Kapitel L:II

II. Aussagenlogik

- Syntax der Aussagenlogik
- Semantik der Aussagenlogik
- Eigenschaften des Folgerungsbegriffs
- Äquivalenz
- Formeltransformation
- Normalformen

- Bedeutung der Folgerung
- Erfüllbarkeitsalgorithmen
- Semantische Bäume
- Weiterentwicklung semantischer Bäume
- Syntaktische Schlussfolgerungsverfahren
- Erfüllbarkeitsprobleme

Erfüllbarkeitsprobleme

Wiederholung (theoretische Informatik)

Die Frage „Gilt $\alpha \models \beta$?“ lässt sich auf einen Erfüllbarkeitstest zurückführen.

Definition 43 (Komplexitätsklasse P)

Die Komplexitätsklasse P enthält alle Probleme, die sich mit einer deterministischen Turingmaschine in polynomieller Rechenzeit lösen lassen.

Erfüllbarkeitsprobleme

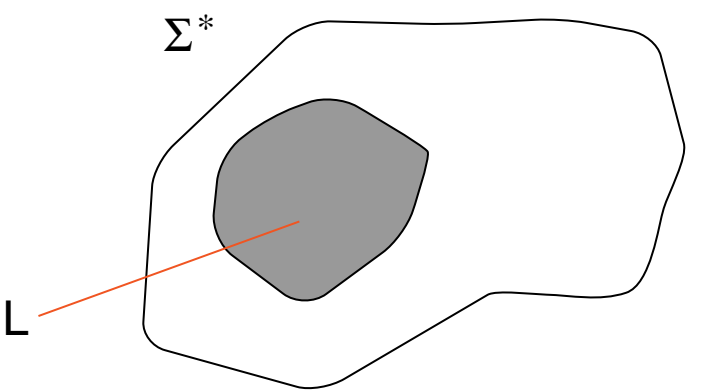
Wiederholung (theoretische Informatik)

Definition 44 (Entscheidungsprobleme)

Gegeben sei ein Problem mit der Eingabemenge Σ^*. Dann bezeichnen wir ein Problem als Entscheidungsproblem, wenn für alle $x \in \Sigma^*$ die Antwort (das Ergebnis, die Ausgabe einer Turingmaschine) nur „0“ (nein) oder „1“ (ja) sein kann.

Definition 45 (Sprache)

Gegeben sei ein Entscheidungsproblem mit der Eingabemenge Σ^*. Dann bezeichnen wir diejenige Teilmenge L von Σ^*, deren Elemente die Antwort „1“ haben, als Sprache.



Erfüllbarkeitsprobleme

Wiederholung (theoretische Informatik)

Definition 46 (Komplexitätsklasse NP)

Die Komplexitätsklasse NP enthält alle Entscheidungsprobleme, von denen mit einer nichtdeterministischen Turingmaschine M in polynomieller Rechenzeit festgestellt werden kann, dass ein Element zur Sprache des Problems gehört.

Sprachgebrauch: M akzeptiert die Elemente der Sprache (eines Entscheidungsproblems aus NP) in polynomieller Zeit.

Bemerkungen:

- Probleme aus der Komplexitätsklasse NP sind entscheidbar.
- Vermutung, dass P $\neq$ NP

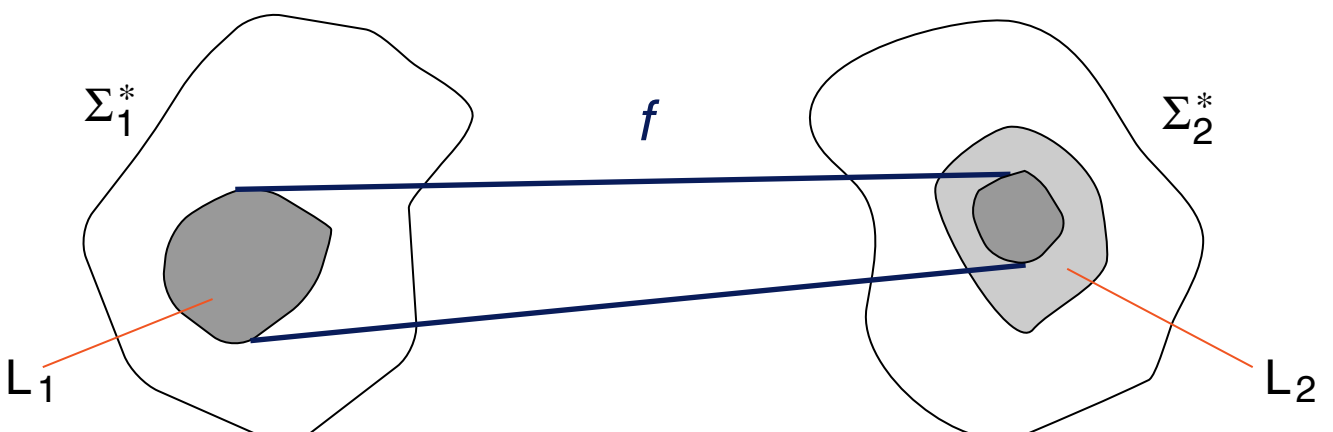
Erfüllbarkeitsprobleme

Wiederholung (theoretische Informatik)

Definition 47 (polynomielle Reduktion)

Eine Sprache $L_1 \subseteq \Sigma_1^*$ lässt sich polynomiell auf eine Sprache $L_2 \subseteq \Sigma_2^*$ reduzieren, in Zeichen: $L_1 \leq L_2$, wenn es eine polynomiell berechenbare Transformation $f : \Sigma_1^* \to \Sigma_2^*$ gibt, so dass gilt:

$$\forall x \in \Sigma_1^* : \quad x \in L_1 \Leftrightarrow f(x) \in L_2$$



Bemerkungen:

- $L_1 \leq L_2$ kann interpretiert werden als: L_1 ist nicht schwerer als L_2.

Erfüllbarkeitsprobleme

Wiederholung (theoretische Informatik)

Definition 48 (hart bzgl. einer Menge von Sprachen)

Eine Sprache L heißt hart für eine Menge von Sprachen $\mathcal{L}$, falls sich jede Sprache $L' \in \mathcal{L}$ auf L reduzieren lässt. In Zeichen: $\forall L' \in \mathcal{L} : L' \leq L$.

Definition 49 (vollständig bzgl. einer Menge von Sprachen)

Eine Sprache L heißt vollständig für eine Menge von Sprachen $\mathcal{L}$, falls L hart für $\mathcal{L}$ ist, und falls zusätzlich $L \in \mathcal{L}$ gilt.

Erfüllbarkeitsprobleme

Definition 50 (SAT*)

SAT* = $\{\alpha \mid \alpha$ aussagenlogische Formel $\wedge\ \alpha$ erfüllbar $\}$

Satz 51 (Komplexität von SAT*)

1. SAT* $\in$ NP
2. SAT* ist NP-hart. [Cook 1971]

Beweis (Skizze)

Zu (1): Elemente aus SAT* werden von einer nichtdeterministischen Turingmaschine in polynomieller Zeit akzeptiert.

Frage: Wie zeigt man das?

Zu (2): Alle Probleme aus NP lassen sich in polynomieller Zeit auf SAT* reduzieren.

Frage: Wie hat Cook das gezeigt?

Bemerkung: SAT ist NP-vollständig.

Erfüllbarkeitsprobleme

Definition 52 (SAT)

SAT = $\{\alpha \mid \alpha \in \text{KNF} \wedge \alpha \text{ erfüllbar } \}$

Satz 53 (Komplexität von SAT)

SAT ist NP-vollständig.

Beweis (Skizze)

Reduktion von SAT* auf SAT, in Zeichen: SAT$^* \leq$ SAT.

Erfüllbarkeitsprobleme

Definition 54 (3SAT)

3SAT = $\{\alpha \mid \alpha \in$ 3KNF $\wedge\ \alpha$ erfüllbar $\}$

Satz 55 (Komplexität von 3SAT)

3SAT ist NP-vollständig.

Beweis (Skizze)

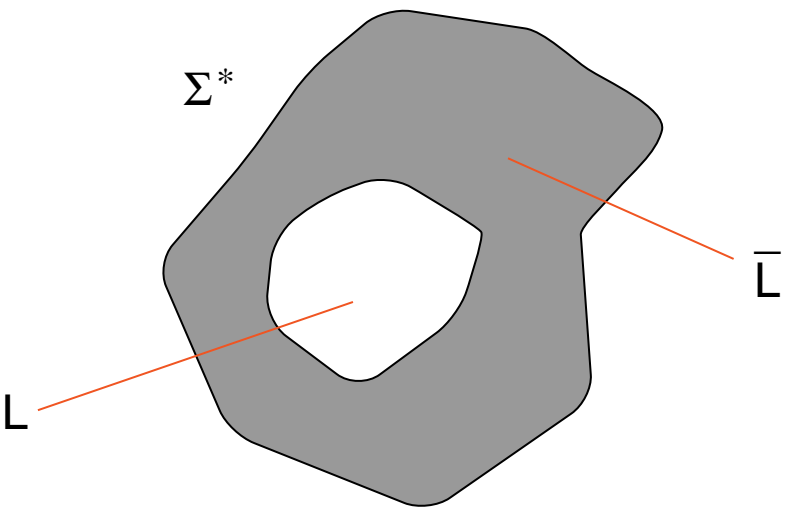
Reduktion von SAT auf 3SAT, in Zeichen: SAT $\leq$ 3SAT.

Erfüllbarkeitsprobleme

Wiederholung (theoretische Informatik)

Definition 56 (Komplexitätsklasse co-NP)

Die Komplexitätsklasse co-NP enthält alle Entscheidungsprobleme, deren Komplementsprachen $\overline{L}$, $\overline{L} := \Sigma^* \setminus L$, in NP liegen.



Vermutung: NP $\neq$ co-NP

Erfüllbarkeitsprobleme

Definition 57 (DEDUCT)

DEDUCT = $\{(\alpha, L) \mid \alpha \in$ KNF $\wedge\ L$ Literal mit $\alpha \models L\}$

Satz 58 (Komplexität von DEDUCT)

DEDUCT ist co-NP-vollständig.

Beweis

Reduktion von SAT auf $\overline{\text{DEDUCT}}$.

- $\overline{\text{DEDUCT}} = \{(\alpha, L) \mid \alpha \in$ KNF $\wedge\ L$ Literal mit $\alpha \not\models L\}$
- Sei A ein Atom mit $A \notin \mathit{atoms}(\alpha)$.
- $\underbrace{\alpha \in}_{x}$ KNF beliebig. $\underbrace{(\alpha, A)}_{f(x)}$ eine spezielle Instanz des Entscheidungsproblems.
- Es gilt: α erfüllbar $\Leftrightarrow$ $\alpha \not\models A$
  Bzw.: $x \in$ SAT $\Leftrightarrow$ $f(x) \in \overline{\text{DEDUCT}}$
- Die Komplementsprache von DEDUCT ist $\overline{\text{DEDUCT}}$ und ist NP-vollständig. Also ist DEDUCT co-NP-vollständig.

Erfüllbarkeitsprobleme

Definition 59 (EQUIV)

EQUIV = $\{(\alpha, \beta) \mid \alpha, \beta \in \text{KNF} \wedge \alpha \approx \beta\}$

Satz 60 (Komplexität von EQUIV)

EQUIV ist co-NP-vollständig.

Beweis

Reduktion von $\overline{\text{SAT}}$ auf EQUIV.

Rest als Übungsaufgabe.

Erfüllbarkeitsprobleme

Definition 61 (2SAT)

2SAT = $\{\alpha \mid \alpha \in$ 2KNF $\wedge\ \alpha$ erfüllbar $\}$

Satz 62 (Komplexität von 2SAT)

2SAT $\in$ P. [Aspvall 1980]

Erfüllbarkeitsprobleme

Beweis (Skizze: Komplexität von 2SAT)

1. Units L durch $L \vee L$ ersetzen $\Rightarrow$ alle Klauseln haben genau zwei Literale.
2. Generierung eines gerichteten Graphen $G = \langle V, E \rangle$. V enthält alle Literale aus α sowie deren Komplemente.
3. Aus jeder Klausel (L_1, L_2) werden zwei Kanten. Es gilt:
   $(L_1, L_2) \approx (L_1 \vee L_2) \wedge (L_2 \vee L_1) \approx (\neg\neg L_1 \vee L_2) \wedge (\neg\neg L_2 \vee L_1) \approx (\neg L_1 \rightarrow L_2) \wedge (\neg L_2 \rightarrow L_1)$
4. Die starken Zusammenhangskomponenten von G sind zyklische Ketten von Implikationen. Sie können nur dann erfüllt sein, wenn alle beteiligten Literale entweder mit 0 oder mit 1 bewertet sind. Folglich dürfen alle starken Zusammenhangskomponenten zu einem Knoten kontrahiert werden.
5. Erzeugung einer Initialbewertung: Bewertung aller Knoten, die nur ausgehende Kanten haben, mit 0. Bewertung aller Knoten, die nur eingehende Kanten haben, mit 1. (Least-Commitment-Prinzip)
6. Propagierung der Initialbewertung entlang einer topologischen Sortierung.
7. α ist erfüllbar $\Leftrightarrow$ keine starke Zusammenhangskomponente enthält ein Literal als positive und negative Instanz.

Erfüllbarkeitsprobleme

Definition 63 (SAT-Probleme in HORN)

1. SAT∩HORN = $\{\alpha \mid \alpha \in$ HORN $\wedge\ \alpha$ erfüllbar $\}$
2. SAT∩DHORN = $\{\alpha \mid \alpha \in$ DHORN $\wedge\ \alpha$ erfüllbar $\}$
3. DEDUCT∩HORN = $\{(\alpha, L) \mid \alpha \in$ HORN $\wedge\ L$ Literal mit $\alpha \models L\}$
4. EQUIV∩HORN = $\{(\alpha, \beta) \mid \alpha, \beta \in$ HORN $\wedge\ \alpha \approx \beta\}$

Satz 64 (Komplexität von SAT-Problemen in HORN)

Die Probleme SAT∩HORN, SAT∩DHORN, DEDUCT∩HORN und EQUIV∩HORN sind in P.

Erfüllbarkeitsprobleme