Bachelorarbeit

Eine Variante des DLS-Algorithmus zum Lösen von Formelgleichungen

ausgeführt am

Institut für
Diskrete Mathematik und Geometrie
TU Wien

unter der Anleitung von

Associate Prof. Dipl.-Ing. Dr.techn. Stefan Hetzl, BSc

durch

Fabian Achammer
Matrikelnummer: 01621489
Reinprechtsdorfer Straße 9/16
1050 Wien

Wien, am 16. März 2020

Danksagung

Hiermit bedanke ich mich bei Professor Stefan Hetzl für die Betreuung und das Zuteilen einer sehr spannenden Bachelorarbeit. Außerdem möchte ich mich bei Jannik Vierling für das konstruktive Feedback und die Hilfestellungen bei der Implementierung des hier präsentierten Algorithmus in GAPT bedanken. Weiters danke ich Marcia Mchemmech und Markus Rinke für das Korrekturlesen der vorliegenden Arbeit. Meiner Familie möchte ich für die ständige Unterstützung in allen Lebenslagen danken.

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Bachelorarbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt bzw. die wörtlich oder sinngemäß entnommenen Stellen als solche kenntlich gemacht habe.

Wien, am 16. März 2020

Fabian Achammer

Inhaltsverzeichnis

1 Einleitung

Sei X eine Relationsvariable und $\varphi[X]$ eine Formel der klassischen Prädikatenlogik erster Stufe mit Gleichheit, die X enthalten kann. Wir nennen Formeln der Form $\exists X\ \varphi[X]$ Formelgleichungen und stellen uns die Frage, ob wir einen Zeugen für diesen Existenzquantor finden können, also eine Formel $W[\overline{x}]$ mit $\models \varphi[W]$?

Ein verwandtes Problem ist die Elimination von Quantoren zweiter Stufe, bei dem eine Formel ψ in der Prädikatenlogik erster Stufe mit $\models \exists X \varphi[X] \leftrightarrow \psi$ gesucht wird. Der DLS-Algorithmus in [GSS08] löst das Problem der Quantorenelimination zweiter Stufe für eine gewisse Klasse von Formeln. Eine Implementierung des DLS-Algorithmus wird in [Gus96] beschrieben. Wir werden eine stärkere Variante des Problems der Elimination von Quantoren zweiter Stufe betrachten: Gesucht ist eine Formel $W[\overline{x}]$ in der Prädikatenlogik erster Stufe mit $\models \exists X \varphi[X] \leftrightarrow \varphi[W]$. In [EHW17] wird gezeigt, dass der DLS-Algorithmus verwendet werden kann, um dieses Problem für quantorenfreie $\varphi[X]$ zu lösen. Die dabei eingesetzte Methode zur Kombination von Zeugen wird in [Wer17] auf nicht-quantorenfreie Formeln verallgemeinert.

Wir präsentieren eine Variante des DLS-Algorithmus, die die Methode in [EHW17] mithilfe von [Wer17] auf eine größere Klasse von Formeln ausweitet. Eine Implementierung des Algorithmus ist in GAPT (General Architecture for Proof Theory) [1] zu finden.

Im nächsten Kapitel führen wir die verwendete Notation ein und geben theoretische Grundlagen zu Formelgleichungen. In Kapitel 3 beschreiben wir den Algorithmus und gehen auf Unterschiede zum ursprünglichen DLS-Algorithmus ein. Das letzte Kapitel beschäftigt sich mit Beschränkungen des Algorithmus und gibt eine Klasse von Formelgleichungen an, für die der vorgelegte Algorithmus einen Zeugen für die Quantorenelimination findet.

[1] https://www.logic.at/gapt/

2 Formelgleichungen

Wir führen in diesem Kapitel die verwendete Notation ein, definieren den Begriff Formelgleichung, setzen das Lösen von Formelgleichungen in Bezug zur Elimination von Quantoren zweiter Stufe und schließen mit zwei wichtigen Resultaten, die als theoretische Grundlagen für den Algorithmus im nächsten Kapitel dienen.

2.1 Notation

Wir arbeiten im Kontext der klassischen Prädikatenlogik zweiter Ordnung mit Gleichheit. Genauer fixieren wir eine Sprache $\mathcal{L}$, die für alle $n \in \mathbb{N}$ abzählbar viele Funktionssymbole $a, b, c, \ldots$ und Relationssymbole $R, Q, S, \ldots$ der Stelligkeit n enthält. Darüber hinaus enthalte $\mathcal{L}$ ein ausgezeichnetes binäres Relationssymbol für Gleichheit, dargestellt durch $\approx$. *Terme* werden wie üblich über Individuenvariablen (dargestellt durch $x, x_1, x_2, \ldots,$ $y, y_1, y_2, \ldots,$ $z, z_1, z_2, \ldots$) und Funktionssymbole definiert. *Atomformeln* werden über Terme, Relationssymbole und Relationsvariablen (durch $X, X_1, X_2, \ldots,$ $Y, Y_1, Y_2, \ldots$ dargestellt) definiert. *Formeln* werden über Atomformeln, die Verknüpfungen $\wedge$, $\vee$ und $\neg$, sowie Quantoren erster Ordnung $\exists x$, $\forall x$ und Quantoren zweiter Ordnung $\exists X$ definiert. Wir werden auch Formeln in Prädikatenlogik erster Ordnung mit Gleichheit betrachten. Die Menge dieser Formeln bezeichnen wir mit P und nennen deren Elemente *P-Formeln*.

Wir verwenden die Schreibweise $\overline{x}$ für ein Tupel von verschiedenen Individuenvariablen, dessen Größe aus dem Kontext hervorgeht. Um verschiedene Tupel zu unterscheiden, schreiben wir $\overline{x}^1, \overline{x}^2, \ldots$. Eine analoge Schreibweise verwenden wir für Tupel von Termen indem wir $\overline{t}$ schreiben. Die einzelnen Elemente adressieren wir dann durch einen Index $\overline{x}_1, \overline{x}_2, \ldots$. Wir verwenden diese Notation auch in Zusammenhang mit Relations- und Funktionssymbolen und schreiben $R(\overline{x})$ für $R(\overline{x}_1, \ldots, \overline{x}_n)$ bzw. $f(\overline{x})$ für $f(\overline{x}_1, \ldots, \overline{x}_n)$. Für Quantoren schreiben wir auch $Q\overline{x}$ für $Q\overline{x}_1 \ldots Q\overline{x}_n$ für $Q \in \{\forall, \exists\}$. Mit $\overline{x} \approx \overline{t}$ drücken wir $\overline{x}_1 \approx \overline{t}_1 \wedge \cdots \wedge \overline{x}_n \approx \overline{t}_n$ aus.

Für die Substitution von Formeln verwenden wir folgende Notation: Sei $\overline{x}$ ein Tupel von n Individuenvariablen. Wenn wir eine Formel durch $\varphi[\overline{x}]$ einführen und ein n-Tupel $\overline{t}$ von Termen gegeben ist, dann bezeichnet $\varphi[\overline{t}]$ die gleichzeitige Substitution aller $\overline{x}_i$ durch $\overline{t}_i$. Falls eine Formel durch $\varphi[X]$ mit einer n-stelligen Relationsvariable X eingeführt wird und eine andere Formel durch $\psi[\overline{x}]$ mit einem n-Tupel $\overline{x}$ von Individuenvariablen, dann bezeichnet $\varphi[\psi]$ die gleichzeitige Substitution jeder Atomformel $X(\overline{t})$ in $\varphi[X]$ mit Termen $\overline{t}$ durch $\psi[\overline{t}]$. Darauf aufbauend, falls eine Formel durch $\varphi[X_1, \ldots, X_n]$ eingeführt wird, und Formeln $\psi_1[\overline{x}^1], \ldots, \psi_n[\overline{x}^n]$ gegeben sind, bezeichnen wir mit $\varphi[\psi_1, \ldots, \psi_n]$ die gleichzeitige Substitution aller Atomformeln $X_i(\overline{t}^j)$ in $\varphi[X_1, \ldots, X_n]$ durch $\psi_i[\overline{t}^j]$.

Eine Formel φ nennen wir *gültig*, wenn sie in der klassischen Prädikatenlogik zweiter Stufe mit Gleichheit gültig ist. Symbolisch schreiben wir dafür $\models \varphi$.

2.2 Formelgleichungen und Quantorenelimination zweiter Stufe

Definition 1 (Formelgleichung, Zeuge)**.** Seien $n \in \mathbb{N}$ und $X_1, \dots, X_n$ Relationsvariablen mit den Stelligkeiten $a_1, \dots, a_n$. Sei weiters $\varphi[X_1, \dots, X_n] \in P$. Eine Formel der Form $\exists X_1 \dots \exists X_n \; \varphi[X_1, \dots, X_n]$ heißt *Formelgleichung in den Unbekannten* $X_1, \dots, X_n$. Wir nennen $\varphi[X_1, \dots, X_n]$ den *P-Teil* der Formelgleichung. Für $i \in \{1, \dots, n\}$ sei $\overline{x}^i$ ein Tupel von a_i verschiedenen Variablen. Ein Tupel $\overline{W} := (W_1[\overline{x}^1], \dots, W_n[\overline{x}^n])$ von P-Formeln heißt *Lösung* der Formelgleichung in den Unbekannten $X_1, \dots, X_n$, falls $\models \varphi[\overline{W}]$. Wir nennen das Tupel $\overline{W}$ in diesem Fall *Zeuge* der Formelgleichung.

Das Lösen von Formelgleichungen in mehreren Unbekannten kann auf das Lösen von Formelgleichungen in einer Unbekannten reduziert werden. Denn sei $\exists X_1 \dots \exists X_n \varphi[X_1, \dots, X_n]$ eine Formelgleichung in den Unbekannten $X_1, \dots, X_n$ und $W_n[X_1, \dots, X_{n-1}]$ ein Zeuge der Formelgleichung $\exists X_n \varphi[X_1, \dots, X_n]$, dann ist

$$\exists X_1 \dots \exists X_{n-1} \varphi[X_1, \dots, X_{n-1}, W_n[X_1, \dots, X_{n-1}]]$$

eine Formelgleichung in den Unbekannten $X_1, \dots, X_{n-1}$. Induktiv können somit Zeugen $W_1, W_2[W_1], \dots, W_n[W_1, \dots, W_{n-1}]$ gefunden werden. Daher beschäftigen wir uns im Rest dieser Arbeit nur mit Formelgleichungen in einer Unbekannten und sprechen einfach von *Formelgleichungen* bzw. *Zeugen*.

Ein dem Lösen von Formelgleichungen verwandtes Problem ist das Problem der Elimination von Quantoren zweiter Stufe: Sei X eine Relationsvariable und $\varphi[X] \in P$. Gesucht ist ein $\psi \in P$ mit

$$\models (\exists X \varphi[X]) \leftrightarrow \psi.$$

Eine stärkere Variante dieses Problems lautet: Gesucht ist ein $W[\overline{x}] \in P$, sodass

$$\models (\exists X \varphi[X]) \leftrightarrow \varphi[W]. \tag{2}$$

Definition 3 (Zeuge der Quantorenelimination, QE-Zeuge)**.** Wir nennen ein $W[\overline{x}] \in P$, das (2) erfüllt, einen *Zeugen der Quantorenelimination* oder *QE-Zeugen* der Formelgleichung $\exists X \varphi[X]$.

Das Problem der Quantorenelimination lässt sich auf diese Variante zurückführen, da mit einem QE-Zeugen $W[\overline{x}] \in P$ bereits $\varphi[W]$ die gesuchte P-Formel ist. Da die Elimination von Quantoren zweiter Stufe nicht entscheidbar ist, ist auch diese Variante des Problems nicht entscheidbar. Auch das Lösen von Formelgleichungen lässt sich auf dieses Problem zurückführen, wenn nach dem Finden von $W[\overline{x}]$ die Gültigkeit der P-Formel $\varphi[W]$ ermittelt werden kann.

Ein wesentliches Hilfsmittel für das Lösen aller dieser Probleme ist das Lemma von Ackermann. Um es zu formulieren, benötigen wir

Definition 4 (Literale, positive/negative Formeln)**.** Wir nennen Atomformeln und deren Negation *Literale*. Sei R ein Relationssymbol oder eine Relationsvariable. Wir nennen Literale, die R enthalten auch *R-Literale*. Eine Formel $\varphi \in P$ heißt *positiv in Bezug auf* R, falls in der Negationsnormalform von φ kein R-Literal in φ innerhalb einer Negation liegt. Eine Formel $\varphi \in P$ heißt *negativ in Bezug auf* R, falls in der Negationsnormalform von φ jedes R-Literal in φ innerhalb einer Negation liegt.

Satz 5 (Lemma von Ackermann)**.** *Sei X eine Relationsvariable mit Stelligkeit n, $\overline{x}$ ein Tupel von n verschiedenen Variablen. Seien weiters $\alpha^+[X], \alpha^-[X] \in P$, wobei $\alpha^+[X]$ positiv und $\alpha^-[X]$ negativ in Bezug auf X ist. Außerdem sei $W[\overline{x}]$ eine P-Formel, die X nicht beinhaltet. Dann gelten*

$$\models \exists X \left(\forall \overline{x} \left(W[\overline{x}] \to X(\overline{x})\right) \wedge \alpha^-[X]\right) \leftrightarrow \alpha^-[W] \tag{6}$$

und

$$\models \exists X \left(\forall \overline{x} \left(X(\overline{x}) \to W[\overline{x}]\right) \wedge \alpha^+[X]\right) \leftrightarrow \alpha^+[W]. \tag{7}$$

Beweis. Siehe [GSS08]. □

Das Lemma von Ackermann erlaubt damit nicht nur die Reduktion einer Formel in der Prädikatenlogik zweiter Stufe auf eine Formel in der Prädikatenlogik erster Stufe, sondern liefert sogar einen QE-Zeugen für die entsprechende Formelgleichung.

Darüber hinaus können wir $\models \exists X(\varphi \vee \psi) \leftrightarrow (\exists X \varphi) \vee (\exists X \psi)$ ausnutzen. In [EHW17] wird eine Methode zur Kombination von QE-Zeugen von Disjunkten für quantorenfreie φ, ψ beschrieben, die in [Wer17] auf allgemeine Formeln in der Prädikatenlogik erster Stufe ausgeweitet wird. Das entsprechende Resultat ist

Satz 8 (EHW-Kombination)**.** *Seien X eine a-stellige Relationsvariable, $n \in \mathbb{N}$ und P-Formeln $\varphi_1[X], \ldots, \varphi_n[X]$ gegeben. Darüberhinaus sei $\overline{x}$ ein Tupel von a verschiedenen Variablen und für $i \in \{1, \ldots, n\}$ sei $W_i[\overline{x}]$ ein QE-Zeuge der Formelgleichung $\exists X \varphi_i[X]$. Dann ist*

$$W[\overline{x}] := \bigwedge_{i=1}^{n} \left(\left(\bigwedge_{j=1}^{i-1} \neg\varphi_j[W_j]\right) \wedge \varphi_i[W_i] \to W_i[\overline{x}]\right)$$

ein QE-Zeuge der Formelgleichung

$$\exists X \left(\bigvee_{i=1}^{n} \varphi_i[X]\right).$$

Beweis. Siehe [Wer17]. □

3 Algorithmus

In diesem Kapitel präsentieren wir einen Algorithmus basierend auf dem DLS-Algorithmus in [GSS08], der für das Lösen von Formelgleichungen adaptiert ist. Der ursprüngliche DLS-Algorithmus ist ein Algorithmus zur Elimination von Quantoren zweiter Stufe und basiert auf dem Lemma von Ackermann. Da das Lemma von Ackermann auch einen QE-Zeugen für die entsprechende Formelgleichung liefert, liegt es nahe zu untersuchen, ob eine Variante des DLS-Algorithmus sogar Formelgleichungen lösen kann. Der erste Abschnitt beschäftigt sich mit der Beschreibung des Algorithmus und der zweite Abschnitt illustriert den Algorithmus anhand einiger Beispiele. Der letzte Abschnitt geht auf die Unterschiede zwischen dem DLS-Algorithmus und dem präsentierten Algorithmus ein. Eine Implementierung des Algorithmus lässt sich in GAPT (General Architecture for Proof Theory) [1] finden.

3.1 Beschreibung des Algorithmus

Wir betrachten eine Formelgleichung $\exists X \varphi[X]$. Um einen QE-Zeugen der Formelgleichung zu finden, wollen wir $\varphi[X]$ in eine Form bringen, die es erlaubt, das Lemma von Ackermann (5) in Kombination mit Satz 8 anzuwenden. Der Algorithmus lässt sich in vier Phasen aufteilen:

1. Trennen von positiven und negativen Literalen,
2. Extraktion von QE-Zeugen,
3. Kombination der QE-Zeugen,
4. Simplifikationen,
5. Überprüfung der Gültigkeit des Kandidaten.

Das Fehlschlagen des Algorithmus bedeutet nicht, dass die gegebene Formelgleichung nicht lösbar ist, sondern lediglich, dass dieser Algorithmus keine Lösung findet. Da das behandelte Problem unentscheidbar ist, müssen wir damit rechnen, dass gewisse Formelgleichungen mit dem präsentierten Algorithmus nicht gelöst werden können.

3.1.1 Trennen von positiven und negativen Literalen

Ziel dieser Phase ist es, die Formelgleichung $\exists X \varphi[X]$ in eine äquivalente Formel der Form

$$\bigvee_{i=1}^{n} \left(\exists X(\alpha_i^+[X] \land \alpha_i^-[X])\right) \text{ mit } \alpha_i^+[X] \text{ positiv und } \alpha_i^-[X] \text{ negativ in Bezug auf } X \quad (9)$$

[1] https://www.logic.at/gapt/

zu bringen. Im Folgenden stehen ψ, ω und ρ für Subformeln von $\varphi[X]$. Die Schritte dieser Phase sind

1. Negationsnormalform von $\varphi[X]$ berechnen,
2. All- und Existenzquantoren mit den Äquivalenzen (10) - (13) so weit wie möglich nach unten bewegen,

$$\forall x\ (\psi \wedge \omega) \leftrightarrow (\forall x\ \psi) \wedge (\forall x\ \omega) \tag{10}$$

$$\exists x\ (\psi \wedge \omega) \leftrightarrow (\exists x\ \psi) \vee (\exists x\ \omega) \tag{11}$$

$$Qx\ (\psi\ C\ \omega) \leftrightarrow (Qx\ \psi)\ C\ \omega \qquad \begin{array}{l}\text{für } Q \in \{\forall, \exists\} \text{ und } C \in \{\wedge, \vee\},\\ \text{falls } x \text{ in } \omega \text{ nicht frei vorkommt.}\end{array} \tag{12}$$

$$Qx\ (\psi\ C\ \omega) \leftrightarrow \psi\ C\ (Qx\ \omega) \qquad \begin{array}{l}\text{für } Q \in \{\forall, \exists\} \text{ und } C \in \{\wedge, \vee\},\\ \text{falls } x \text{ in } \psi \text{ nicht frei vorkommt.}\end{array} \tag{13}$$

3. redundante Quantoren mit (14) eliminieren,

$$Qx\ \psi \leftrightarrow \psi \text{ für } Q \in \{\forall, \exists\}, \text{ falls } x \text{ in } \psi \text{ nicht vorkommt.} \tag{14}$$

4. durch rekursive Anwendung von (15) Disjunkte aus Konjunktionen extrahieren, solange kein Quantor angetroffen wird,

$$\psi \wedge (\omega \vee \rho) \leftrightarrow (\psi \wedge \omega) \vee (\psi \wedge \rho). \tag{15}$$

5. $\exists X$ mit (16) auf die einzelnen Disjunkte verteilen,

$$\exists X(\psi \vee \omega) \leftrightarrow (\exists X \psi) \vee (\exists X \omega). \tag{16}$$

Zusätzlich sind noch allgemeine Simplifikationsschritte sinnvoll, um gewisse Existenzquantoren zu eliminieren, wie zum Beispiel in der Formel

$$\exists x\ (x = a \wedge X(x)),$$

in der der Existenzquantor redundant ist, da x durch a ersetzt werden kann. Sollte die Formelgleichung nach diesen Schritten nicht in der Form (9) sein, schlägt der Algorithmus fehl.

3.1.2 Extraktion von QE-Zeugen

Diese Phase wird auf die einzelnen Disjunkte $\exists X(\alpha_i^+[X] \wedge \alpha_i^-[X])$ angewandt, wobei $\alpha_i^+[X]$ positiv und $\alpha_i^-[X]$ negativ in Bezug auf X ist. Ziel dieser Phase ist es, mithilfe des Lemmas von Ackermann einen QE-Zeugen für die einzelnen Formelgleichungen zu finden. Sei $\overline{x}$ ein Tupel von Variablen der Länge a, wobei a die Stelligkeit von X bezeichnet. Keine der

Variablen in $\overline{x}$ soll in der Formelgleichung vorkommen. Wir definieren induktiv die Funktion $\mathrm{wit}^+ : P \to P \cup \{\mathrm{FAIL}\}$ durch

$$\mathrm{wit}^+(\varphi)[\overline{x}] := \begin{cases} \bot, & \text{falls } X \text{ in } \varphi \text{ nicht vorkommt,} \\ \overline{x} \approx \overline{t}, & \text{falls } \varphi = X(\overline{t}) \text{ für ein Tupel } \overline{t} \text{ von Termen,} \\ \mathrm{wit}^+(\psi) \vee \mathrm{wit}^+(\omega), & \text{falls } \varphi = \psi \wedge \omega \text{ und } \mathrm{wit}^+(\psi), \mathrm{wit}^+(\omega) \neq \mathrm{FAIL}, \\ \exists y \;\; \mathrm{wit}^+(\psi), & \text{falls } \varphi = \forall y \; \psi \text{ und } \mathrm{wit}^+(\psi) \neq \mathrm{FAIL}, \\ \mathrm{wit}^+(\psi) \wedge \neg\omega, & \begin{array}{l} \text{falls } \varphi = \psi \vee \omega \text{ oder } \varphi = \omega \vee \psi, \\ X \text{ kommt in } \omega \text{ nicht vor und} \\ \mathrm{wit}^+(\psi) \neq \mathrm{FAIL}, \end{array} \\ \mathrm{FAIL} & \text{sonst.} \end{cases}$$

Ähnlich definieren wir die Funktion $\mathrm{wit}^- : P \to P \cup \{\mathrm{FAIL}\}$ durch

$$\mathrm{wit}^-(\varphi)[\overline{x}] := \begin{cases} \top, & \text{falls } X \text{ in } \varphi \text{ nicht vorkommt,} \\ \overline{x} \not\approx \overline{t}, & \text{falls } \varphi = \neg X(\overline{t}) \text{ für ein Tupel } \overline{t} \text{ von Termen,} \\ \mathrm{wit}^-(\psi) \wedge \mathrm{wit}^-(\omega), & \text{falls } \varphi = \psi \wedge \omega, \\ \forall y \;\; \mathrm{wit}^-(\psi), & \text{falls } \varphi = \forall y \; \psi, \\ \mathrm{wit}^-(\psi) \vee \omega, & \begin{array}{l} \text{falls } \varphi = \psi \vee \omega \text{ oder } \varphi = \omega \vee \psi, \\ X \text{ kommt in } \omega \text{ nicht vor und} \\ \mathrm{wit}^-(\psi) \neq \mathrm{FAIL}, \end{array} \\ \mathrm{FAIL} & \text{sonst.} \end{cases}$$

Diese Funktionen können QE-Zeugen für die Formelgleichung liefern, denn es gilt

Satz 17 (wit-Funktionen). *Sei $\exists X(\alpha^+[X] \wedge \alpha^-[X])$ eine Formelgleichung, wobei $\alpha^+[X]$ positiv und $\alpha^-[X]$ negativ in Bezug auf X ist. Für* $\mathrm{wit}^+(\alpha^+[X]) \neq \mathrm{FAIL}$ *ist* $\mathrm{wit}^+(\alpha^+[X])$ *ein QE-Zeuge der Formelgleichung $\exists X(\alpha^+[X] \wedge \alpha^-[X])$. Analog, falls* $\mathrm{wit}^-(\alpha^-[X]) \neq \mathrm{FAIL}$, *dann ist* $\mathrm{wit}^-(\alpha^-[X])$ *ein QE-Zeuge der Formelgleichung.*

Für den Beweis zeigen wir zunächst

Lemma 18. *Sei X eine Relationsvariable mit Stelligkeit a und $\alpha[X] \in P$ positiv bzw. negativ in Bezug auf X. Sei zudem $\overline{x}$ ein Tupel von a verschiedenen Variablen, die in $\alpha[X]$ nicht vorkommen. Falls* $\mathrm{wit}^+(\alpha[X]) \neq \mathrm{FAIL}$ *bzw.* $\mathrm{wit}^-(\alpha[X]) \neq \mathrm{FAIL}$, *dann existiert ein $\beta \in P$, das X nicht enthält, mit*

$$\models \alpha[X] \leftrightarrow ((\forall \overline{x} \;\; \mathrm{wit}^+(\alpha[X]) \to X(\overline{x})) \wedge \beta)$$

bzw.

$$\models \alpha[X] \leftrightarrow ((\forall \overline{x} \;\; (X(\overline{x}) \to \mathrm{wit}^-(\alpha[X])) \wedge \beta).$$

Beweis. Wir betrachten die Aussage nur für den Fall, dass $\alpha[X]$ positiv in Bezug auf X ist. Der Fall für $\alpha[X]$ negativ in Bezug auf X ergibt sich aus analogen Überlegungen. Wir führen den Beweis durch Induktion über die Struktur von $\alpha[X]$.

Falls X in $\alpha[X]$ nicht vorkommt, folgt die Aussage mit $\beta = \alpha[X]$ aus

$$\begin{aligned} \models \alpha[X] &\leftrightarrow \top \wedge \alpha[X] \\ &\leftrightarrow (\bot \rightarrow X(\overline{x})) \wedge \alpha[X] \\ &\leftrightarrow (\forall \overline{x}\ (\mathrm{wit}^{+}(\alpha[X]) \rightarrow X(\overline{x}))) \wedge \alpha[X]. \end{aligned}$$

Falls $\alpha[X] = X(\overline{t})$ folgt die Behauptung mit $\beta = \top$ aus

$$\models X(\overline{t}) \leftrightarrow (\forall \overline{x}\ \overline{x} \approx \overline{t} \rightarrow X(\overline{x})).$$

Falls $\alpha[X] = \psi \wedge \omega$ existieren laut Induktionsvoraussetzung β_ψ und β_ω, die X nicht enthalten, mit

$$\begin{aligned} \models \alpha[X] &\leftrightarrow \psi \wedge \omega \\ &\leftrightarrow ((\forall \overline{x}\ (\mathrm{wit}^{+}(\psi) \rightarrow X(\overline{x}))) \wedge \beta_\psi) \wedge ((\forall \overline{x}\ (\mathrm{wit}^{+}(\omega) \rightarrow X(\overline{x}))) \wedge \beta_\omega) \\ &\leftrightarrow \forall \overline{x}\ ((\mathrm{wit}^{+}(\psi) \rightarrow X(\overline{x})) \wedge (\mathrm{wit}^{+}(\omega) \rightarrow X(\overline{x}))) \wedge \beta_\psi \wedge \beta_\omega \\ &\leftrightarrow \forall \overline{x}\ ((\mathrm{wit}^{+}(\psi) \vee \mathrm{wit}^{+}(\omega)) \rightarrow X(\overline{x})) \wedge \beta_\psi \wedge \beta_\omega \\ &\leftrightarrow \forall \overline{x}\ (\mathrm{wit}^{+}(\alpha[X]) \rightarrow X(\overline{x})) \wedge \beta_\psi \wedge \beta_\omega. \end{aligned}$$

Mit $\beta = \beta_\psi \wedge \beta_\omega$ folgt die Behauptung.

Falls $\alpha[X] = \forall y\ \psi$ existiert laut Induktionsvoraussetzung ein β_ψ mit

$$\begin{aligned} \models \alpha[X] &\leftrightarrow \forall y\ \psi \\ &\leftrightarrow \forall y\ ((\forall \overline{x}\ (\mathrm{wit}^{+}(\psi) \rightarrow X(\overline{x}))) \wedge \beta_\psi) \\ &\leftrightarrow \forall (\overline{x}\ (\forall y\ (\neg\, \mathrm{wit}^{+}(\psi) \vee X(\overline{x}))) \wedge \forall y\ \beta_\psi \\ &\leftrightarrow \forall \overline{x}\ ((\forall y\ \neg\, \mathrm{wit}^{+}(\psi)) \vee X(\overline{x})) \wedge \forall y\ \beta_\psi \\ &\leftrightarrow \forall \overline{x}\ (\neg(\exists y\ \ \mathrm{wit}^{+}(\psi)) \vee X(\overline{x})) \wedge \forall y\ \beta_\psi \\ &\leftrightarrow \forall \overline{x}\ ((\exists y\ \ \mathrm{wit}^{+}(\psi)) \rightarrow X(\overline{x})) \wedge \forall y\ \beta_\psi \\ &\leftrightarrow \forall \overline{x}\ (\mathrm{wit}^{+}(\alpha[X]) \rightarrow X(\overline{x})) \wedge \forall y\ \beta_\psi. \end{aligned}$$

Mit $\beta = \forall y\ \beta_\psi$ folgt die Behauptung.

Falls $\alpha[X] = \psi \vee \omega$ (analog $\alpha[X] = \omega \vee \psi$), sodass X in ω nicht vorkommt, existiert laut Induktionsvoraussetzung ein β_ψ mit

$$\begin{aligned} \models \alpha[X] &\leftrightarrow \psi \vee \omega \\ &\leftrightarrow ((\forall \overline{x}\ (\mathrm{wit}^{+}(\psi) \rightarrow X(\overline{x}))) \wedge \beta_\psi) \vee \omega \\ &\leftrightarrow (\forall \overline{x}\ (\neg\, \mathrm{wit}^{+}(\psi) \vee X(\overline{x}) \vee \omega)) \wedge (\beta_\psi \vee \omega) \\ &\leftrightarrow (\forall \overline{x}\ (\neg\, \mathrm{wit}^{+}(\psi) \vee \neg\neg\omega \vee X(\overline{x}))) \wedge (\beta_\psi \vee \omega) \\ &\leftrightarrow (\forall \overline{x}\ (\neg(\mathrm{wit}^{+}(\psi) \wedge \neg\omega) \vee X(\overline{x}))) \wedge (\beta_\psi \vee \omega) \\ &\leftrightarrow (\forall \overline{x}\ ((\mathrm{wit}^{+}(\psi) \wedge \neg\omega) \rightarrow X(\overline{x}))) \wedge (\beta_\psi \vee \omega) \\ &\leftrightarrow (\forall \overline{x}\ (\mathrm{wit}^{+}(\alpha[X]) \rightarrow X(\overline{x}))) \wedge (\beta_\psi \vee \omega). \end{aligned}$$

Mit $\beta = \beta_\psi \vee \omega$ folgt die Behauptung. $\square$

Beweis von Satz 17. Hier beschränken wir uns auf die Aussage über wit^+. Die Aussage über wit^- ergibt sich mit analogen Beweisschritten. Laut Lemma 18, angewandt auf $\alpha^+[X]$, existiert ein $\beta \in P$, das X nicht enthält, mit

$$\models \alpha^+[X] \leftrightarrow ((\forall \overline{x}\ \mathrm{wit}^+(\alpha[X]) \to X(\overline{x})) \land \beta).$$

Einsetzen dieser Äquivalenz in die Formelgleichung $\exists X(\alpha^+[X] \land \alpha^-[X])$ und Anwendung des Lemmas von Ackermann ergibt

$$\begin{aligned}\models \exists X(\alpha^+[X] \land \alpha^-[X]) &\leftrightarrow \exists X((\forall \overline{x}\ \mathrm{wit}^+(\alpha^+[X]) \to X(\overline{x})) \land \beta \land \alpha^-[X]) \\ &\leftrightarrow \alpha^-[\mathrm{wit}^+(\alpha^+[X])] \land \beta.\end{aligned}$$

Aus Lemma 18 folgt zudem $\models \alpha^+[\mathrm{wit}^+(\alpha^+[X])] \leftrightarrow \beta$, also ist $\mathrm{wit}^+(\alpha^+[X])$ ein QE-Zeuge der Formelgleichung $\exists X(\alpha^+[X] \land \alpha^-[X])$. □

Satz 17 erlaubt das Finden von QE-Zeugen für gewisse Formelgleichungen der Form $\exists X(\alpha_i^+[X] \land \alpha_i^-[X])$. Sollte genau eine der wit-Funktionen FAIL liefern, so kann das Ergebnis der anderen verwendet werden. Wenn beide wit-Funktionen einen QE-Zeugen liefern, kann man einen von beiden aussuchen. In diesem Fall ist es zweckmäßig, jenen zu wählen, der kleiner ist. Wenn für ein $i \in \{1, \ldots, n\}$ beide wit-Funktionen FAIL liefern, schlägt der Algorithmus fehl.

3.1.3 Kombination der QE-Zeugen

Konnten QE-Zeugen für alle Formelgleichungen $\exists X(\alpha_i^+[X] \land \alpha_i^-[X])$ gefunden werden, werden sie unter Anwendung von Satz 8 zu einem QE-Zeugen der gesamten Formelgleichung kombiniert. Damit die Größe des kombinierten QE-Zeugen möglichst klein bleibt, bietet es sich an, nur jene QE-Zeugen in die Kombination einzubeziehen, in deren zugehöriger Formelgleichung die gesuchte Variable X tatsächlich vorkommt. Da die EHW-Kombination nicht nur die QE-Zeugen sondern auch den P-Teil der entsprechenden Formelgleichung einbezieht, kann dies zu kleineren QE-Zeugen führen, falls die P-Teile der Formelgleichungen groß sind.

3.1.4 Simplifikationen

Um die Größe des resultierenden QE-Zeugen zu verkleinern, ist es sinnvoll, noch generelle Simplifikationsschritte durchzuführen. In manchen Fällen kann dies helfen, die Rechenzeit zu verkürzen, vor allem wenn eine Formelgleichung in mehreren Variablen zu lösen ist, da in diesem Fall auch mehrere Substitutionen der Zeugen notwendig sind.

3.1.5 Überprüfung der Gültigkeit des Kandidaten

Bis hierhin haben wir einen QE-Zeugen W der Formelgleichung gefunden. Für Zeugen der Formelgleichung verlangen wir aber auch Gültigkeit von $\varphi[W]$. Dazu wird in diesem Schritt die Gültigkeit von $\varphi[W]$ überprüft. In GAPT ist dies beispielsweise mit dem Theorembeweiser *Escargot* möglich. Stellt sich heraus, dass $\varphi[W]$ nicht gültig ist, schlägt der Algorithmus an dieser Stelle fehl.

3.2 Beispiele

Wir illustrieren die Vorgehensweise des Algorithmus anhand einiger Beispiele. Dazu sei $\overline{x}$ ein Tupel von Variablen, das die Argumente des gesuchten Zeugen darstellen soll.

Beispiel 19. $\exists X\ R(a)$. Da X nicht im P-Teil der Formelgleichung vorkommt, sind positive und negative Literale bereits getrennt. Berechnen von $\mathrm{wit}^+(R(a))$ liefert $\bot$ als QE-Zeuge. $\bot$ ist aber kein Zeuge, da $R(a)$ nicht gültig ist.

Beispiel 20. $\exists X\ X(a)$. Positive und negative Literale von X sind getrennt. $\mathrm{wit}^+(X(a))$ liefert den QE-Zeugen $\overline{x}_1 \approx a$, $\mathrm{wit}^-(\top)$ liefert den QE-Zeugen $\top$. Beide sind auch Zeugen der Formelgleichung.

Beispiel 21. $\exists X\ (X(a) \vee \neg X(b))$. Hier liegen zwei Disjunkte vor, deren X-Literale bereits getrennt sind. Wir berechnen $\mathrm{wit}^+(X(a)) = \overline{x}_1 \approx a$, $\mathrm{wit}^-(\top) = \top$, $\mathrm{wit}^+(\top) = \bot$ und $\mathrm{wit}^-(\neg X(b)) = \overline{x}_1 \not\approx b$. Die kleineren QE-Zeugen sind $\top$ und $\bot$. Deren EHW-Kombination liefert $(\top \to \top) \wedge (\neg\top \wedge \neg\bot \to \bot)$, was sich zu $\top$ vereinfacht. $\top$ ist auch Zeuge der Formelgleichung.

Beispiel 22. $\exists X\ ((X(a) \vee \neg X(b)) \wedge (X(c) \vee \neg X(d)))$. Hier sind positive und negative Literale noch nicht getrennt, deshalb extrahieren wir die Disjunkte aus der Konjunktion und erhalten

$$\exists X\ ((X(a) \wedge X(c)) \vee (X(a) \wedge \neg X(d)) \vee (\neg X(b) \wedge X(c)) \vee (\neg X(b) \wedge \neg X(d))).$$

Die kleinsten QE-Zeugen der einzelnen Disjunkte sind $\top$, $\overline{x}_1 \approx a$, $\overline{x}_1 \approx c$ und $\bot$. EHW-Kombination liefert

$$\begin{aligned}
&(\top \wedge \top \to \top)\\
\wedge&(\neg(\top \wedge \top) \wedge (a \approx a \wedge d \not\approx a) \to \overline{x}_1 \approx a)\\
\wedge&(\neg(\top \wedge \top) \wedge \neg(a \approx a \wedge d \not\approx a) \wedge (\neg(b \approx c) \wedge c \approx c) \to \overline{x}_1 \approx c)\\
\wedge&(\neg(\top \wedge \top) \wedge \neg(a \approx a \wedge d \not\approx a) \wedge \neg(\neg(b \approx c) \wedge c \approx c) \wedge (\neg\bot \wedge \neg\bot) \to \bot).
\end{aligned}$$

Nach Simplifikationen erhalten wir den QE-Zeugen $\top$, der auch Zeuge ist.

Beispiel 23. $\exists X \exists y\ X(y)$. Hier kann mit $\mathrm{wit}^-(\top)$ der QE-Zeuge $\top$ gefunden werden, der auch Zeuge ist.

Beispiel 24. $\exists X\ \neg(\exists y\ (X(y,a) \vee \neg X(y,b)))$. Wir berechnen zunächst die Negationsnormalform:

$$\exists X \forall y\ (\neg X(y,a) \wedge X(y,b)).$$

Dann bewegen wir den Allquantor nach unten:

$$\exists X((\forall y\ \neg X(y,a)) \wedge (\forall y\ X(y,b))).$$

Damit sind positive und negative Literale getrennt und wir können die QE-Zeugen

$$\mathrm{wit}^+(\forall y\ X(y,b)) = \exists y\ \overline{x} \approx (y,b)$$

und

$$\mathrm{wit}^-(\forall y\ \neg X(y,a)) = \forall y\ \overline{x} \not\approx (y,a)$$

berechnen. Einsetzen des wit$^+$-QE-Zeugen liefert nach Simplifikationsschritten

$$\forall y \forall z (y,a) \not\approx (z,b).$$

Diese Formel ist nicht gültig, da sie beispielsweise im Modell mit der einelementigen Menge $\{0\}$ nicht gültig ist. Damit sind beide QE-Zeugen keine Zeugen der Formelgleichung.

Beispiel 25. $\exists X \exists y \; (y = a \wedge X(y,b) \wedge \neg X(y,c))$. Simplifikation der Formelgleichung führt zu $\exists X \; (X(a,b) \wedge \neg X(a,c))$. Mit den wit-Funktionen werden die QE-Zeugen $\overline{x} \approx (a,b)$ und $\overline{x} \not\approx (a,c)$ gefunden. Einsetzen des ersten QE-Zeugen liefert nach Simplifikationsschritten $(a,c) \not\approx (a,b)$, was nicht gültig ist, da für gleich belegte b und c die Formel nicht gültig ist. Also sind die QE-Zeugen keine Zeugen.

Beispiel 26. $\exists X \; \exists y_1 \; (X(a) \wedge \neg X(b) \wedge \forall y_2 \exists y_3 \; R(y_1,y_2,y_3))$. Wir bewegen den Existenzquantor $\exists y_1$ nach unten und erhalten

$$\exists X \; ((X(a) \wedge \neg X(b)) \wedge \exists y_1 \forall y_2 \exists y_3 \; R(y_1,y_2,y_3)).$$

Jetzt sind positive und negative Literale getrennt und wir können die QE-Zeugen

$$\mathrm{wit}^+(X(a)) = \overline{x} \approx a$$

und

$$\mathrm{wit}^-(\neg X(b)) = \overline{x} \not\approx b$$

berechnen. Einsetzen des ersten QE-Zeugen liefert

$$b \not\approx a \wedge \exists y_1 \forall y_2 \exists y_3 \; R(y_1,y_2,y_3),$$

was für gleich belegte a und b nicht gültig ist. Also sind die gefundenen QE-Zeugen keine Zeugen.

Beispiel 27. $\exists X_1 \exists X_2 \; ((X_2(a) \wedge \neg X_2(b)) \vee X_1(c))$. Wir behandeln zuerst die innere Formelgleichung $\exists X_2 \; ((X_2(a) \wedge \neg X_2(b)) \vee X_1(c))$. Für das Disjunkt $\exists X_2(X_2(a) \wedge \neg X_2(b))$ bekommen wir einen QE-Zeugen $\overline{x}^2 \approx a$ und für das Disjunkt $\exists X_2 \; X_1(c)$ bekommen wir den QE-Zeugen $\top$. Allerdings ist es nicht notwendig, eine EHW-Kombination der beiden QE-Zeugen zu konstruieren, da X_2 nur im ersten Disjunkt vorkommt. Substitution des X_2-QE-Zeugen in den P-Teil liefert die Formelgleichung $\exists X_1 \; ((a \not\approx b) \vee X_1(c))$. Hier finden wir für X_1 den QE-Zeugen $\top$. Einsetzen der beiden QE-Zeugen ergibt, dass $(\top, \overline{x}^2 \approx a)$ ein Zeuge der Formelgleichung ist.

Beispiel 28. $\exists X((\forall y \; (X(y) \vee R(y))) \wedge \neg X(b))$. Berechnung von $\mathrm{wit}^+(\forall y \; (X(y) \vee R(y)))$ liefert

$$\exists y \; \overline{x} \approx y \wedge \neg R(y).$$

Vereinfachung des QE-Zeugen führt auf $R(\overline{x})$. Einsetzen des QE-Zeugen ergibt

$$(\forall y R(y)) \wedge \neg R(b),$$

was für die Belegung $R \mapsto \emptyset$ nicht gültig ist und damit ist der QE-Zeuge kein Zeuge.

Beispiel 29. $\exists X \exists y\ (X(y,a) \wedge \neg X(y,b))$. Für diese Formelgleichung findet der Algorithmus keinen Zeugen, da positive und negative Literale nicht getrennt werden können.

Beispiel 30. $\exists X((\exists y\ X(y,a)) \wedge (\exists y\ \neg X(y,b))$. Hier sind positive und negative Literale getrennt, aber der Algorithmus findet dennoch keinen Zeugen, da sowohl $\mathrm{wit}^+(\exists y\ X(y,a)) =$ FAIL als auch $\mathrm{wit}^-(\exists y\ \neg X(y,b)) =$ FAIL.

Beispiel 31. $\exists X((\forall y\ (X(y,a) \vee X(y,c))) \wedge (\forall y\ (\neg X(y,b) \vee \neg X(y,d))))$. In diesem Fall schlägt der Algorithmus fehl, da sowohl im positiven als auch im negativen Teil Disjunktionen auftreten, in denen beide Disjunkte X-Literale enthalten.

3.3 Unterschiede zum ursprünglichen DLS-Algorithmus

Ein wesentlicher Unterschied zum ursprünglichen DLS-Algorithmus ist das Fehlen von Skolemisierung. In der ursprünglichen Version dient die Skolemisierung zur Elimination von Existenzquantoren, die innerhalb eines Allquantors stehen. Nach der Anwendung des Lemmas von Ackermann muss dann eine Deskolemisierung stattfinden. Um Problemfälle bei der Deskolemisierung zu vermeiden, lassen wir die Skolemisierung weg. Das führt dazu, dass die Klasse der Formelgleichungen, für die der Algorithmus eine Lösung findet, kleiner wird. Einer genaueren Untersuchung dieser Problematik widmen wir uns im nächsten Kapitel.

Insbesondere folgt daraus, dass Formeln mit Existenzquantoren, die Variablen binden, die als Argument in X auftreten, nicht verarbeitet werden können. Aus diesem Grund werden Existenzquantoren in der ersten Phase des Algorithmus so weit wie möglich nach unten bewegt, um für Formelgleichungen wie $\exists X \exists x(X(a) \wedge R(x))$ dennoch einen QE-Zeugen zu finden.

Ein weiterer Unterschied zum originalen DLS-Algorithmus besteht darin, dass das Verteilen der Konjunktionen über die Disjunktionen rekursiv erfolgt und nicht nur auf der obersten Ebene. Dies kann zu exponentiellem Wachstum der Formeln führen, was vor allem bei Formelgleichungen in mehreren Variablen zu tragen kommt. Allerdings erlaubt uns dies das Lösen aller Formelgleichungen mit quantorenfreiem P-Teil.

4 Ergebnisse

In diesem Kapitel beschäftigen wir uns mit Fällen, in denen der originale DLS-Algorithmus ein Ergebnis liefert, aber der hier präsentierte Algorithmus fehlschlägt und gehen dabei auf Limitationen des Algorithmus ein. Zum Abschluss zeigen wir ein hinreichendes Kriterium für das Finden von QE-Zeugen für Formelgleichungen mit dem vorgestellten Algorithmus.

4.1 Disjunktionen und Existenzquantoren

Eine Konsequenz des Weglassens von Skolemisierung ist, dass die Klasse von Formelgleichungen, für die der Algorithmus eine Lösung findet, kleiner wird. Betrachte die Formelgleichung

$$\exists X((\forall x\ (X(x,a) \lor X(x,b))) \land (\forall x \exists y\ \neg X(x,y)))).$$

Der ursprüngliche Algorithmus verwendet in dieser Situation die Äquivalenz

$$X(x,a) \lor X(x,b) \leftrightarrow \exists z\ ((z \approx a \lor z \approx b) \land X(z)),$$

um mit Disjunktionen umzugehen. Anschließende Skolemisierung des positiven Teils der Formelgleichung ergibt

$$\exists f \exists X((\forall x\ ((f(x) \approx a \lor f(x) \approx b) \land X(x, f(x)))) \land (\forall x \exists y\ \neg X(x,y))).$$

Nach weiteren Umformungen und Anwendung des Lemmas von Ackermann und Simplifikationsschritten lässt sich der QE-Zeuge $x_2 \approx f(x_1)$ finden, wobei x_1 und x_2 für die Argumente der gesuchten Relation stehen. Substitution in den Teil erster Ordnung der Formelgleichung ergibt nach Vereinfachungen

$$\exists f(\forall x\ ((f(x) \approx a \lor f(x) \approx b) \land (\exists y\ y \not\approx f(x)))).$$

Deskolemisierung ergibt

$$\forall x \exists z((z \approx a \lor z \approx b) \land \exists y\ y \not\approx z),$$

was eine P-Formel ist, die äquivalent zur ursprünglichen Formelgleichung ist. Allerdings lässt sich die Deskolemisierung nicht in offensichtlicher Weise auf den gefundenen QE-Zeugen $x_2 \approx f(x_1)$ übertragen, da die Skolemfunktion frei vorkommt, weshalb unser Algorithmus in diesem Fall kein Ergebnis liefert. Aus diesem Grund lassen sich Disjunktionen nur in Fällen verarbeiten, in denen X nur einem Disjunkt vorkommt. Existenzquantoren innerhalb von Allquantoren lassen sich gar nicht verarbeiten.

Ein ähnliches Problem ergibt sich auch bei Existenzquantoren, die nicht innerhalb eines Allquantors stehen. Betrachten wir die Formelgleichung $\exists X((\exists x\ X(x)) \land \neg X(a))$. In diesem

Fall ist das Argument x gebunden und kann ein Zeuge gefunden werden, sollte er nicht abhängig von der Variable x sein. Da bei unserer Methode die Zeugenfindung aber auf dem Gleichsetzen von Argumenttermen und -variablen beruht, kann sie in diesem Fall keinen Zeugen liefern. Betrachten wir die äquivalente Formel $\exists x \exists X\ (X(x) \wedge \neg X(a))$, so lässt sich für die enthaltene Formelgleichung ein QE-Zeuge finden, da x innerhalb der Formelgleichung frei vorkommt. In diesem Sinne ist die Vertauschung von verschiedenartigen Quantoren nicht äquivalent. Skolemisierung kann Abhilfe schaffen, führt aber zu dem gleichen Problem wie vorher.

4.2 Eine Klasse von lösbaren Formelgleichungen

Wir beschreiben eine Klasse Formelgleichungen, für die der vorgelegte Algorithmus einen QE-Zeugen findet:

Satz 32. *Sei $\exists X\ \varphi[X]$ eine Formelgleichung und $\varphi[X]$ habe die Form*

$$\bigvee_{i=1}^{n} (\alpha_i^+[X] \wedge \alpha_i^-[X]) \tag{33}$$

mit $n \in \mathbb{N}$ und

- *kein X-Literal kommt in $\varphi[X]$ innerhalb eines Existenzquantors erster Stufe vor,*
- *für jedes $i \in \{1, \ldots, n\}$ ist $\alpha_i^+[X]$ positiv und $\alpha_i^-[X]$ negativ in Bezug auf X und*
- *für jedes $i \in \{1, \ldots, n\}$ gilt*
  - *$\alpha_i^+[X]$ enthält kein X-Literal oder*
  - *$\alpha_i^-[X]$ enthält kein X-Literal oder*
  - *in jeder Disjunktion in $\alpha_i^+[X] \wedge \alpha_i^-[X]$ kommt höchstens ein X-Literal vor.*

Dann findet der vorgelegte Algorithmus einen QE-Zeugen für $\exists X\ \varphi[X]$.

Beweis. Der Algorithmus kann fehlschlagen, wenn sich $\varphi[X]$ nicht in positive und negative X-Literale trennen lässt. Das ist ausgeschlossen, da $\varphi[X]$ die Form (33) hat.

Für $i \in \{1, \ldots, n\}$ zeigen wir, dass unter den gegebenen Voraussetzungen $\mathrm{wit}^+(\alpha_i^+[X]) \neq \mathrm{FAIL}$ oder $\mathrm{wit}^-(\alpha_i^-[X]) \neq \mathrm{FAIL}$ gilt. Angenommen, es gelte $\mathrm{wit}^-(\alpha_i^-[X]) = \mathrm{FAIL}$, dann zeigen wir per Induktion über die Struktur von $\alpha_i^+[X]$, dass $\mathrm{wit}^+(\alpha_i^+[X]) \neq \mathrm{FAIL}$ gilt.

Falls $\alpha_i^+[X]$ eine Atomformel ist, die X nicht enthält, gilt $\mathrm{wit}^+(\alpha_i^+[X]) = \bot$.

Falls $\alpha_i^+[X]$ eine Atomformel ist, die X enthält, muss $\alpha_i^+[X]$ ein positives X-Literal sein, da $\alpha_i^+[X]$ selbst positiv ist. In dem Fall gilt auch $\mathrm{wit}^+(\alpha_i^+[X]) \neq \mathrm{FAIL}$.

Die folgenden Fälle setzen voraus, dass $\alpha_i^+[X]$ zumindest ein X-Literal enthält, da ansonsten ohnehin $\mathrm{wit}^+(\alpha_i^+[X]) = \bot$ gilt.

Falls $\alpha_i^+[X] = \psi \wedge \omega$, dann gilt laut Induktionsvoraussetzung $\mathrm{wit}^+(\psi), \mathrm{wit}^+(\omega) \neq \mathrm{FAIL}$ und $\mathrm{wit}^+(\alpha_i^+[X])$ ist laut Definition ungleich FAIL.

Falls $\alpha_i^+[X] = \forall y\ \psi$, dann gilt laut Induktionsvoraussetzung $\mathrm{wit}^+(\psi) \neq \mathrm{FAIL}$ und damit laut Definition auch $\mathrm{wit}^+(\alpha_i^+[X]) \neq \mathrm{FAIL}$.

Falls $\alpha_i^+[X] = \psi \vee \omega$, dann enthält laut Voraussetzung entweder ψ oder ω ein X-Literal, aber nicht beide, da zudem $\alpha_i^-[X]$ ein X-Literal enthalten muss. Dann gilt laut Definition und Induktionsvoraussetzung $\mathrm{wit}^+(\alpha_i^+[X]) \neq \mathrm{FAIL}$.

Der Fall $\alpha_i^+[X] = \exists y\ \psi$ kann laut Voraussetzung nicht eintreten. □

Damit folgt insbesondere

Korollar 34. *Sei $\exists X\ \varphi[X]$ eine Formelgleichung und $\varphi[X]$ quantorenfrei. Dann findet der vorgelegte Algorithmus einen QE-Zeugen für $\exists X\ \varphi[X]$.*

Beweis. Durch das Extrahieren der Disjunktionen in Schritt 4 von Phase 1 des Algorithmus wird eine disjunktive Normalform von $\varphi[X]$ berechnet, die die Voraussetzungen von Satz 32 erfüllt. □

Das angegebene Kriterium ist hinreichend aber nicht notwendig, wie Beispiel 25 zeigt, da etwaige Simplifikationsschritte nicht berücksichtigt werden.

Literaturverzeichnis

[EHW17] S. Eberhard, S. Hetzl, and D. Weller. Boolean unification with predicates. *Journal of Logic and Computation*, 27:109–128, 2017.

[GSS08] D. M. Gabbay, R. Schmidt, and A. Szalas. *Second Order Quantifier Elimination: Foundations, Computational Aspects and Applications*, chapter Direct Methods. College Publications, 2008.

[Gus96] J. Gustafsson. An implementation and optimization of an algorithm for reducing formulae in second-order logic. Technical Report LiTH-MAT-R-96-04, Department of Mathematics, Linkoping University, 1996.

[Wer17] C. Wernhard. The boolean solution problem from the perspective of predicate logic - extended version. *ArXiv*, abs/1706.08329, 2017.