort w , entscheide ob $w \in L$.

- Das Entscheidungsproblem für den Duden:

Entscheide, ob ein vorgegebenes Wort im Duden ist.

Das Entscheidungsproblem für eine Sprache L :

Für ein vorgegebenes Wort w , entscheide ob $w \in L$.

- Das Entscheidungsproblem für den Duden:

Entscheide, ob ein vorgegebenes Wort im Duden ist.

Einfach: Bitte nachschauen.

Das Entscheidungsproblem für eine Sprache L :

Für ein vorgegebenes Wort w , entscheide ob $w \in L$.

- Das Entscheidungsproblem für den Duden:

Entscheide, ob ein vorgegebenes Wort im Duden ist.

Einfach: Bitte nachschauen.

- Das Entscheidungsproblem für Deutsch:

Entscheide, ob ein Satz syntaktisch richtig **und** sinnvoll ist.

Das Entscheidungsproblem für eine Sprache L :

Für ein vorgegebenes Wort w , entscheide ob $w \in L$.

- Das Entscheidungsproblem für den Duden:

Entscheide, ob ein vorgegebenes Wort im Duden ist.

Einfach: Bitte nachschauen.

- Das Entscheidungsproblem für Deutsch:

Entscheide, ob ein Satz syntaktisch richtig **und** sinnvoll ist.

Sehr schwierig.

Entscheidungsprobleme

Das Entscheidungsproblem für eine Sprache L :

Für ein vorgegebenes Wort w , entscheide ob $w \in L$.

- Das Entscheidungsproblem für den Duden:

Entscheide, ob ein vorgegebenes Wort im Duden ist.

Einfach: Bitte nachschauen.

- Das Entscheidungsproblem für Deutsch:

Entscheide, ob ein Satz syntaktisch richtig **und** sinnvoll ist.

Sehr schwierig.

- Das Entscheidungsproblem für C++:

Entscheide, ob ein C++ Programm syntaktisch korrekt ist.

Entscheidungsprobleme

Das Entscheidungsproblem für eine Sprache L :

Für ein vorgegebenes Wort w , entscheide ob $w \in L$.

- Das Entscheidungsproblem für den Duden:

Entscheide, ob ein vorgegebenes Wort im Duden ist.

Einfach: Bitte nachschauen.

- Das Entscheidungsproblem für Deutsch:

Entscheide, ob ein Satz syntaktisch richtig **und** sinnvoll ist.

Sehr schwierig.

- Das Entscheidungsproblem für C++:

Entscheide, ob ein C++ Programm syntaktisch korrekt ist.

Compiler können das.

Entscheidungsprobleme

Das Entscheidungsproblem für eine Sprache L :

Für ein vorgegebenes Wort w , entscheide ob $w \in L$.

- Das Entscheidungsproblem für den Duden:

Entscheide, ob ein vorgegebenes Wort im Duden ist.

Einfach: Bitte nachschauen.

- Das Entscheidungsproblem für Deutsch:

Entscheide, ob ein Satz syntaktisch richtig **und** sinnvoll ist.

Sehr schwierig.

- Das Entscheidungsproblem für C++:

Entscheide, ob ein C++ Programm syntaktisch korrekt ist.

Compiler können das.

Wie schwer ist das Entscheidungsproblem für stets stets haltende C++ Programme?

Hält ein C++ Programme immer?

```
main(int n)
int i;
{
  i=n;
  while (NOT(prim(i) && prim(i+2)) )
    i++; }
```

- Die Funktion `prim(i)` gebe den Wert `wahr` genau dann zurück, wenn i eine Primzahl ist.

Hält ein C++ Programme immer?

```
main(int n)
int i;
{   i=n;
    while (NOT (prim(i) && prim(i+2)) )
        i++; }
```

- Die Funktion `prim(i)` gebe den Wert `wahr` genau dann zurück, wenn i eine Primzahl ist.
- Das Programm hält genau dann für eine Eingabe n , wenn es Primzahl-Zwillinge $i, i + 2$ mit $i \geq n$ gibt.
 - ▶ Das Programm hält genau dann immer, wenn es unendlich viele Primzahl-Zwillinge gibt.
 - ▶ Die Frage, ob es unendlich viele Primzahl-Zwillinge gibt, ist bis heute offen!

Hält ein C++ Programme immer?

```
main(int n)
int i;
{   i=n;
    while (NOT (prim(i) && prim(i+2)) )
        i++; }
```

- Die Funktion `prim(i)` gebe den Wert `wahr` genau dann zurück, wenn i eine Primzahl ist.
- Das Programm hält genau dann für eine Eingabe n , wenn es Primzahl-Zwillinge $i, i + 2$ mit $i \geq n$ gibt.
 - ▶ Das Programm hält genau dann immer, wenn es unendlich viele Primzahl-Zwillinge gibt.
 - ▶ Die Frage, ob es unendlich viele Primzahl-Zwillinge gibt, ist bis heute offen!

Das Entscheidungsproblem, ob ein C++ Programme stets hält, ist

unentscheidbar!

Entscheidungsprobleme: Ein vorläufiges Fazit

Die Komplexität des Entscheidungsproblems variiert stark:

- von **trivialen** Problemen wie dem Dudenproblem, zu **erst zunehmenden** Problemen wie dem Compiler-Problem für C++ Programme,
- zu den **schwierigen** NP-vollständigen Problemen,
- zu den **noch schwierigeren** PSPACE-vollständigen Problemen bis hin zu **unentscheidbaren** Problemen wie dem Entscheidungsproblem für stets haltende C++ Programme.

Entscheidungsprobleme: Ein vorläufiges Fazit

Die Komplexität des Entscheidungsproblems variiert stark:

- von **trivialen** Problemen wie dem Dudenproblem, zu **erst zunehmenden** Problemen wie dem Compiler-Problem für C++ Programme,
- zu den **schwierigen** NP-vollständigen Problemen,
- zu den **noch schwierigeren** PSPACE-vollständigen Problemen bis hin zu **unentscheidbaren** Problemen wie dem Entscheidungsproblem für stets haltende C++ Programme.

Wir betrachten später Entscheidungsprobleme für

- reguläre Sprachen (das Wortproblem oder Äquivalenzprobleme),
- kontextfreie Sprachen (das Wortproblem oder Äquivalenzprobleme),
- Existenz von Gewinnzügen in 2-Personen Spielen,
- Für die Aussagenlogik bzw. die Prädikatenlogik:
 - ▶ Ist eine Formel aus einem Axiomensystem ableitbar?
 - ▶ Ist eine Formel in einer Struktur wahr?

- **Komplexitätsklassen** charakterisieren die Schwierigkeit des Entscheidungsproblems.
 - ▶ P ist die Klasse aller effizient berechenbaren Probleme.
 - ▶ Ein NP-vollständiges Probleme ist wahrscheinlich nicht effizient lösbar.
- Wir führen **weitere fundamentale Komplexitätsklassen** ein, um mehr zu verstehen über
 - ▶ reguläre, kontextfreie und kontextsensitive Sprachen: NC und Speicherklassen,
 - ▶ die Komplexität der Berechnung von Gewinnstrategien, Probabilismus, Quantenberechnungen: PSPACE
 - ▶ Probleme mit super-schnellen parallelen Algorithmen: NC.