

FORMALE SYSTEME

ÜBUNG 13

Eric Kunze

`eric.kunze@tu-dresden.de`

TU Dresden, 28. Januar 2022

letzte Änderung:
01.02.2022, 11:40

Aufgabe 1

Resolution

Aufgabe 2

k-Färbbarkeit

Aufgabe 3

Aussagenlogik nur mit \rightarrow ?

Aufgabe 4

Polynomielle Abschlüsse

Aufgabe 5

Komplexität co-endlicher Sprachen

Aufgabe 1

Resolution

KLAUSELFORM & RESOLUTION

- ▶ Eine Klausel $L_1 \vee \dots \vee L_n$ wird dargestellt als Menge $\{L_1, \dots, L_n\}$
- ▶ Eine Konjunktion von Klauseln $K_1 \wedge \dots \wedge K_\ell$ wird dargestellt als Menge $\{K_1, \dots, K_\ell\}$

Eine Menge von Mengen von Literalen unter dieser Interpretation heißt **Klauselform**.

Gegeben seien zwei Klauseln K_1 und K_2 für die es ein Atom $p \in \mathbf{P}$ gibt mit $p \in K_1$ und $\neg p \in K_2$.

Die **Resolvente von K_1 und K_2 bezüglich p** ist die Klausel

$$(K_1 \setminus \{p\}) \cup (K_2 \setminus \{\neg p\})$$

Eine Klausel R ist eine **Resolvente einer Klauselmengen \mathcal{K}** wenn R Resolvente zweier Klauseln $K_1, K_2 \in \mathcal{K}$ ist.

BEDEUTUNG VON RESOLUTIONSSCHRITTEN

Wir betrachten Klauseln $\{L_1, \dots, L_n, p\}$ und $\{\neg p, M_1, \dots, M_\ell\}$:

- ▶ $\{L_1, \dots, L_n, p\} \equiv (L_1 \vee \dots \vee L_n \vee p) \equiv (\neg L_1 \wedge \dots \wedge \neg L_n) \rightarrow p$
- ▶ $\{\neg p, M_1, \dots, M_\ell\} \equiv (\neg p \vee M_1 \vee \dots \vee M_\ell) \equiv p \rightarrow (M_1 \vee \dots \vee M_\ell)$

Daraus folgt unmittelbar $(\neg L_1 \wedge \dots \wedge \neg L_n) \rightarrow (M_1 \vee \dots \vee M_\ell)$

\rightsquigarrow dies entspricht der Klausel $\{L_1, \dots, L_n, M_1, \dots, M_\ell\}$

Satz: Wenn R Resolvente der Klauseln K_1 und K_2 ist, dann gilt $\{K_1, K_2\} \models R$.

- ▶ Die leere Klausel ist eine Disjunktion von 0 Literalen, also gerade \perp (neutrales Element von \vee)
- ▶ \perp in einer Konjunktion macht die gesamte Formel falsch (unerfüllbar).
- ▶ Lässt sich \perp ableiten, so ist die Formel unerfüllbar.

DAS RESOLUTIONSKALKÜL

Resolution

Gegeben: Formel \mathcal{F} in Klauselform

Gesucht: Ist \mathcal{F} erfüllbar oder unerfüllbar?

- (1) Finde ein Klauselpaar $K_1, K_2 \in \mathcal{F}$ mit Resolvente $R \notin \mathcal{F}$
- (2) Setze $\mathcal{F} := \mathcal{F} \cup \{R\}$
- (3) Wiederhole Schritt (1) und (2) bis keine neuen Resolventen gefunden werden können
- (4) Falls $\perp \in \mathcal{F}$, dann gib „unerfüllbar“ aus;
andernfalls gib „erfüllbar“ aus

Beobachtung: Unerfüllbarkeit steht fest, sobald \perp abgeleitet wurde
 \rightsquigarrow dann kann man das Verfahren frühzeitig abbrechen

Erfüllbarkeit kann dagegen erst erkannt werden, wenn alle Resolventen erschöpfend gebildet worden sind

AUFGABE 1

Prüfen Sie mittels Resolution die folgenden Formeln jeweils auf Erfüllbarkeit:

a) $a \wedge \left((c \wedge b) \wedge ((\neg c \vee \neg b) \vee (a \wedge (c \wedge b))) \right)$

b) $(\neg a \vee \neg b) \wedge (a \vee b) \wedge (\neg a \vee c) \wedge (a \vee \neg c) \\ \wedge (b \vee \neg d) \wedge (\neg b \vee d) \wedge (\neg c \vee d) \wedge (c \vee \neg d)$

Aufgabe 2

k -Färbbarkeit

AUFGABE 2

Eine k -Färbung für einen endlichen Graphen G ist eine Zuordnung der Knoten von G zu Werten („Farben“) in $\{1, \dots, k\}$, so dass Knoten, die in G durch eine Kante verbunden sind, nicht denselben Wert zugeordnet bekommen.

Geben Sie für einen endlichen Graphen $G = (V, E)$ mit n Knoten und einen Wert k eine aussagenlogische Formel $\varphi_{G,k}$ an, so dass $\varphi_{G,k}$ genau dann erfüllbar ist, wenn es eine k -Färbung von G gibt.

Wir bezeichnen die Menge aller Farben mit $C_k = \{1, \dots, k\}$. Wir verwenden die aussagenlogischen Variablen $p_{v,c}$ für $v \in V$ und $c \in C_k$ um auszudrücken, dass der Knoten v mit der Farbe c gefärbt wird.

- Jeder Knoten hat mindestens eine Farbe:

$$\varphi_{\geq 1} := \bigwedge_{v \in V} \bigvee_{c \in C_k} p_{v,c}$$

- Jeder Knoten hat höchstens eine Farbe:

$$\varphi_{\leq 1} := \bigwedge_{v \in V} \bigwedge_{\substack{c_1, c_2 \in C_k \\ c_1 \neq c_2}} \neg (p_{v,c_1} \wedge p_{v,c_2})$$

- Benachbarte Knoten haben unterschiedliche Farben

$$\varphi_{\neq} := \bigwedge_{\substack{v_1, v_2 \in V \\ (v_1, v_2) \in E}} \bigwedge_{c \in C_k} \neg (p_{v_1,c} \wedge p_{v_2,c})$$

Die Formel $\varphi_{G,k}$ ergibt sich dann als

$$\varphi_{G,k} := \varphi_{\geq 1} \wedge \varphi_{\leq 1} \wedge \varphi_{\neq}.$$

Aufgabe 3

Aussagenlogik nur mit \rightarrow ?

AUFGABE 3

- a) Es sei φ eine Formel, die ausschließlich den Junktor \rightarrow verwendet. Zeigen Sie: Wenn $w(p_i) = 1$ für alle $p_i \in \text{Var}(\varphi)$ ist, dann ist auch $w(\varphi) = 1$.
- b) Es sei φ eine allgemeine aussagenlogische Formel. Beweisen oder widerlegen Sie: φ ist äquivalent zu einer Formel, die ausschließlich den Junktor \rightarrow verwendet.

(a) Induktion über den Aufbau von Formeln.

(IA) Sei $\varphi = p \in \mathbf{P}$ ein Atom. Dann gilt $w(p) = 1 \implies w(\varphi) = 1$.

(IV) Seien φ_1, φ_2 Formeln, die nur den Junktor \rightarrow verwenden und Folgendes erfüllen: Gilt $w(p_1) = 1$ für alle $p_1 \in \text{Var}(\varphi_1)$, dann gilt $w(\varphi_1) = 1$; sowie gilt $w(p_2) = 1$ für alle $p_2 \in \text{Var}(\varphi_2)$, dann gilt $w(\varphi_2) = 1$.

(IS) Sei $\varphi := (\varphi_1 \rightarrow \varphi_2)$. Es gilt $\text{Var}(\varphi) = \text{Var}(\varphi_1) \cup \text{Var}(\varphi_2)$. Ist $w(p) = 1$ für alle $p \in \text{Var}(\varphi)$, so auch $w(p) = 1$ für alle $p \in \text{Var}(\varphi_1)$ und $w(p) = 1$ für alle $p \in \text{Var}(\varphi_2)$. Per Induktionsvoraussetzung gilt dann auch $w(\varphi_1) = 1$ und $w(\varphi_2) = 1$; und somit schließlich $w(\varphi) = w(\varphi_1 \rightarrow \varphi_2) = 1$.

(b) Die Aussage ist falsch. Sei $p \in \mathbf{P}$ ein Atom mit $w(p) = 1$. Dann gibt es keine zu $\neg p$ äquivalente Formel mit nur dem Junktor \rightarrow .

Achtung: auch die „Abkürzungen“ \top und \perp dürfen nicht verwendet werden, d.h. $p \rightarrow \perp$ ist keine Formel, die nur den Junktor \rightarrow verwendet.

Aufgabe 4

Polynomielle Abschlüsse

AUFGABE 4

Seien Σ ein Alphabet und $L, L_1, L_2 \subseteq \Sigma^*$ Sprachen mit $L, L_1, L_2 \in \mathbf{P}$.

Zeigen Sie:

- a) $L_1 \cup L_2, L_1 \circ L_2$ und \bar{L} sind in polynomieller Zeit entscheidbar.
- b) $L^R = \{w^R : w \in L\}$ ist in polynomieller Zeit entscheidbar, wobei für $w = a_1 \dots a_n \in \Sigma^*$ das Rückwärtswort w^R definiert ist durch

$$w^R = a_n a_{n-1} \dots a_1$$

(und $\varepsilon^R = \varepsilon, a^R = a$ für $a \in \Sigma$).

(a) Seien \mathcal{M} , \mathcal{M}_1 und \mathcal{M}_2 polynomiell zeitbeschränkte Turingmaschinen.

\bar{L} : Die folgende Turingmaschine $\bar{\mathcal{M}}$ entscheidet \bar{L} :

Eingabe w

Simuliere \mathcal{M} auf w .

falls \mathcal{M} akzeptiert: verwerfe

falls \mathcal{M} verwirft: akzeptiere

Da \mathcal{M} in polynomieller Zeit entscheidet, entscheidet auch $\bar{\mathcal{M}}$ in polynomieller Zeit. Es gilt also $\bar{L} \in \mathbf{P}$.

$L_1 \cup L_2$: Die folgende Turingmaschine \mathcal{M}_U entscheidet $L_1 \cup L_2$:

Eingabe w

Simuliere \mathcal{M}_1 auf w .

falls \mathcal{M}_1 akzeptiert: akzeptiere

falls \mathcal{M}_1 verwirft:

Simuliere \mathcal{M}_2 auf w .

falls \mathcal{M}_2 akzeptiert: akzeptiere

falls \mathcal{M}_2 verwirft: verwerfe

Sowohl \mathcal{M}_1 als auch \mathcal{M}_2 entscheiden ihren Teil in polynomieller Zeit und „polynomiell + polynomiell = polynomiell“, d.h. auch \mathcal{M}_U entscheidet in polynomieller Zeit. Somit gilt $L_1 \cup L_2 \in \mathbf{P}$.

(a) Seien weiterhin \mathcal{M} , \mathcal{M}_1 und \mathcal{M}_2 polynomiell zeitbeschränkte Turingmaschinen.

$L_1 \circ L_2$: Die folgende Turingmaschine \mathcal{M}_o entscheidet $L_1 \circ L_2$:

Eingabe $w = a_1 \dots a_n$

Für alle $1 \leq k \leq n$:

Simuliere \mathcal{M}_1 auf $a_1 \dots a_k$.

falls \mathcal{M}_1 verwirft: wähle nächstes k

falls \mathcal{M}_1 akzeptiert:

Simuliere \mathcal{M}_2 auf $a_{k+1} \dots a_n$.

falls \mathcal{M}_2 akzeptiert: akzeptiere

falls \mathcal{M}_2 verwirft: wähle nächstes k

Sowohl \mathcal{M}_1 als auch \mathcal{M}_2 entscheiden ihren Teil in polynomieller Zeit. Die äußere Schleife wird maximal n mal durchlaufen, d.h. der Aufwand lässt sich salopp mit

$$n \cdot (\text{polynomiell} + \text{polynomiell}) = \text{polynomiell}$$

angeben. Daher ist $L_1 \circ L_2 \in \mathbf{P}$.

(b) *Eingabe: $w \in \Sigma^*$.*

Transformiere w zu w^R .

Simuliere \mathcal{M} auf w^R .

falls \mathcal{M} akzeptiert: akzeptiere

falls \mathcal{M} verwirft: verwerfe

Die Simulation von \mathcal{M} ist polynomiell nach Voraussetzung und die Transformation von w zu w^R ist offensichtlich auch polynomiell.

Somit ist $L^R \in \mathbf{P}$.

Aufgabe 5

Komplexität co-endlicher Sprachen

AUFGABE 5

Eine Sprache $L \subseteq \Sigma^*$ ist *co-endlich* genau dann, wenn \bar{L} (also $\Sigma^* \setminus L$) endlich ist.

Beweisen oder widerlegen Sie:

- a) Jede co-endliche Sprache ist in $\text{DTIME}(n)$.
- b) Jede co-endliche Sprache ist in $\text{DTIME}(1)$.

- (a) Die Aussage gilt. Jede endliche Sprache ist regulär und reguläre Sprachen sind unter Komplement abgeschlossen. Daher ist jede co-endliche Sprache regulär.

Da das Wortproblem für reguläre Sprachen in linearer Zeit entscheidbar ist, ist jede co-endliche Sprache in $\text{DTIME}(n)$.

- (b) Auch diese Aussage gilt. Sei $L \subseteq \Sigma^*$ co-endlich. Dann ist per Definition \bar{L} endlich.

- ▷ Ist $\bar{L} = \emptyset$, so ist $L = \Sigma^*$ und damit offensichtlich in konstanter Zeit entscheidbar (akzeptiere alle Eingaben).
- ▷ Ist $\bar{L} \neq \emptyset$, so existiert (da \bar{L} endlich ist) ein längstes Wort w_{\max} mit $|w_{\max}| = m$. Somit muss jede Turingmaschine \mathcal{M} höchstens m Zeichen der Eingabe lesen (unabhängig von der Eingabe) und kann dann sofort $w \in \bar{L}$ und damit $w \in L$ entscheiden. Damit gilt nun: die Anzahl der Schritte von \mathcal{M} bei Eingabe $w \in \Sigma^*$ sind durch $f : \mathbb{N} \rightarrow \mathbb{R}$ mit $f(n) = c_m$ charakterisiert, wobei $c_m \in \mathbb{R}$ eine positive Konstante ist, die nur von m abhängt. Für dieses f gilt $f \in \mathcal{O}(1)$, da $f(n) \leq c_m \cdot 1$ für alle $n \geq 0$ gilt. Damit folgt $L \in \text{DTIME}(1)$.