Karlsruher Institut für Technologie
Institut für Theoretische Informatik
Prof. Dr. Bernhard Beckert
Thorsten Bormer, Dr. Vladimir Klebanov, Dr. Mattias Ulbrich

Formale Systeme, WS 2014/2015

Übungsblatt 3

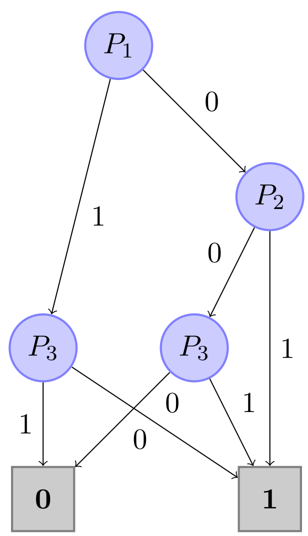
Dieses Übungsblatt wird in der Übung am 14.11.2014 besprochen.

Aufgabe 1

Geben Sie zu folgendem Shannongraphen je eine äquivalente aussagenlogische Formel in

(a) disjunktiver Normalform und

(b) konjunktiver Normalform an.



Lösung zu Aufgabe 1

(a) Die natürlichste DNF ist die Disjunktion aller Pfade zur **1**. Die genommenen Kanten eines Pfades werden dabei konjunktiv verknüpft: Bei Kanten-Markierung **1** mit dem Literal des verlassenen Knotens, bei Kanten-Markierung **0** mit der Negation des entsprechenden Literals.

$$(P_1 \land \neg P_3) \lor (\neg P_1 \land \neg P_2 \land P_3) \lor (\neg P_1 \land P_2)$$

(b) Wie so oft ist das das Duale: Die Konjunktion aller Pfade zur 0. Die genommenen Kanten eines Pfades werden dabei disjunktiv verknüpft: Bei Kanten-Markierung $\boxed{\mathbf{0}(!)}$ mit dem Literal des verlassenen Knotens, bei Kanten-Markierung $\boxed{\mathbf{1}(!)}$ mit der Negation des entsprechenden Literals.

$$(\neg P_1 \lor \neg P_3) \land (P_1 \lor P_2 \lor P_3)$$

Idee: Während man bei (a) alle möglichen Wege zur **1** aufzählt, stellt man in (b) Bedingungen auf, damit die **0** vermieden wird.

Aufgabe 2

Zeigen Sie mit Hilfe des David-Putnam-Verfahrens, dass die Klauselmenge

$$\{ \quad \{\neg B, C\}, \quad \{\neg A, B, C\}, \quad \{\neg A, B, \neg C\}, \quad \{\neg B, \neg C\}, \quad \{A, B, C\}, \quad \{A, B, \neg C\} \quad \}$$

unerfüllbar ist.

Lösung zu Aufgabe 2

Die Ausgangsmenge S hat keine Einerklausel, daher muss ein Literal L gewählt werden, über das die Fallunterscheidung stattfinden soll. Geschickterweise wählen wir B, damit bei diesem Schritt dann Einerklauseln entstehen.

- Fall S_B: Im zweiten Schritt wird hier die Einerklausel $\{C\}$ gewählt und wahr gemacht:

S_B		
$\{B\}$,	–	–
$\{\neg B, C\}$,	$\{C\}$	–
$\{\neg A, B, C\}$,	–	–
$\{\neg A, B, \neg C\}$,	–	–
$\{\neg B, \neg C\}$,	$\{\neg C\}$	$\square$
$\{A, B, C\}$,	–	–
$\{A, B, \neg C\}$	–	–

Es entsteht die leere Klausel, also ist die Menge S_B unerfüllbar.

- Fall $S_{\neg B}$: Hier gibt es für den zweiten Schritt keine Einerklauseln zum Wählen, so dass erneut eine Verzweigung (z.B. nach A) stattfinden muss. Danach entstehen dann komplementäre Einerklauseln, die wie oben aufgelöst werden.

$S_{\neg B}$		$S_{\neg B, A}$		$S_{\neg B, \neg A}$	
		$\{A\}$	–	$\{\neg A\}$	–
$\{\neg B\}$,	–	–	–	–	–
$\{\neg B, C\}$,	–	–	–	–	–
$\{\neg A, B, C\}$,	$\{\neg A, C\}$	$\{\neg A, C\}$	$\{C\}$	$\{\neg A, C\}$	–
$\{\neg A, B, \neg C\}$,	$\{\neg A, \neg C\}$	$\{\neg A, \neg C\}$	$\{\neg C\}$	$\{\neg A, \neg C\}$	–
$\{\neg B, \neg C\}$,	–	–	–	–	–
$\{A, B, C\}$,	$\{A, C\}$	$\{A, C\}$	–	$\{A, C\}$	$\{C\}$
$\{A, B, \neg C\}$	$\{A, \neg C\}$	$\{A, \neg C\}$	–	$\{A, \neg C\}$	$\{\neg C\}$

Aufgabe 3

Die Linux-Distribution openSUSE verwendet für die Paketverwaltung einen SAT-Solver. Das Verfahren ist unter `https://en.opensuse.org/openSUSE:Libzypp_satsolver_basics` beschrieben.

Formalisieren Sie das folgende Szenario mittels Aussagenlogik, benutzen Sie die kursiv gedruckten Begriffe als Variablennamen:

1. Der Benutzer möchte den Mail-Client *mutt* installieren.
2. Der Mail-Client erfordert einen *smtp daemon*.
3. Ein gültiger *smtp daemon* ist entweder *sendmail*, *postfix* oder *exim* (es kann nur einer gleichzeitig installiert werden).
4. *sendmail* macht das installierte Legacy-Paket *sendmail-tls* obsolet, das aber nicht deinstalliert werden darf.

Kann die Paketverwaltung die Wünsche des Benutzers erfüllen?

Lösung zu Aufgabe 3

Die Formalisierung ist die Konjunktion folgender Teile:

Der Benutzer möchte den Mail-Client *mutt* installieren.	$mutt$
Der Mail-Client erfordert einen *smtp daemon*.	$mutt \rightarrow smtp\_daemon$
Ein gültiger *smtp daemon* ist entweder *sendmail*, *postfix* oder *exim*,	$smtp\_daemon \rightarrow$ $(sendmail \vee postfix \vee exim)$
es kann nur einer gleichzeitig installiert werden.	$\neg(sendmail \wedge postfix)$ $\neg(sendmail \wedge exim)$ $\neg(postfix \wedge exim)$
sendmail macht das installierte Legacy-Paket *sendmail-tls* obsolet,	$\neg(sendmail \wedge sendmail\text{-}tls)$
das aber nicht deinstalliert werden darf.	$sendmail\text{-}tls$

Die Formel hat zwei verschiedene Modelle:

$$I(mutt) = W, I(smtp\_daemon) = W, I(sendmail) = F, I(postfix) = W, I(exim) = F, I(sendmail\text{-}tls) = W$$

oder

$$I(mutt) = W, I(smtp\_daemon) = W, I(sendmail) = F, I(postfix) = F, I(exim) = W, I(sendmail\text{-}tls) = W$$

Die als wahr interpretierten Atome geben genau die Pakete an, die installiert sind oder werden müssen, um den Benutzerwunsch zu erfüllen.

Im allgemeinen wählt der in openSuSE implementierter SAT-Solver das Modell, das sich am wenigsten von dem Modell unterscheidet, das den Anfangszustand des Systems beschreibt.

Aufgabe 4

Handelt es sich bei den folgenden Zeichenketten um Terme oder Formeln der Prädikatenlogik erster Stufe? Welche Vorkommen welcher Variablen sind frei, welche gebunden?

(a) $j(f(x), g(x), h(z), k)$

(b) $\forall y \exists p\, p(y)$

(c) $\forall x \forall z\, (g(f(z), f(y)) \to z)$

(d) $\forall x \exists y\, (p(f(x)) \to q(y, z))$

Die Signatur enthalte dabei folgende Symbole: $F_\Sigma = \{f, g, h, i, j, k\}$, $P_\Sigma = \{p, q, r\}$. Die Stelligkeiten der Symbole können Sie als korrekt verwendet annehmen. Außerdem sei $Var = \{x, y, z\}$.

Lösung zu Aufgabe 4

(a) $j(f(x), g(x), h(z), k)$ ist ein wohlgeformter Term.

(b) $\forall y \exists p\, p(y)$ ist weder Term noch Formel. In PL1 ist es nicht möglich, über Prädikatensymbole zu quantifizieren ($\exists p$).

(c) $\forall x \forall z\, (g(f(z), f(y)) \to z)$ ist weder Term noch Formel. Implikation zwischen Termen ist nicht wohlgeformt.

(d) $\forall x \exists y\, (p(f(\underset{\substack{\uparrow \\ \text{gebunden}}}{x})) \to q(\overset{\substack{\text{gebunden} \\ \downarrow}}{y}, \underset{\substack{\uparrow \\ \text{frei}}}{z}))$ ist eine Formel. Die Variablenvorkommen sind entsprechend markiert.