Leibniz Universität Hannover
Institut für Theoretische Informatik

Bachelorarbeit

Äquivalenz von deterministischen Kellerautomaten

vorgelegt von

Tim Ole Christian Hagen
Matrikelnummer: 10013585

Betreuer	Anselm Haak, M. Sc.
Erstprüfer	Prof. Dr. Heribert Vollmer
Zweitprüfer	PD. Dr. Arne Meier

26. August 2020

Inhaltsverzeichnis

1 Einleitung

Die Äquivalenz von zwei deterministischen Kellerautomaten ist entscheidbar. 2001 veröffentlichte Stirling einen Beweis ([Sti01]) in dem dies bewiesen wird. Dafür wird das Problem auf die Äquivalenz der Sprachen von zwei Startkonfigurationen einer strikt deterministischen kontextfreien Grammatik nach Harrison und Havel ([HH73]) reduziert. Der Beweis konstruiert nichtdeterministisch ein Tableau und kombinierte zwei Semientscheidungsalgorithmen zu einem Entscheidungsalgorithmus.

Ziel dieser Arbeit ist es, den Beweis für die Entscheidbarkeit der Äquivalenz von deterministischen Kellerautomaten, wie er in [Sti01] beschrieben ist, nachzuvollziehen. Grundlegende Definitionen werden dafür von [Vol19] – in gegebenenfalls angepasster Form – übernommen. Weitere Erkenntnisse und Definitionen stammen maßgeblich aus Stirlings beiden Beweisen ([Sti01] und [Sti02]).

Die grobe Beweisstruktur ist in Abbildung 1 abgebildet und wird in den folgenden Kapiteln weiter behandelt. Abschnitte 2 und 3 enthalten grundlegende Definitionen und beschreiben die Übersetzung der Kellerautomaten in eine spezielle Form der kontextfreien Grammatiken. Auf Basis dieser Form können weitere Annahmen getroffen und neue Definitionen in Abschnitt 4 eingeführt werden, die schlussendlich in Abschnitt 5 verwendet werden, um einen Algorithmus zu beschreiben, der berechnet, ob die ursprünglichen Kellerautomaten dieselbe Sprache entscheiden.

Zusätzlich ist in Abschnitt 6 jeweils ein Beispiel für ein positives und ein negatives Ergebnis des gesamten Entscheidungsprozesses gegeben. Abschnitt 7 umreißt abschließend eine 2002 von Stirling präsentierte Alternative zu dem hier nachvollzogenen Verfahren.

*Kontextfreie Grammatik

Abbildung 1: Die Beweisstruktur, um zu prüfen, ob die Automaten M_1 und M_2 äquivalent sind.

2 Deterministische Kellerautomaten

Ein deterministischer Kellerautomat (DKA) ist ein 5-Tupel, bestehend aus einer endlichen Menge von *Zuständen* Z, dem endlichen *Eingabealphabet* Σ, dem endlichen *Kelleralphabet* Γ, der *Überführungsfunktion* $\delta\colon Z \times \Gamma \times (\Sigma \cup \{\varepsilon\}) \to Z \times \Gamma^*$ und der *Startkonfiguration* $K \in Z \times \Gamma^+$.

Eine Konfiguration $k \in Z \times \Gamma^*$ eines DKA ist eine *Momentaufnahme* des Automaten und gibt seinen aktuellen Zustand und Kellerinhalt an. Das oberste Kellersymbol steht in der Konfiguration zuerst.

Der Lesbarkeit halber wird zusätzlich eine neue Notation von [Sti01, S. 1] übernommen, die die Veränderung der aktuellen Konfiguration durch eine Eingabe beschreibt.

Definition 1. Für einen Kellerautomaten $M = (Z, \Sigma, \Gamma, \delta, K)$ schreibe $pS \xrightarrow{a}_M q\alpha$ falls $\delta(p, S, a) = (q, \alpha)$ und wenn $pS \xrightarrow{a}_M q\alpha$, dann wird auch $pS\beta \xrightarrow{a}_M q\alpha\beta$ für beliebige $\beta \in \Gamma^*$ geschrieben.

Die durch $\to_M$ beschriebene Arbeitsweise ist äquivalent zu der Erläuterung in [Vol19, S. 16] und wird zusätzlich intuitiv für die Eingabe von Worten erweitert. Wenn M aus dem Zusammenhang klar ist, wird $\to$ ohne Index notiert. δ muss zudem nicht total sein. Wenn ein Übergang nicht definiert ist, dann terminiert der Automat und verwirft. Da δ ε-Transitionen (Transitionen ohne das Einlesen eines Eingabezeichens) erlaubt, muss für Determinismus gegeben sein, dass, wenn eine ε-Transition möglich ist, keine andere Transition möglich ist:

$$pS \xrightarrow{\varepsilon} q\alpha \land pS \xrightarrow{a} r\beta \implies a = \varepsilon.$$

Die Sprache, die eine Maschine akzeptiert, ist dann die Menge der Worte, deren Abarbeitung in einem leeren Keller resultiert:

$$L(M) = \{w \in \Sigma^* \mid \exists q \in Z\colon K \xrightarrow{w}_M q\varepsilon\}.$$

Im Gegensatz zu Vollmers Definition der deterministischen Kellerautomaten, die anhand von Endzuständen akzeptiert ([Vol19, S. 16]), akzeptieren die hier verwendeten DKA also nur eine Unterklasse der deterministischen kontextfreien Sprachen, die präfixfreien deterministischen kontextfreien Sprachen:

$$w \in L(M) \implies \nexists v \in \Sigma^+\colon wv \in L(M),$$

da, wenn w den Keller leert, keine weitere Transition möglich ist und somit jede weitere Eingabe verworfen wird. Dies ist aber keine Einschränkung, da für jeden DKA, der durch Endzustände akzeptiert, ein DKA mit Akzeptanz durch leeren Stack konstruiert werden kann.

Lemma 2. Das Problem der Äquivalenz von DKA mit Akzeptanz durch Endzustände kann auf das Problem der Äquivalenz von DKA mit Akzeptanz durch leeren Stack reduziert werden.

Beweis. Sei $M = (Z, \Sigma, \Gamma, \delta, z_0, \#, E)$ ein DKA nach [Vol19, S. 16]. Wir konstruieren wie folgt einen DKA M' mit Akzeptanz durch leeren Stack:

1. $M' \coloneqq (Z \cup \{z_e\}, \Sigma \cup \{\$\}, \Gamma \cup \{\square\}, \delta', z_0\#\square)$[1]
2. δ' ist δ mit den neuen Transitionen:
   - $pS \xrightarrow{\$} z_e\varepsilon$ für jedes $p \in E$ und $S \in \Gamma \cup \{\square\}$
   - Der Zustand z_e löscht den Keller: $z_eS \xrightarrow{\varepsilon} z_e\varepsilon$

[1] O.B.d.A. gilt $z_e \notin Z$, $\$ \notin \Sigma$, $\square \notin \Gamma$

Es gilt, dass $w \in L(M) \iff w\$ \in L(M')$, da

$$
\begin{array}{lll}
a \in L(M) & \Longrightarrow & M \text{ ist nach Eingabe } a \text{ in Zustand } q \in E \\
& \Longrightarrow & M' \text{ ist nach Eingabe } a \text{ in Zustand } q \in E \\
& \Longrightarrow & M' \text{ ist nach Eingabe } a\$ \text{ in Zustand } z_e \text{ mit leerem Stack} \\
& \Longrightarrow & a\$ \in L(M') \\
a \notin L(M) & \Longrightarrow & \text{① } M \text{ ist nach Eingabe } a \text{ in Zustand } q \notin E \text{ oder} \\
& & \text{② } M \text{ liest } a \text{ nicht vollständig, da der Keller vorher geleert wird oder} \\
& & \text{③ } M \text{ liest } a \text{ nicht vollständig, da keine passende Transition existiert.}
\end{array}
$$

Für die drei Fälle ①, ② und ③ ergibt sich:

$$
\begin{array}{lll}
① & \Longrightarrow & M' \text{ ist nach Eingabe } a \text{ in Zustand } q \notin E \text{ mit nichtleerem Keller} \\
& \Longrightarrow & M' \text{ terminiert bei Eingabe } a\$, \text{ da } \delta' \text{ für diese Transition nicht definiert ist} \\
& \Longrightarrow & a\$ \notin L(M') \\
② & \Longrightarrow & M \text{ liest das echte Präfix } a' \text{ von } a \text{ und terminiert mit leerem Stack} \\
& \Longrightarrow & M' \text{ liest } a' \text{ und hat den Kellerinhalt } \square \\
& \Longrightarrow & M' \text{ terminiert, da } \delta' \text{ für diese Transition nicht definiert ist} \\
& \Longrightarrow & a\$ \notin L(M') \\
③ & \Longrightarrow & M \text{ terminiert, da } \delta \text{ für diese Transition nicht definiert ist} \\
& \Longrightarrow & M' \text{ terminiert, da } \delta' \text{ für diese Transition nicht definiert ist} \\
& \Longrightarrow & a\$ \notin L(M')
\end{array}
$$

Daraus folgt direkt, dass sich Äquivalenz von DKA mit Akzeptanz durch Endzustände auf Äquivalenz von DKA mit Akzeptanz durch leeren Keller reduzieren lässt. □

Beispiel 3. Ein einfaches Beispiel für einen Kellerautomaten ist $M = (Z, \Sigma, \Gamma, \delta, pS)$ mit $Z = \{p, q\}, \Sigma = \{a, b\}$, $\Gamma = \{S, A, B\}$ und den Transitionen

$$
\begin{array}{rll}
\delta: & pS \xrightarrow{a} pA, & pS \xrightarrow{b} pAB, \\
& pA \xrightarrow{\varepsilon} q\varepsilon, & qB \xrightarrow{b} q\varepsilon
\end{array}
$$

Die Sprache $L(M)$ ist $\{a, bb\}$.

Es wird im Folgenden angenommen, dass die DKA die Sprache $\{\varepsilon\}$ nicht akzeptieren und der Automat in der Normalform ist, dass für jede Transition $pS \xrightarrow{a} q\alpha$ gilt, dass $\alpha < 3$ und ε-Transitionen nur von dem Stack löschen können ($\alpha = \varepsilon$). Dass die behandelten DKA die Sprache $\{\varepsilon\}$ nicht akzeptieren, ist ein Nebeneffekt der Übersetzung von oben und jeder Automat, dessen Transitionen die geforderte Normalform nicht erfüllen, kann in einen äquivalenten Automaten übersetzt werden, der die Normalform erfüllt.

Um das zu zeigen, werden in Abbildung 2 Beispiele für Transitionen gegeben, die gegen diese Normalform verstoßen und Möglichkeiten veranschaulicht, wie die Normalform hergestellt werden kann.

Lemma 4. Jeder DKA M kann in einen äquivalenten DKA M' übersetzt werden, dessen ε-Transitionen nur vom Stack löschen und für dessen Transitionen $pS \xrightarrow{a}_{M'} q\alpha$ gilt, dass $\alpha < 3$.

Beweis nach [Sti02, S. 3]. Sei $M = (Z, \Sigma, \Gamma, \delta, K)$ ein DKA, der entsprechend in einen DKA $M' = (Z, \Sigma, \Gamma', \delta', K)$ übersetzt wird, indem einzelne endliche Sequenzen von Kellersymbolen $S_1 S_2 \dots S_n$ durch ein einziges Kellersymbol $[S_1 S_2 \dots S_n]$ ersetzt werden, das sich identisch zu der ursprünglichen Sequenz verhält. Zudem werden ε-Transitionen, die

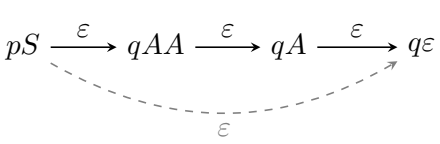
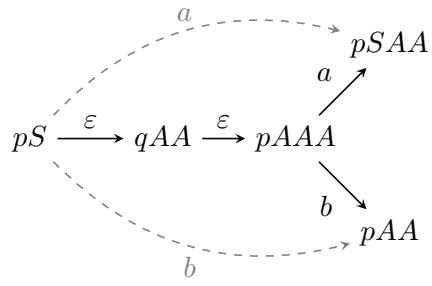
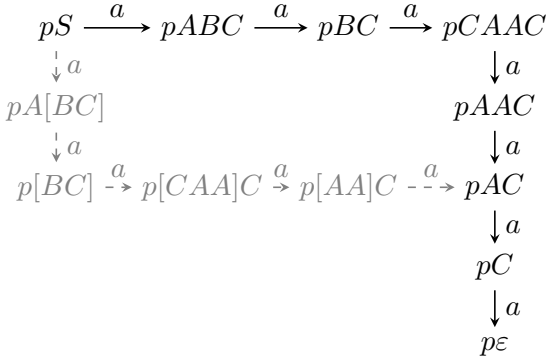
Transitionen		Mögliche Lösung
$pS \xrightarrow{\varepsilon} qAA$ $qA \xrightarrow{\varepsilon} q\varepsilon$		$pS \xrightarrow{\varepsilon} q\varepsilon$ $qA \xrightarrow{\varepsilon} q\varepsilon$
$pS \xrightarrow{\varepsilon} qAA$ $qA \xrightarrow{\varepsilon} pAA$ $pA \xrightarrow{a} pS$ $pA \xrightarrow{b} p\varepsilon$		$pS \xrightarrow{a} pSAA$ $pS \xrightarrow{b} pAA$ $qA \xrightarrow{a} pSA$ $qA \xrightarrow{b} pA$ $pA \xrightarrow{a} pS$ $pA \xrightarrow{b} p\varepsilon$
$pS \xrightarrow{\varepsilon} pSA$	$pS \xrightarrow{\varepsilon} pSA \xrightarrow{\varepsilon} \cdots \xrightarrow{\varepsilon} pSAA\ldots$	Keine Transitionen
$pS \xrightarrow{a} pABC$ $pA \xrightarrow{a} p\varepsilon$ $pB \xrightarrow{a} pCAA$ $pC \xrightarrow{a} p\varepsilon$		$pS \xrightarrow{a} pA[BC]$ $pA \xrightarrow{a} p\varepsilon$ $pB \xrightarrow{a} pC[AA]$ $pC \xrightarrow{a} p\varepsilon$ $p[BC] \xrightarrow{a} p[CAA]C$ $p[AA] \xrightarrow{a} pA$ $p[CAA] \xrightarrow{a} p[AA]$

Abbildung 2: Exemplarische Herstellung der Normalform

nicht nur vom Stack löschen, durch äquivalente Transitionen ersetzt. Die Übersetzung ist exemplarisch in Abbildung 2 dargestellt und wird im Folgenden genauer beschrieben.

1. Es gibt 3 Arten von ε-Transitionen:
   - **a)** $\underline{pS \xrightarrow{\varepsilon} p_1S_1\alpha_1 \xrightarrow{\varepsilon} \ldots \xrightarrow{\varepsilon} p_n\varepsilon}$: die Transition $pS \xrightarrow{\varepsilon} p_1S_1\alpha_1$ wird durch $pS \xrightarrow{\varepsilon} p_n\varepsilon$ ersetzt.
   - **b)** $\underline{pS \xrightarrow{\varepsilon} p_1S_1\alpha_1 \xrightarrow{\varepsilon} \ldots \xrightarrow{\varepsilon} p_nS_n\alpha_n}$ und für p_nS_n gibt es keine ε-Transition: dann wird $pS \xrightarrow{\varepsilon} p_1S_1\alpha_1$ ersetzt durch $pS \xrightarrow{a} q\beta\alpha_n$ für alle Transitionen $p_nS_n \xrightarrow{a} q\beta$.
   - **c)** $\underline{pS \xrightarrow{\varepsilon} p_1S_1\alpha_1 \xrightarrow{\varepsilon} \ldots}$: die Transition $pS \xrightarrow{\varepsilon} p_1S_1\alpha_1$ kann entfernt werden, da sie nie akzeptiert.
2. Für jede Transition $pS \xrightarrow{a} qS_1S_2\ldots S_n$ mit $n \geq 3$:
   - Füge das Kellersymbol $[S_2\ldots S_n]$ hinzu
   - Modifiziere die Transition, sodass $pS \xrightarrow{a} qS_1[S_2\ldots S_n]$
3. Füge neue Transitionen (und ggf. Stacksymbole) hinzu, sodass wenn $pS_1 \xrightarrow{a} q\alpha$ in δ für alle Nichtterminale $[S_1\beta]$ gilt, dass δ' eine Transition $p[S_1\beta] \xrightarrow{a} q[\alpha][\beta]$ enthält
4. Falls $[S] \in \Gamma$ und S ist ein einzelnes Kellersymbol, dann ersetze alle $[S]$ durch S und entferne $[S]$ aus Γ

Die Konfiguration $p[S_1\ldots S_n]$ verhält sich aufgrund der Konstruktion äquivalent zu der Konfiguration $pS_1\ldots S_n$, sodass sich die Sprache des Automaten nicht ändert. □

3 Strikt deterministische kontextfreie Grammatiken

Harrison und Havel führen in [HH73, S. 242] strikt deterministische kontextfreie Grammatiken als eine eingeschränkte Form der deterministischen kontextfreien Grammatiken ein.

Definition 5. Eine deterministische kontextfreie Grammatik (KFG) in 3-Greibach-Normalform (3-GNF) ist ein Tupel

$$G = (V, \Sigma, P)$$

aus der endlichen Menge der *Nichtterminale* V, dem *Terminalalphabet* Σ und der Produktionsmenge $P \subseteq V \times \Sigma \times V^{\leq 2}$ mit $V^{\leq n} \coloneqq \bigcup_{0 \leq i \leq n} V^i$.

Im Vergleich zu [Vol19, S. 4f.] entfällt damit die *Startvariable* $S \in V$ aus dem Tupel, da der von Stirling präsentierte Entscheidungsalgorithmus, ob zwei DKA äquivalent sind, das Problem auf die Äquivalenz der Sprachen zweier Konfigurationen derselben Grammatik reduziert und die Startvariable der Grammatik nicht verwendet. Zudem besteht die rechte Seite jeder Produktion immer aus einem Terminalzeichen gefolgt von Nichtterminalen. In Vollmers Definition haben die Produktionen die Form $X \to \alpha$ mit $X \in V$ und $\alpha \in (\Sigma \cup V)^+$. Analog können die hier verwendeten Produktionen notiert werden als $X \to a\alpha$ mit $X \in V, a \in \Sigma$ und $\alpha \in V^{\leq 2}$.

Bemerkung 6. Stirling hat die Repräsentation der Automaten als strikt deterministische Grammatiken gewählt, da diese äquivalent sind zu ε-freien DKA, die nur einen Zustand haben [Sti02, S. 6]. Im Folgenden erinnert die Notation daher mehr an einen Automaten als an eine Grammatik.

Definition 7. Eine *einfache Konfiguration* der Grammatik $G = (V, \Sigma, P)$ ist eine endliche Folge von Nichtterminalen: $\alpha \in V^*$.

Analog zu der Definition für DKA wird die Notation $\to$ auch für KFG in 3-Greibach-Normalform definiert.

Definition 8. Sei $G = (V, \Sigma, P)$ eine Grammatik. Schreibe $X \xrightarrow{a}_G \alpha$ falls $X \to a\alpha \in P$ und wenn $X \xrightarrow{a}_G \alpha$, dann schreibe auch $X\beta \xrightarrow{a}_G \alpha\beta$. Für Worte $w = a_1 \dots a_n$ gilt $\alpha \xrightarrow{w}_G \beta$ falls $\alpha \xrightarrow{a_1}_G \dots \xrightarrow{a_n}_G \beta$.

Definition 9. Die *Sprache*, die eine einfache Konfiguration α der Grammatik $G = (V, \Sigma, P)$ akzeptiert, ist definiert als $L(\alpha) = \{w \in \Sigma^* \mid \alpha \xrightarrow{w}_G \varepsilon\}$.

$\to_G$ wird auch ohne den Index G notiert, falls G aus dem Zusammenhang klar ist. Ein Nichtterminal X, für das $L(X)$ die leere Menge ist, heißt redundant, da es keine Worte akzeptiert. Die hier behandelten Grammatiken besitzen keine redundanten Nichtterminale[2]. Damit existiert für jedes Nichtterminal X ein kleinstes Wort $w(X) \in L(X)$. Wenn mehrere Worte derselben Länge möglich sind, bezeichnet $w(X)$ das lexikographisch erste.

Definition 10. $M_G \coloneqq \max\{|w(X)| \mid X \in V\}$ ist die *maximale Norm* der Grammatik $G = (V, \Sigma, P)$.

Auch M_G wird ohne Index notiert, wenn G aus dem Kontext eindeutig ist.

Definition 11. Sei $G = (V, \Sigma, P)$. Eine Partitionierung $\equiv$ von V^* heißt *strikt* falls

1. Für $\alpha, \beta \in V^*$ gilt $\alpha \equiv \beta$ gdw. $\alpha = \beta$ oder $\exists \delta \in V^*\colon \alpha = \delta X \alpha_1 \wedge \beta = \delta Y \beta_1$ mit $X \equiv Y$ und $X \neq Y$ (X und Y sind beliebige Nichtterminale)
2. Für alle Paare von Produktionen $X \to a\alpha$ und $Y \to a\delta$ mit $X \equiv Y$ und $a \in \Sigma$ gilt:

[2] Dies ergibt sich direkt aus der Übersetzungsprozedur, die folgt.

a) $\alpha \equiv \delta$
b) Wenn zusätzlich $\alpha = \delta$, dann $X = Y$

Punkt 1 aus der Definition für die Partitionierung ist nicht in Harrison und Havels originaler Definition enthalten, da die Partitionierung dort über $V \cup \Sigma$ definiert wurde (vgl. [HH73, S. 242]). Harrison und Havel bezeichnen X und Y aus Punkt 1 als das „unterscheidende Paar von α und β“ ([HH73, S. 244]). Punkt 1 kann äquivalent ausgedrückt werden als „$\alpha \equiv \beta$ wenn ihre Sprachen präfixfrei sind und für das unterscheidende Paar X, Y $X \equiv Y$ gilt“. Die zweite Anforderung besagt, dass unterschiedliche Produktionen aus Nichtterminalen, die in derselben Partition von $\equiv$ liegen, nur auf Sequenzen von Nichtterminalen abbilden dürfen, die ebenfalls in derselben Partition liegen, aber nicht identisch sind. Streng nach Stirling ist Punkt 2 die Anforderung, damit eine Partitionierung von V strikt ist. Punkt 1 erweitert diese Partitionierung dann auf Sequenzen von Nichtterminalen.

Definition 12. Eine *strikt deterministische kontextfreie Grammatik* $G = (V, \Sigma, P)$ ist eine KFG, für die eine strikte Partitionierung $\equiv$ von V^* existiert.

Harrison und Havel wählten die Bezeichnung „strikt deterministisch“ für diese Klasse der kontextfreien Sprachen, da es sich um die Sprachen handelt, die von einem deterministischen Kellerautomaten (entsprechend der Definition in [Vol19]) entschieden werden können und zusätzlich präfixfrei sind. Die Entscheidung, ob der Automat terminiert, wird also eindeutig in Abhängigkeit von der aktuellen Konfiguration geprüft und bedarf keinem Vorgriff auf die Information, ob es ein nächstes Kellersymbol gibt (vgl. [HH73, S. 242]).

Beispiel 13. Ein Beispiel für eine strikt deterministische kontextfreie Grammatik ist $G = (\{S, B, C\}, \{a, b, c\}, P)$ mit den Produktionen P:

$$\begin{array}{ll} S \xrightarrow{a} B & B \xrightarrow{b} \varepsilon \\ S \xrightarrow{a} C & C \xrightarrow{c} \varepsilon \end{array}$$

Für $\equiv$ sind zwei Partitionierungen möglich: $\{\{S, B, C\}\}$ und $\{\{S\}, \{B, C\}\}$. B und C müssen allerdings in derselben Partition sein, da $S \equiv S$. Die gegebene Grammatik wäre zudem nicht strikt deterministisch wenn die Transition $C \xrightarrow{c} \varepsilon$ zu $C \xrightarrow{b} \varepsilon$ geändert würde.

Wie in Beispiel 13 präsentiert, erlaubt die Definition von strikt deterministischen Grammatiken Nichtdeterminismus. Eine Grammatik kann die Produktionen $X \xrightarrow{a} \alpha$ und $X \xrightarrow{a} \beta$ beinhalten solange $\alpha \equiv \beta$. Um die Produktionen deterministisch auszudrücken, wird eine neue Art der Konfigurationen eingeführt, sodass die Produktion die Form $X \xrightarrow{a} \alpha + \beta$ annimmt.

Definition 14. Eine *zusammengesetzte Konfiguration* E ist eine endliche Menge von einfachen Konfigurationen und wird notiert wie folgt: $E = \{\alpha_1, \ldots, \alpha_n\} = \alpha_1 + \cdots + \alpha_n$.

Definition 15. Eine zusammengesetzte Konfiguration $\alpha_1 + \cdots + \alpha_n$ heißt *zulässig*, falls für alle i und j gilt, dass $\alpha_i \equiv \alpha_j$.

Die Sprache einer zusammengesetzten Konfiguration ist die Vereinigung der Sprachen der einfachen Konfigurationen, die die zusammengesetzte Konfiguration bilden: $L(\alpha_1 + \cdots + \alpha_n) = \bigcup_{1 \leq i \leq n} L(\alpha_i)$. Die Länge einer zusammengesetzten Konfiguration E ist die Länge der längsten einfachen Konfiguration in E: $|\alpha_1 + \cdots + \alpha_n| = \max\{|\alpha_i| \mid 1 \leq i \leq n\}$ und $|\emptyset| := 0$. Die Notation $\to_G$ wird für zusammengesetzte Konfigurationen erweitert, sodass $\alpha_1 + \cdots + \alpha_n \xrightarrow{w}_G \{\alpha' \mid \alpha_i \xrightarrow{w}_G \alpha', 1 \leq i \leq n\}$. Wenn es also keine Produktionen gibt, sodass $\alpha \xrightarrow{w}_G \alpha'$ möglich ist, kann $\alpha \xrightarrow{w}_G \emptyset$ geschrieben werden.

Aufgrund von Punkt 2 in Definition 11 ist die rechte Seite jeder Produktion einer strikt deterministischen KFG zulässig.

Harrison und Havel zeigen, dass DKA mit Akzeptanz durch leeren Stack äquivalent sind zu strikt deterministischen KFG, indem sie Prozeduren zur Übersetzung zwischen beiden beschreiben. Sei M ein DKA. Berechne G_M und ihre Startkonfiguration S_M wie folgt (G_M heißt die *kanonische Grammatik* von M):

1. Nichtterminale sind Tripel $[\mathtt{pSq}]$ aus $p, q \in Z$ und $S \in \Gamma$
2. Transitionen werden wie folgt übersetzt[3] ($a \neq \varepsilon$)
   - Übersetze $pS \xrightarrow{a} q\varepsilon$ zu $[\mathtt{pSq}] \xrightarrow{a} \varepsilon$
   - Übersetze $pS \xrightarrow{a} qT$ zu $[\mathtt{pSr}] \xrightarrow{a} [\mathtt{qTr}]$ für alle $r \in Z$
   - Übersetze $pS \xrightarrow{a} qTU$ zu $[\mathtt{pSr}] \xrightarrow{a} [\mathtt{qTp'}][\mathtt{p'Ur}]$ für alle $r, p' \in Z$
3. ε-Nichtterminale ($[\mathtt{pSq}]$ mit $pS \xrightarrow{\varepsilon} q\varepsilon$) werden von der rechten Seite jeder Produktion entfernt
4. Alle Transitionen, die redundante Nichtterminale enthalten, werden entfernt
5. Die Startkonfiguration $pS_1 \dots S_n$ wird übersetzt zu

$$S_M = \sum_{p,p_1,\dots,p_n \in Z} [\mathtt{pS_1p_1}][\mathtt{p_1S_2p_2}] \dots [\mathtt{p_{n-1}S_np_n}]$$

   - Entferne alle einfach Konfigurationen, die redundante Nichtterminale enthalten
   - Entferne alle ε-Nichtterminale
6. Wenn eine Transition mehrfach vorkommt, werden ihre rechten Seiten zu einer zusammengesetzten Konfiguration vereinigt

Lemma 16. Für jeden DKA M gilt, dass G_M strikt deterministisch ist und S_M ist zulässig.

Beweis angelehnt an [HH73, S. 260]. Definieren die Partitionierung $\equiv$ mit $X \equiv Y$ gdw. $X = [\mathtt{pSp'}]$ und $Y = [\mathtt{pSp''}]$ und erweitere diese Partitionierung gemäß Definition 11.1 für V^*.

Seien $[\mathtt{pSr}] \xrightarrow{a} \alpha$ und $[\mathtt{pSr'}] \xrightarrow{a} \delta$ zwei Transitionen. Offenbar ist $[\mathtt{pSr}] \equiv [\mathtt{pSr'}]$. Entsprechend der Konstruktion von G_M müssen diese Transition aus derselben Regel $pS \xrightarrow{a} q\alpha'$ übersetzt worden sein. Es können die 3 Fälle unterschieden werden:

- Wenn $|\alpha'| = 0$: dann gilt $\alpha = \delta = \varepsilon$ und $[\mathtt{pSr}] = [\mathtt{pSr'}] = [\mathtt{pSq}]$
- Wenn $|\alpha'| = 1$: α und δ könnten ungleich ε sein mit $\alpha = [\mathtt{qTr}]$ und $\delta = [\mathtt{qTr'}]$ und damit ist $\alpha \equiv \delta$. Falls zusätzlich $r = r'$ gilt, dann ist $\alpha = [\mathtt{qTr}] = [\mathtt{qTr'}] = \delta$ und $[\mathtt{pSr}] = [\mathtt{pSr'}]$. Ansonsten könnte $\alpha = \varepsilon$ oder $\delta = \varepsilon$ gelten (o.B.d.A. sei $\alpha = \varepsilon$), wenn α ursprünglich ein entferntes ε-Nichtterminal $[\mathtt{qTr}]$ gewesen wäre, da M eine Transition $qT \xrightarrow{\varepsilon} r\varepsilon$ enthielt. Aufgrund des Determinismus von M sind also alle Transitionen, die $[\mathtt{qTr'}]$ mit $r \neq r'$ enthielten, gestrichen worden, da $[\mathtt{qTr'}]$ redundant war. Somit gilt $\alpha = \delta = \varepsilon$ und $[\mathtt{pSr}] = [\mathtt{pSr'}]$.
- Wenn $|\alpha'| = 2$: Sei $|\alpha| = |\delta| = 2$, dann ist $\alpha = [\mathtt{qTq'}][\mathtt{q'Ur}]$ und $\delta = [\mathtt{qTq''}][\mathtt{q''Ur'}]$. Somit gilt wieder $\alpha \equiv \delta$ und wenn $\alpha = \delta$, dann ist $r = r'$, sodass wieder $[\mathtt{pSr}] = [\mathtt{pSr'}]$. Für den Fall, dass α oder δ mindestens ein ε-Nichtterminal beinhaltet haben, gilt eine analoge Argumentation wie im Fall $|\alpha'| = 1$.

Die Partitionierung $\equiv$ ist per Definition 11 also strikt. Damit existiert eine strikte Partitionierung von V^* für G_M ergo ist G_M strikt deterministisch. Es gilt zuletzt zu zeigen, dass S_M zulässig ist. Aufgrund der Konstruktion von S_M enthalten zwei verschiedene einfache Konfigurationen in S_M jeweils mindestens ein Nichtterminalpaar $[\mathtt{pSq}]$ und $[\mathtt{pSq'}]$, dass sie unterscheidet ($q \neq q'$), aber in derselben Partition liegt ($[\mathtt{pSq}] \equiv [\mathtt{pSq'}]$ gilt per Definition von $\equiv$). □

[3] Die Übersetzung zielt darauf ab, dass $[\mathtt{pSq}] \xrightarrow{w} \varepsilon$ gdw. $pS \xrightarrow{w} q\varepsilon$

Lemma 17. Für jeden DKA M gilt, dass $L(M) = L(S_M)$

Beweis. Definiere

$$H \coloneqq \{\alpha \in V^* \mid \alpha = [\mathtt{p}\mathtt{S}_1\mathtt{p}_1][\mathtt{p}_1\mathtt{S}_2\mathtt{p}_2]\dots[\mathtt{p}_{n-1}\mathtt{S}_n\mathtt{p}_n] \text{ für } n \geq 0\}$$
$$ {}_xN \coloneqq \{[\mathtt{p}\mathtt{S}\mathtt{q}] \in N \mid p = x\}$$
$$ N_x \coloneqq \{[\mathtt{p}\mathtt{S}\mathtt{q}] \in N \mid q = x\}$$
$$f\colon H \to \Gamma^* \text{ mit } f([\mathtt{p}\mathtt{S}_1\mathtt{p}_1][\mathtt{p}_1\mathtt{S}_2\mathtt{p}_2]\dots[\mathtt{p}_{n-1}\mathtt{S}_n\mathtt{p}_n]) = S_1S_2\dots S_n$$

Bevor ε-Nichtterminale entfernt werden, gilt, dass für beliebige $w \in \Sigma^*$:

$$[\mathtt{p}\mathtt{S}\mathtt{q}] \xrightarrow{w} \alpha \text{ gdw entweder } \alpha = \varepsilon \text{ und } pS \xrightarrow{w} q\varepsilon$$
$$\text{oder } pS \xrightarrow{w} p'f(\alpha) \text{ und } \alpha \in {}_{p'}NH \cap HN_q$$

Der Beweis dieser Aussage erfolgt durch Induktion über die Wortlänge angefangen bei $[\mathtt{p}\mathtt{S}\mathtt{q}] \xrightarrow{a} \alpha$ mit $a \in \Sigma$. Aufgrund der gewählten Normalform gibt es für α nur die drei Möglichkeiten: ε, $[\mathtt{r}\mathtt{T}\mathtt{q}]$ und $[\mathtt{r}\mathtt{T}\mathtt{p}'][\mathtt{p}'\mathtt{U}\mathtt{q}]$. Für alle drei Möglichkeiten ist schnell zu sehen, dass die Aussage aufgrund der Konstruktion gilt, solange keine ε-Nichtterminale entfernt wurden. Im Induktionsschritt gilt es zu zeigen, dass die Aussage für $[\mathtt{p}\mathtt{S}\mathtt{q}] \xrightarrow{w} \alpha' \xrightarrow{a} \alpha$ gilt, wenn sie für $[\mathtt{p}\mathtt{S}\mathtt{q}] \xrightarrow{w} \alpha'$ gilt. Offenbar ist $\alpha' \neq \varepsilon$. Aufgrund der Induktionsannahme gilt $pS \xrightarrow{w} p'f(\alpha')$ mit $\alpha' \in {}_{p'}NH \cap HN_q$. Für das Resultat der Anwendung von dem Eingabesymbol a sind 2 Fälle zu unterscheiden:

- Wenn $\alpha = \varepsilon$: α' ist also ein einfaches Nichtterminal $[\mathtt{p}'\mathtt{X}\mathtt{q}]$ und entsprechend muss es eine Produktion $[\mathtt{p}'\mathtt{X}\mathtt{q}] \xrightarrow{a} \varepsilon$ geben. Die gibt es genau dann, wenn der Automat eine Transition $p'X \xrightarrow{a} q\varepsilon$ hat. Entsprechend gilt: $pS \xrightarrow{wa} q\varepsilon \iff [\mathtt{p}\mathtt{S}\mathtt{q}] \xrightarrow{wa} \varepsilon$.
- Wenn $\alpha \in {}_rNH \cap HN_{r'}$: Es gilt, dass $r' = q$. Wenn α' ein einzelnes Nichtterminal ist, begründet es sich in den Transitionen, da die rechte Seite von Produktionen eines Nichtterminals in N_q immer auf ein Nichtterminal in N_q endet. Für den Fall, dass α' aus mehreren Nichtterminalen besteht gilt es, da nur das vorderste Nichtterminal durch eine Produktion geändert wird. Es gilt also: $[\mathtt{p}\mathtt{S}\mathtt{q}] \xrightarrow{wa} \alpha \iff pS \xrightarrow{wa} rf(\alpha)$. Die rechte Seite, $rf(\alpha)$, ist durch die Übersetzung gewährleistet.

Die Sprache bleibt zudem gleich wenn ε-Nichtterminale durch ε ersetzt werden, da ε-Nichtterminale keine Eingabe lesen und direkt auf ε abbilden. Dass $L(M) = L(S_M)$ folgt dann direkt aus der Beobachtung von oben. □

Beispiel 18. Die Übersetzung wird an dem Kellerautomaten aus Beispiel 3 präsentiert. Das Ergebnis der Übersetzung der Transitionen (mit redundanten Nichtterminalen und ε-Nichtterminalen) ist:

$$[\mathtt{p}\mathtt{S}\mathtt{p}] \xrightarrow{a} [\mathtt{p}\mathtt{A}\mathtt{p}] \qquad [\mathtt{p}\mathtt{S}\mathtt{q}] \xrightarrow{a} [\mathtt{p}\mathtt{A}\mathtt{q}]$$
$$[\mathtt{p}\mathtt{S}\mathtt{p}] \xrightarrow{b} [\mathtt{p}\mathtt{A}\mathtt{p}][\mathtt{p}\mathtt{B}\mathtt{p}] \quad [\mathtt{p}\mathtt{S}\mathtt{p}] \xrightarrow{b} [\mathtt{p}\mathtt{A}\mathtt{q}][\mathtt{q}\mathtt{B}\mathtt{p}] \quad [\mathtt{p}\mathtt{S}\mathtt{q}] \xrightarrow{b} [\mathtt{p}\mathtt{A}\mathtt{p}][\mathtt{p}\mathtt{B}\mathtt{q}] \quad [\mathtt{p}\mathtt{S}\mathtt{q}] \xrightarrow{b} [\mathtt{p}\mathtt{A}\mathtt{q}][\mathtt{q}\mathtt{B}\mathtt{q}]$$
$$[\mathtt{q}\mathtt{B}\mathtt{q}] \xrightarrow{b} \varepsilon$$

Im nächsten Schritt werden alle ε-Nichtterminale ($[\mathtt{p}\mathtt{A}\mathtt{q}]$) durch ε ersetzt.

$$[\mathtt{p}\mathtt{S}\mathtt{p}] \xrightarrow{a} [\mathtt{p}\mathtt{A}\mathtt{p}] \qquad [\mathtt{p}\mathtt{S}\mathtt{q}] \xrightarrow{a} \varepsilon$$
$$[\mathtt{p}\mathtt{S}\mathtt{p}] \xrightarrow{b} [\mathtt{p}\mathtt{A}\mathtt{p}][\mathtt{p}\mathtt{B}\mathtt{p}] \quad [\mathtt{p}\mathtt{S}\mathtt{p}] \xrightarrow{b} [\mathtt{q}\mathtt{B}\mathtt{p}] \quad [\mathtt{p}\mathtt{S}\mathtt{q}] \xrightarrow{b} [\mathtt{p}\mathtt{A}\mathtt{p}][\mathtt{p}\mathtt{B}\mathtt{q}] \quad [\mathtt{p}\mathtt{S}\mathtt{q}] \xrightarrow{b} [\mathtt{q}\mathtt{B}\mathtt{q}]$$
$$[\mathtt{q}\mathtt{B}\mathtt{q}] \xrightarrow{b} \varepsilon$$

Zuletzt gilt es, Transitionen zu entfernen, die redundante Nichtterminale enthalten (lediglich $[\mathtt{p}\mathtt{S}\mathtt{q}]$ und $[\mathtt{q}\mathtt{B}\mathtt{q}]$ sind produktiv):

$$[\mathtt{p}\mathtt{S}\mathtt{q}] \xrightarrow{a} \varepsilon \quad [\mathtt{p}\mathtt{S}\mathtt{q}] \xrightarrow{b} [\mathtt{q}\mathtt{B}\mathtt{q}] \quad [\mathtt{q}\mathtt{B}\mathtt{q}] \xrightarrow{b} \varepsilon$$

Die Startkonfiguration wird dann übersetzt zu $[\mathtt{p}\mathtt{S}\mathtt{p}] + [\mathtt{p}\mathtt{S}\mathtt{q}]$. Da $[\mathtt{p}\mathtt{S}\mathtt{p}]$ redundant ist, ist $S_M = [\mathtt{p}\mathtt{S}\mathtt{q}]$. Es ist an dieser Stelle leicht zu sehen, dass $L([\mathtt{p}\mathtt{S}\mathtt{q}]) = \{a, bb\}$.

Nachdem die grundlegenden Begriffe eingeführt wurden, folgen ein paar wichtige Erkenntnisse, die für die weiteren Beweise benötigt werden.

Lemma 19.

1. Falls $\alpha \equiv \beta$ und $\alpha \xrightarrow{w} \alpha'$ und $\beta \xrightarrow{w} \beta'$, dann $\alpha' \equiv \beta'$.
2. Falls $\alpha \equiv \beta$ und $\alpha \xrightarrow{w} \gamma$ und $\beta \xrightarrow{w} \gamma$, dann $\alpha = \beta$.
3. Sei $\alpha_1 + \cdots + \alpha_n \xrightarrow{w} \alpha'_1 + \cdots + \alpha'_n$. Wenn $\alpha_1 + \cdots + \alpha_n$ zulässig ist, dann ist auch $\alpha'_1 + \cdots + \alpha'_n$ zulässig.
4. $L(E) = \emptyset \iff E = \emptyset$.
5. $\alpha \equiv \beta \iff \delta\alpha \equiv \delta\beta$.
6. $\alpha \equiv \beta \wedge \alpha \neq \beta \implies \alpha\gamma \equiv \beta\delta$.

Beweis.

1. Sei $\alpha \equiv \beta$, dann gilt der Satz für $\alpha = \beta$, da $\alpha' = \beta'$. Der Beweis für $\alpha \neq \beta$ erfolgt durch Induktion über die Wortlänge.
   Für den Induktionsanfang $w = \varepsilon$ gilt die Aussage, da $\alpha' = \alpha$ und $\beta' = \beta$. Wenn also $\alpha \equiv \beta$, dann auch $\alpha' \equiv \beta'$. Im Induktionsschritt sei angenommen, dass $w = av$ für $a \in V$ und $v \in V^*$ und die Aussage gilt für v. Da $\alpha \equiv \beta$ und $\alpha \neq \beta$ gibt es $\delta, \alpha_1, \beta_1 \in V^*, X, Y \in V$, sodass $\alpha = \delta X \alpha_1$ und $\beta = \delta Y \beta_1$. Nun können zwei Fälle unterschieden werden:
   - $\underline{\delta = \varepsilon}$: Es gilt, dass $X\alpha_1 \xrightarrow{a} X'\alpha_1 \xrightarrow{v} \alpha'$ und $Y\beta_1 \xrightarrow{a} Y'\beta_1 \xrightarrow{v} \beta'$. Da $\equiv$ strikt ist, gilt $X' \equiv Y'$, und somit ist per Definition $X'\alpha_1 \equiv Y'\beta_1$ und nach der Induktionsannahme ist damit $\alpha' \equiv \beta'$.
   - $\underline{\delta \neq \varepsilon}$: Es gilt, dass $\delta X\alpha_1 \xrightarrow{a} \delta' X\alpha_1 \xrightarrow{v} \alpha'$ und $\delta Y\beta_1 \xrightarrow{a} \delta' Y\beta_1 \xrightarrow{v} \beta'$. Per Definition ist $\delta' X\alpha_1 \equiv \delta' Y\beta_1$ und nach der Induktionsannahme ist damit $\alpha' \equiv \beta'$.
2. Der Beweis erfolgt wieder durch Induktion über die Wortlänge. Für den Induktionsanfang $w = \varepsilon$ gilt die Aussage, da $\gamma = \alpha$ und $\gamma = \beta$ also $\alpha = \beta$. Im Induktionsschritt sei angenommen, dass $w = av$ für $a \in V$ und $v \in V^*$ und die Aussage gilt für v. Es gilt also, dass $\alpha \xrightarrow{a} \alpha' \xrightarrow{v} \gamma$ und $\beta \xrightarrow{a} \beta' \xrightarrow{v} \gamma$. Nach 1 ist, da $\alpha \equiv \beta$, also $\alpha' \equiv \beta'$ und somit gemäß der Induktionsannahme $\alpha' = \beta'$. Es gilt also, dass $\alpha \xrightarrow{a} \gamma' \xrightarrow{v} \gamma$ und $\beta \xrightarrow{a} \gamma' \xrightarrow{v} \gamma$. Wie in dem Beweis zu 1 gilt wieder $\alpha = \delta X\alpha_1$ und $\beta = \delta Y\beta_1$ mit den beiden Fällen:
   - $\underline{\delta = \varepsilon}$: Aus $X\alpha_1 \xrightarrow{a} \gamma'$ und $Y\beta_1 \xrightarrow{a} \gamma'$ folgt, dass $\alpha_1 = \beta_1$ und per Definition von $\rightarrow$ gelten $X\alpha_1 \xrightarrow{a} \gamma''\alpha_1$ und $Y\alpha_1 \xrightarrow{a} \gamma''\alpha_1$. Da $\equiv$ strikt ist, gilt also, dass $X = Y$. Damit ist $\alpha = \beta$, da $\alpha = X\alpha_1$ und $\beta = Y\beta_1$ mit $\alpha_1 = \beta_1$ und $X = Y$.
   - $\underline{\delta \neq \varepsilon}$: Aus $\delta X\alpha_1 \xrightarrow{a} \gamma'$ und $\delta Y\beta_1 \xrightarrow{a} \gamma'$ folgt, dass $\gamma' = \delta' X\alpha_1$ und $\gamma' = \delta' Y\beta_1$ mit $\delta \xrightarrow{a} \delta'$. Also muss gelten, dass $X = Y$ und $\alpha_1 = \beta_1$ und damit $\alpha = \beta$.
3. Folgt direkt aus 1.
4. Offensichtlich gilt $L(\emptyset) = \emptyset$. $L(E) = \emptyset \implies E = \emptyset$ gilt, da es keine redundanten Nichtterminale gibt.
5. Folgt direkt aus Definition 11.1.
6. Sei $\alpha \equiv \beta$ und $\alpha \neq \beta$. Per Definition gibt es ein $\Delta \in V^*$, sodass $\alpha = \Delta X\alpha_1$ und $\beta = \Delta Y\beta_1$. Einsetzen in $\alpha\gamma$ und $\beta\delta$ liefert $\Delta X\alpha_1\gamma$ und $\Delta Y\beta_1\delta$. Per Definition gilt also $\alpha\gamma \equiv \beta\delta$, da dieses Δ laut Annahme existiert. □

Lemma 19.3 erlaubt es, im Folgenden weitestgehend von Zulässigkeit auszugehen, da die Startkonfigurationen zulässig sind und das Anwenden von Worten auf diesen Konfigurationen die Zulässigkeit erhält. Im Verlauf des Beweises, dass die Äquivalenz zweier DKA berechenbar ist, ist neben der Erkenntnis, dass die behandelten Konfigurationen inhärent zulässig sind, die dadurch implizierte Eigenschaft wichtig, dass die Sprachen $L(\alpha)$ und $L(\beta)$ für $\alpha \equiv \beta$ präfixdisjunkt sind – kein Wort der einen Sprache ist Präfix eines Wortes der anderen – (und disjunkt wenn $\alpha \neq \beta$).

Lemma 20. $\alpha \equiv \beta$ und $w \in L(\alpha)$, dann gilt für jedes $u \in \Sigma^+$, dass $wu \notin L(\beta)$.

Beweis. Sei $\alpha \equiv \beta$ und $w \in L(\alpha)$ und $u \in \Sigma^+$ beliebig. Sei $w \in L(\alpha)$, dann gilt $\alpha \xrightarrow{w} \varepsilon$. Sei $\beta \xrightarrow{w} \gamma$, dann gilt nach Lemma 19.1, dass $\gamma \equiv \varepsilon$. Das gilt nur für $\gamma = \varepsilon$.

- Sei $\alpha = \beta$: Für die Konfiguration ε ist keine Produktion definiert, da eine Produktion mindestens ein Nichtterminal voraussetzt, sodass $\beta \xrightarrow{w} \varepsilon \xrightarrow{u} \emptyset$ und damit $wu \notin L(\beta)$.
- Sei $\alpha \neq \beta$: Da $\alpha \xrightarrow{w} \varepsilon$ und $\beta \xrightarrow{w} \gamma$ und damit $\beta \xrightarrow{w} \varepsilon$ gilt, muss nach Lemma 19.2 $\alpha = \beta$ gelten. Das ist ein Widerspruch in der Annahme. Es gilt also, dass $\beta \xrightarrow{w} \emptyset \xrightarrow{u} \emptyset$ und damit $wu \notin L(\beta)$. □

Daraus folgt, dass, wenn eine Konfiguration $\alpha_1 + \cdots + \alpha_n$ zulässig ist und das Wort w akzeptiert, für genau ein α_i gilt, dass $\alpha_i \xrightarrow{w} \varepsilon$ und für alle $j \neq i$ gilt $\alpha_j \xrightarrow{w} \emptyset$.

Bemerkung 21. Nach der Übersetzung der Automaten in strikt deterministische KFG handelt es sich noch immer um zwei separate KFG. Für den weiteren Beweis ist es aber notwendig, dass es sich um eine einzige Grammatik handelt. Daher müssen die beiden Grammatiken in einem zusätzlichen Schritt disjunkt vereinigt werden. Wenn die Mengen der Nichtterminalzeichen oder der Terminalzeichen nicht disjunkt sind, können die betroffenen Zeichen umbenannt werden, sodass eine Vereinigung immer möglich ist.

4 Formen, rekursive Nichtterminale und Äquivalenz

Stirlings Beweis, dass die Äquivalenz von DKA berechenbar ist, begründet sich in der Existenz eines Beweisbaumes (Tableau), der die Äquivalenz zeigt oder widerlegt. Um diesen Beweisbaum und seine Ableitungsregeln beschreiben zu können, ist zuvor notwendig, Operationen zu definieren, die auf zulässigen Konfigurationen angewendet werden können, wie auch den Begriff der Äquivalenz formal zu definieren. Für die Regel CUT wird insbesondere eine neue Form von Nichtterminalen benötigt, die kanonische Familie rekursiver Nichtterminale, die verwendet werden kann, um die Größe von Konfigurationen zu kürzen.

Die folgenden Operationen auf zulässigen Konfigurationen erlauben es, diese auf verschiedene Weise (in verschiedenen *Formen*) zu notieren.

Definition 22.

1. $E + F \coloneqq E \cup F$.
2. $EF \coloneqq \{\alpha\beta \mid \alpha \in E, \beta \in F\}$.

Lemma 23. Sei $E_1 + \cdots + E_n$ eine zulässige Konfiguration und die E_i sind paarweise disjunkt und nicht ε

1. Dann ist $E_1G_1 + \cdots + E_nG_n$ zulässig falls alle G_i zulässig sind.
2. Falls $E_1G_1 + \cdots + E_nG_n$ zulässig ist und $E_i \neq \emptyset$, dann ist G_i ebenfalls zulässig.

Beweis.

1. Angenommen alle G_i sind zulässig und $\alpha \in E_iG_i, \alpha' \in E_jG_j$. Dann gilt per Definition, dass $\alpha = \beta\gamma$ und $\alpha' = \beta'\gamma'$ für $\beta \in E_i, \gamma \in G_i, \beta' \in E_j, \gamma' \in G_j$ mit $\beta \neq \varepsilon$ und $\beta' \neq \varepsilon$. Dann gibt es zwei Fälle:
   - Wenn $\beta \equiv \beta'$ und $\beta \neq \beta'$, dann ist nach Lemma 19.6 $\beta\gamma \equiv \beta'\gamma'$ und somit $\alpha \equiv \alpha'$.
   - Wenn $\beta = \beta'$, dann ist $i = j$ und somit auch $\gamma \equiv \gamma'$ nach Lemma 19.5 gilt also $\beta\gamma \equiv \beta\gamma'$ und somit $\alpha \equiv \alpha'$.
2. Sei $\gamma, \gamma' \in G_i$ und $\beta \in E_i$. Da $E_1G_1 + \cdots + E_nG_n$ zulässig ist, ist $\beta\gamma \equiv \beta\gamma'$ und somit gilt nach Lemma 19.5 $\gamma \equiv \gamma'$. □

Dadurch kann dieselbe Konfiguration in verschiedenen *Formen* dargestellt werden. Wenn des Weiteren $E + F$ geschrieben wird, wird angenommen, dass $E + F$ zulässig ist.

Beispiel 24. Sei $AB + ACD$ eine zulässige Konfiguration, dann bezeichnet $A(B + CD)$ dieselbe zulässige Konfiguration und da A zulässig ist, muss auch $B + CD$ zulässig sein.

Wie einleitend beschrieben, benötigt CUT eine neue Form von Nichtterminalen, die verwendet werden können, um zulässige Konfigurationen zu ersetzen aber Äquivalenz beizubehalten.

Definition 25. $(V_1, \ldots, V_n)$ ist eine *Familie rekursiver Nichtterminale* und für alle $1 \leq i \leq n$ gilt:

1. entweder $V_i \overset{\text{def}}{=} V_j$ für ein $j \leq i$
2. oder $V_i \overset{\text{def}}{=} H_1V_1 + \cdots + H_nV_n$ mit $H_1 + \cdots + H_n$ ist zulässig und die H_k sind paarweise disjunkt und nicht ε.

Die Länge eines rekursiven Nichtterminals V, $|V|$, ist per Definition 1 und ein rekursives Nichtterminal V_i mit $V_i \overset{\text{def}}{=} V_i$ heißt *terminierendes Nichtterminal*. Die Verwendung von rekursiven Nichtterminalen ist zudem sehr beschränkt. Sie dürfen nur als letztes Symbol einer Konfiguration auftreten, und wenn eine Konfiguration einer zulässigen Konfiguration auf ein rekursives Nichtterminal endet, dann müssen alle Konfigurationen auf ein rekursives Nichtterminal derselben Familie enden. Eine Konfiguration ist also zulässig falls sie

$\{\alpha_1, \ldots, \alpha_n\}$ oder $\{\alpha_1 V_1, \ldots, \alpha_n V_n\}$ ist mit $\alpha_i \equiv \alpha_j$ und $\alpha_i \neq \alpha_j$. Zudem wird für den weiteren Beweis eine neue Notation eingeführt: $E \cdot u$ (gelesen „E nach u“).

Definition 26. Falls $E \xrightarrow{u} F$ und F ist kein rekursives Nichtterminal, dann ist $E \cdot u = F$. Wenn F ein rekursives Nichtterminal ist mit $F \stackrel{\text{def}}{=} H$, ist $E \cdot u = H$.

Offenbar ist das rekursive Nichtterminal V_i also terminierend, falls $E \cdot u = V_i$. Zudem ist ein Korrolar der Definition von $\to$ für Worte und der Definition von $\cdot$, dass $E \cdot uv = (E \cdot u) \cdot v$.

Definition 27. Eine zulässige Konfiguration $E = \alpha_1 G_1 + \cdots + \alpha_k G_k$ ist in *n-Head Normal Form* (n-HNF) für $n \geq 1$, wenn gilt:

- Die α_i sind paarweise verschieden und kein α_i ist ε.
- $\alpha_1 + \cdots + \alpha_k$ ist zulässig.
- Falls E keine rekursiven Nichtterminale enthält, gilt entweder $|\alpha_i| = n$ oder $|\alpha_i| < n \wedge G_i = \varepsilon$.
- Falls E rekursive Nichtterminale enthält, gilt entweder $|\alpha_i| = n \wedge |G_i| > 0$ oder $|\alpha_i| < n \wedge G_i$ ist ein einzelnes rekursives Nichtterminal.

Lemma 28. Sei $E = \beta_1 G_1 + \cdots + \beta_k G_k$ in n-HNF, dann gilt:

1. Falls $w \in L(\beta_i)$, dann ist $E \xrightarrow{w} G_i$.
2. Wenn $\beta_i \cdot w$ weder ε noch $\emptyset$ ist, dann ist $E \cdot w = (\beta_1 \cdot w) G_1 + \cdots + (\beta_k \cdot w) G_k$.

Beweis. 1. und 2. folgen direkt aus Lemma 20. □

Weiterführend sind zulässige Konfigurationen $E_1 G_1 + \cdots + E_n G_n$ in *Head Tail Form*: Die E_i sind die Heads und die G_i die Tails. $E_1 + \cdots + E_n$ ist zulässig und die E_i sind paarweise disjunkt. Kein Tail ist $\emptyset$ und kein Head enthält rekursive Nichtterminale oder ist das leere Wort. Wenn des Weiteren V_i geschrieben wird, ist ein beliebiges rekursives Nichtterminal gemeint.

Zwei Konfigurationen heißen äquivalent (geschrieben $E \sim F$) genau dann, wenn sie dieselbe Sprache akzeptieren und (falls sie rekursive Nichtterminale enthalten) bezüglich des terminierenden Nichtterminals übereinstimmen.

Lemma 29. $E \sim F$ genau dann, wenn für alle Worte $w \in \Sigma^*$:

- $E \cdot w = \emptyset \iff F \cdot w = \emptyset$
- und $E \cdot w = V_i \iff F \cdot w = V_i$

Beweis. Sei $E \sim F$ und $E \cdot w = \emptyset$ aber $F \cdot w = F' \neq \emptyset$. Laut Lemma 19.4 gibt es also mindestens ein v sodass $F' \cdot v = \varepsilon$ oder V_i. Damit ist $wv \in F$ aber $wv \notin E$, da $E \cdot wv = \emptyset$. Das ist ein Widerspruch in der Annahme.

Angenommen die Punkte gelten aber $E \not\sim F$. O.B.d.A. ist $E \cdot w = \varepsilon$ und $F \cdot w = F' \neq \varepsilon$ laut Annahme ist F' weder $\emptyset$ noch ein V_i. Damit existiert ein a mit $F' \cdot a \neq \emptyset$. Das führt aber zu einem Widerspruch, da $E \cdot wa = \emptyset$. □

Äquivalenz kann zusätzlich durch n-Äquivalenz angenähert werden, indem lediglich dasselbe Verwurfverhalten und Übereinstimmung bezüglich terminierender Nichtterminale für Worte maximal der Länge n geprüft wird.

Definition 30. $E \sim_n F$ gdw. für alle Worte $w \in \Sigma^*$ mit $|w| \leq n$ gilt, dass $E \cdot w = \emptyset \iff F \cdot w = \emptyset$ und $E \cdot w = V_i \iff F \cdot w = V_i$

Es kann aber nicht jede beliebige Familie rekursiver Nichtterminale verwendet werden, um zulässige Konfigurationen zu ersetzen und Äquivalenz beizubehalten, weswegen eine neue Art der Familien rekursiver Nichtterminale eingeführt wird.

Definition 31. Eine Familie rekursiver Nichtterminale $(V_1, \ldots, V_n)$ heißt *kanonisch* für die Familie $E_1^i G_1 + \cdots + E_n^i G_n \sim F_1^i G_1 + \cdots + F_n^i G_n$ mit $1 \leq i \leq k$ genau dann, wenn:

1. $E_1^i V_1 + \cdots + E_n^i V_n \sim F_1^i V_1 + \cdots + F_n^i V_n$ für alle $1 \leq i \leq k$
2. Falls $V_i \stackrel{\text{def}}{=} H_1 V_1 + \cdots + H_n V_n$, dann $G_i \sim H_1 G_1 + \cdots + H_n G_n$
3. Falls $V_i \stackrel{\text{def}}{=} V_j$, dann $G_i \sim G_j$

Für die Berechnung dieser kanonischen Familien ist es allerdings zuvor notwendig, das Konzept der *Verfeinerung* von Familien rekursiver Nichtterminale zu erläutern. Die Familie $(V_1', \ldots, V_n')$ verfeinert die Familie $(V_1, \ldots, V_n)$ genau dann, wenn:

$$\begin{array}{ll} \text{Wenn } V_i \stackrel{\text{def}}{=} H_1 V_1 + \cdots + H_n V_n & \text{dann } V_i' \stackrel{\text{def}}{=} H_1 V_1' + \cdots + H_n V_n' \\ \text{Wenn } V_i \stackrel{\text{def}}{=} V_j \text{ und } V_i' \stackrel{\text{def}}{=} H & \text{dann } V_j' \stackrel{\text{def}}{=} H \end{array}$$

Die Verfeinerung behält allerdings n-Äquivalenz – und damit auch Äquivalenz – bei.

Lemma 32. Sei $E_1 V_1 + \cdots + E_k V_k \sim_n F_1 V_1 + \cdots + F_k V_k$ und $(V_1', \ldots, V_k')$ verfeinert $(V_1, \ldots, V_k)$, dann ist $E_1 V_1' + \cdots + E_k V_k' \sim_n F_1 V_1' + \cdots + F_k V_k'$.

Beweis. Sei $E = E_1 V_1 + \cdots + E_k V_k, F = F_1 V_1 + \cdots + F_k V_k, E' = E_1 V_1' + \cdots + E_k V_k'$ und $F' = F_1 V_1' + \cdots + F_k V_k'$. Angenommen $E \sim_n F$ aber es gibt ein w mit $|w| \leq n$ und $E' \cdot w = \emptyset$ und $F' \cdot w \neq \emptyset$. Dann gibt es ein längstes Präfix w' von w, für das $F \cdot w' \neq \emptyset$. Das Präfix darf nicht gleich w sein, da sonst $E \cdot w \neq \emptyset$ gilt und gemäß der Konstruktion ist somit $E' \cdot w \neq \emptyset$, ein Widerspruch. Für diesen längsten Präfix gilt also, dass $F \cdot w' = E \cdot w' = V_i$ und V_i ist terminierend (da $F \cdot w = E \cdot w = \emptyset$). Entsprechend muss $F' \cdot w' = E' \cdot w' = H$ sein und $V_i' \stackrel{\text{def}}{=} H$. Das ist ein Widerspruch zu der Annahme. Wenn $F' \cdot w' = E' \cdot w'$, dann muss auch $F' \cdot w = E' \cdot w$ gelten. □

Durch wiederholtes gezieltes Verfeinern der rekursiven Nichtterminalen kann dann unter der Vorraussetzung von Äquivalenz eine kanonische Familie berechnet werden.

Lemma 33. Sei $E_1^i G_1 + \cdots + E_n^i G_n \sim F_1^i G_1 + \cdots + F_n^i G_n$ mit $1 \leq i \leq k$, dann existiert eine kanonische Familie rekursiver Nichtterminale für diese Konfigurationen.

Beweis. Der Beweis erfolgt durch die iterative Berechnung einer kanonischen Familie für gegebenen Konfigurationen und dem simultan präsentierten Korrektheitsbeweis der Berechnung. Sei eingehend die Familie rekursiver Nichtterminale definiert mit $(V_1^0, \ldots, V_n^0)$, wobei jedes Nichtterminal terminierend ist: $V_x^0 \stackrel{\text{def}}{=} V_x^0$. Es ist leicht zu sehen, dass 2 und 3 aus der Definition für diese Familie zutreffen. Wenn zuletzt $E_1^i V_1^0 + \cdots + E_n^i V_n^0 \sim F_1^i V_1^0 + \cdots + F_n^i V_n^0$ für alle i gilt, dann ist 1 ebenso erfüllt und $(V_1^0, \ldots, V_n^0)$ ist die gesuchte kanonische Familie. Sei nun Iteration j betrachtet mit $E_1^i V_1^j + \cdots + E_n^i V_n^j \not\sim F_1^i V_1^j + \cdots + F_n^i V_n^j$, dann gibt es ein kleinstes l und ein kleinstes k, sodass

$$\underbrace{E_1^l V_1^j + \cdots + E_n^l V_n^j}_{E'} \not\sim_k \underbrace{F_1^l V_1^j + \cdots + F_n^l V_n^j}_{F'}$$

mit dem Wort $u = a_1 \ldots a_k$, dass die beiden Konfigurationen unterscheidet. E bezeichnet im Folgenden $E_1^l G_1 + \cdots + E_n^l G_n$ und F ist $F_1^l G_1 + \cdots + F_n^l G_n$. Es ist nicht möglich, dass entweder E' oder F' u verwirft, da $E \sim F$. $E' \cdot u$ und $F' \cdot u$ einigen sich also nicht auf dasselbe rekursive Nichtterminal. Wir betrachten im Folgenden die Sequenzen:

$$Z_X^i = X \cdot a_1 \ldots a_i \text{ mit } X \in \{E, F, E', F'\}$$

Wähle m (falls es existiert), sodass $Z_{E'}^m = H_1 V_1^j + \cdots + H_n V_n^j$ und $Z_{E'}^{m-1} \xrightarrow{a_m} V_i^j$. Also ist $V_i^j \stackrel{\text{def}}{=} H_1 V_1^j + \cdots + H_n V_n^j$ und wegen Anforderunge 2 auch $G_i \sim H_1 G_1 + \cdots + H_n G_n$. Es

gilt zudem, dass $Z_E^m = G_i$, da beide Sequenzen dieselben Heads haben. Die Sequenz Z_E^i wird nun aktualisiert zu

$$Z_E^i = (H_1G_1 + \cdots + H_nG_n) \cdot a_{m+1} \dots a_i,$$

sodass die Sequenzen wieder dieselben Heads haben. Diese Schritte werden für E' (und analog für F') wiederholt bis kein solches m mehr existiert. Es kann nicht nur einer von $Z_{E'}^k$ oder $Z_{F'}^k$ $\emptyset$ sein. Angenommen, $Z_{E'}^k = \emptyset$, dann ist auch $Z_E^k = \emptyset$ und damit auch $Z_F^k = \emptyset$, da $E \sim F$. Damit würde aber auch gelten, dass $Z_{F'}^k = \emptyset$ und somit $E' \sim F'$. Also ist mindestens einer von $Z_{E'}^k$ und $Z_{F'}^k$ ein terminierendes Nichtterminal V_i^j. O.B.d.A. ist $Z_{E'}^k$ dieses Nichtterminal. $Z_{F'}^k$ ist dann entweder ein anderes terminierendes Nichtterminal $V_{i'}^j$ oder eine Konfiguration $H_1'V_1^j + \cdots + H_n'V_n^j$. Aufgrund der Sequenzen ist $Z_E^k = G_i$ und $Z_F^k = G_{i'}$ beziehungsweise $Z_F^k = H_1'G_1 + \cdots + H_n'G_n$. Im letzten Schritt wird die Familie $(V_1^j, \dots, V_n^j)$ dann verfeinert zu $(V_1^{j+1}, \dots, V_n^{j+1})$ mit der entsprechenden Fallunterscheidung von oben:

1. Falls $Z_{F'}^k = V_{i'}^j$: O.B.d.A. sei $i' \leq i$. Für jedes t, für das $V_t^j \stackrel{\text{def}}{=} V_i^j$ definiere $V_t^{j+1} \stackrel{\text{def}}{=} V_{i'}^{j+1}$. Ansonsten werden die Indices j lediglich durch $j+1$ ersetzt. Die Konstruktion erhält die von der Definition geforderten Eigenschaften 2 und 3 für die neue Familie.
2. Falls $Z_{F'}^k = H_1'V_1^j + \cdots + H_n'V_n^j$: Für jedes t, für das $V_t^j \stackrel{\text{def}}{=} V_i^j$ definiere $V_t^{j+1} \stackrel{\text{def}}{=} H_1'V_1^{j+1} + \cdots + H_n'V_n^{j+1}$. Ansonsten werden die Indices j lediglich durch $j+1$ ersetzt. Die Konstruktion erhält die von der Definition geforderten Eigenschaften 2 und 3 für die neue Familie.

Jede Iteration resultiert damit in einer Familie rekursiver Nichtterminale, die die vorherige verfeinert. Mit jeder Iteration wird dabei genau ein terminierendes Nichtterminal verfeinert und die anderen Nichtterminale werden lediglich entsprechend angepasst. Der Algorithmus terminiert also nach $n-1$ Iterationen, da dann jedes rekursive Nichtterminal entsprechend berechnet wurde. □

Der von Stirling präsentierte Algorithmus zur Berechnung einer solchen kanonischen Familie sieht vor, diese, angefangen bei der Familie $(V_1^0, \dots, V_n^0)$, in der jedes rekursive Nichtterminal terminierend ist, schrittweise zu verfeinern bis sie die Anforderungen erfüllt. Kanonische Familien sind damit aber nur für Paare von äquivalenten Konfigurationen definiert und eignen sich daher nicht für einen Tableau, der ebenso mit ambivalenten Konfigurationspaaren umgehen muss. Daher wird eine Approximation von kanonischen Familien definiert, den so genannten *m-Unifiern*, die lediglich m-Äquivalenz voraussetzen.

Sei $E_1^iG_1 + \cdots + E_n^iG_n \sim_m F_1^iG_1 + \cdots + F_n^iG_n$ für jedes i, dann ist garantiert, dass ein m-Unifier existiert. Die Berechnung erfolgt iterativ nach dem angepassten Verfahren aus dem Beweis zu Lemma 33:

> Unterscheide iterativ – angefangen bei $(V_1^0, \dots, V_n^0)$ mit $V_i^0 \stackrel{\text{def}}{=} V_i^0$ und der „Tiefe“ $d_0 = 0$ – für die aktuelle Familie $(V_1^j, \dots, V_n^j)$ und Tiefe d_j – zwischen den beiden Fällen:
>
> 1. Für alle $1 \leq i \leq k$ ist $E_1^iV_1^j + \cdots + E_n^iV_n^j \sim_{m-d_j} F_1^iV_1^j + \cdots + F_n^iV_n^j$: $(V_1^j, \dots, V_n^j)$ ist der gesuchte m-Unifier.
> 2. $\underbrace{E_1^lV_1^j + \cdots + E_n^lV_n^j}_{E'} \not\sim_{m-d_j} \underbrace{F_1^lV_1^j + \cdots + F_n^lV_n^j}_{F'}$ für ein kleinstes l:
>
>   Sei u das kleinste Wort, das zwischen den beiden Konfigurationen unterscheidet. Da $E_1^lG_1^j + \cdots + E_n^lG_n^j \sim_m F_1^lG_1^j + \cdots + F_n^lG_n^j$ kann es nicht sein, dass entweder E' oder F' u verwerfen, entweder $E' \cdot u$ oder $F' \cdot u$ ist also ein terminierendes Nichtterminal (oder beide sind unterschiedlich). O.B.d.A. ist $E' \cdot u = V_i^j$ und $V_i^j \stackrel{\text{def}}{=} V_i^j$:

a) $F' \cdot u = V_{i'}^j$ mit $i \neq i'$. O.B.d.A. sei $i' < i$, dann verfeiner die Familie zu $(V_1^{j+1}, \ldots, V_n^{j+1})$ mit $V_t^{j+1} \stackrel{\text{def}}{=} V_{i'}^{j+1}$ für jedes t, für das $V_t^j \stackrel{\text{def}}{=} V_i^j$, und $V^{j+1} = V^j$ sonst.

b) $F' \cdot u = H_1' V_1^j + \cdots + H_n' V_n^j$ und kein H_i' ist ε. Verfeiner die Familie zu $(V_1^{j+1}, \ldots, V_n^{j+1})$ mit $V_t^{j+1} \stackrel{\text{def}}{=} H_1' V_1^{j+1} + \cdots + H_n' V_n^{j+1}$ für jedes t, für das $V_t^j \stackrel{\text{def}}{=} V_i^j$, und $V^{j+1} = V^j$ sonst.

Die neue Tiefe d_{j+1} ist dann $d_j + |u|$. Da $|u| \leq m - d_j$, ist d_{j+1} nie größer als m.

Fakt 34. Wenn $(V_1, \ldots, V_n)$ für $E_1^i G_1 + \cdots + E_n^i G_n \sim F_1^i G_1 + \cdots + F_n^i G_n$ kanonisch ist, dann ist es auch für $E_1^i J_1 + \cdots + E_n^i J_n \sim F_1^i J_1 + \cdots + F_n^i J_n$ kanonisch mit $1 \leq i \leq k$.

Fakt 35. Wenn $E_1^i G_1 + \cdots + E_n^i G_n \sim F_1^i G_1 + \cdots + F_n^i G_n$, dann existiert ein m', sodass jeder m-Unifier der Heads $(E_1^i + \cdots + E_n^i, F_1^i + \cdots + F_n^i)$ mit $m \geq m'$ kanonisch ist für $E_1^i G_1 + \cdots + E_n^i G_n \sim F_1^i G_1 + \cdots + F_n^i G_n$.

Um die Beziehung zwischen den Tails und den rekursiven Nichtterminalen kanonischer Familien zu veranschaulichen, sei noch ein Beispiel gegeben.

Beispiel 36. Gegeben seien die Konfigurationen $E = E_1 G_1 + E_2 G_2$ und $F = F_1 G_1 + F_2 G_2$.

- Sei $E_1 \sim F_1$ und $E_2 \sim F_2$ (und damit auch $E \sim F$), dann ist die Familie (V_1, V_2) mit $V_i \stackrel{\text{def}}{=} V_i$ kanonisch, da $E_1 V_1 + E_2 V_2 \sim F_1 V_1 + F_2 V_2$.
- Sei $G_1 \not\sim G_2$ und $E_1 \sim F_2$ und $E_2 \sim F_1$, dann ist die Familie (V_1, V_2) mit $V_i \stackrel{\text{def}}{=} V_i$ nicht kanonisch, da $E_1 V_1 + E_2 V_2$ und $F_1 V_1 + F_2 V_2$ sich nicht auf die rekursiven Nichtterminale einigen. Dieser Fall veranschaulicht also, warum diese zusätzliche Anforderung für Äquivalenz geprüft werden muss.
- Sei $G_1 \sim G_2$ und $E_1 \sim F_2$ und $E_2 \sim F_1$, dann ist die Familie (V_1, V_2) mit $V_1 \stackrel{\text{def}}{=} V_1$ und $V_2 \stackrel{\text{def}}{=} V_1$ kanonisch.

Bevor zuletzt der Beweisbaum und seine Konstruktion erläutert werden können, werden noch ein paar Eigenschaften von Äquivalenz und n-Äquivalenz präsentiert, die für die weiteren Beweise notwendig sind.

Lemma 37.

1. $E \sim F \implies E \cdot w \sim F \cdot w$
2. $E \sim E' \wedge F \sim F' \implies E + F \sim E' + F'$
3. $EF \sim G \wedge F \sim F' \implies EF' \sim G$
4. $E \sim_n E' \wedge F \sim_n F' \implies E + F \sim_n E' + F'$
5. $E \sim_n F \wedge F \not\sim_n G \implies E \not\sim_n G$
6. $EF \sim_n G \wedge |E| > 0 \wedge F \sim_{n-1} F' \implies EF' \sim_n G$

Beweis.

1. Angenommen, $E \sim F$ aber $E \cdot w \not\sim F \cdot w$, dann existiert ein kleinstes Wort v, für das nicht gilt, dass $E \cdot w \cdot v = \emptyset \iff F \cdot w \cdot v = \emptyset$ und $E \cdot w \cdot v = V_i \iff F \cdot w \cdot v = V_i$. Damit unterscheidet das Wort wv aber auch E und F, sodass $E \not\sim F$, ein Widerspruch in der Annahme.
2. Für $w \in L(E+F)$ beliebig sei $\alpha_i \in E+F$ die einfache Konfiguration, die w akzeptiert. Da $E + F$ eine zulässige Vereinigung aus E und F ist, ist α_i ebenfalls in E oder in F enthalten. O.B.d.A. ist $\alpha_i \in E$. Da $E \sim E'$ existiert also eine Konfiguration $\alpha_i' \in E'$, die w akzeptiert. Da $E' + F'$ eine zulässige Vereinigung von E' und F' ist, ist dieses α_i' ebenfalls in $E' + F'$ und somit ist $w \in E' + F'$. Die Argumentation ist Analog, wenn $(E + F) \cdot w$ ein terminierendes Nichtterminal ist, denn dann ist $(E' + F') \cdot w$ dasselbe terminierende Nichtterminal.

3. Zuerst sei gezeigt, dass $F \sim F' \implies EF \sim EF'$. Sei w beliebig, dann gibt es zwei Möglichkeiten:

- Fall 1: $E \cdot u = \varepsilon$ für ein Präfix u von w: Sei $w = uv$

$$\begin{aligned} EF \cdot uv = \emptyset &\iff F \cdot v = \emptyset \\ &\iff F' \cdot v = \emptyset \\ &\iff EF' \cdot uv = \emptyset \end{aligned}$$

- Fall 2: $E \cdot u \neq \varepsilon$ für jedes Präfix u von w: Dann gilt für jedes u, v mit $w = uv$ entweder, dass $E \cdot u = \emptyset$ und damit direkt $EF \cdot uv = \emptyset \iff EF' \cdot uv = \emptyset$ oder $E \cdot u = E'$ aber dann muss $u = w$ sein und somit ist $EF \cdot w = E'F$ und $EF' \cdot w = E'F'$ also beide nicht $\emptyset$ und damit $EF \cdot uv = \emptyset \iff EF' \cdot uv = \emptyset$

Analog kann gezeigt werden, dass $EF \cdot w = V_i \iff EF' \cdot w = V_i$ und damit gilt, dass $EF \sim EF'$.

Die zu zeigende Eigenschaft ist ein Korollar dieser Erkenntnis. Da $F \sim F'$ gilt $EF \sim EF'$ und da EF und G sowie EF und EF' bezüglich Akzeptanzverhalten äquivalent sind und bei den terminierenden Nichtterminalen übereinstimmen, stimmen also auch EF' und G überein: $EF' \sim G$.

4. Der Beweis ist komplett Analog zu 2.

5. Sei $w \in \Sigma^*$ mit $|w| \leq n$ ein beliebiges Wort, das F und G unterscheidet. Angenommen $E \sim_n F$, dann gilt nach Definition 30 $G \cdot w \neq \emptyset \iff F \cdot w = \emptyset \iff E \cdot w = \emptyset$ oder $G \cdot w \neq V_i \iff F \cdot w = V_i \iff E \cdot w = V_i$ für jedes terminierende Nichtterminal V_i. Damit unterscheidet w auch E und G, sodass $E \not\sim_n G$.

6. Der Beweis erfolgt weitestgehend analog zu Lemma 37.3. Zuerst wird allgemeiner gezeigt, dass $F \sim_{n-1} F' \wedge |E| > 0 \implies EF \sim_n EF$. Dafür reicht dieselbe Fallunterscheidung (wobei für die Präfixe u gilt, dass $|u| \geq 1$, da $|E| > 0$). Die zu beweisende Aussage folgt dann aufgrund der Transitivität von n-Äquivalenz. □

5 Tableaubeweiser

Um zu prüfen, ob $E \sim F$ gilt, wird ein Tableau konstruiert, das – angefangen bei dem *Goal* $E \stackrel{?}{=} F$ (gelesen „gilt $E \sim F$?“) – iterativ das aktuelle Goal durch Anwendung von Regeln auf *Subgoals* reduziert, bis ein inhärent wahres („erfolgreiches“) oder inhärent falsches („erfolgloses“) Goal erreicht wird.

Definition 38. Regeln sind entweder *einfach*, oder *konditional* und werden wie folgt notiert – wobei C die Kondition ist, unter der sie angewendet werden können. Für C wird in dem konstruierten Tableau der Name der angewandten Regel geschrieben.

Einfache Regeln

$$\frac{\text{Goal}}{\text{Subgoal}_1 \quad \dots \quad \text{Subgoal}_n}\ \text{C}$$

Konditionale Regeln

$$\frac{\begin{array}{c}\text{Goal}_1\\ \vdots\\ \text{Goal}_k\\ \vdots\\ \text{Goal}\end{array}}{\text{Subgoal}}\ \text{C}$$

Eine Regel heißt *complete*, wenn durch ihre Anwendung auf ein wahres Goal alle Subgoals wahr sind. In der Gegenrichtung heißt sie *sound*, falls das Goal wahr ist, wenn alle Subgoals wahr sind.

Definition 39. Sei $E \not\sim_n F$, dann heißt n der *Kontravalenzindex*.

5.1 Intuitive Konstruktion

Stirlings Algorithmus zur Konstruktion des Tableaus basiert auf der Tatsache, dass für eine beliebige Größe $S \geq 0$ die Menge der Goals $E \stackrel{?}{=} F$ mit $|E|, |F| \leq S$ endlich ist. Wenn die Goals bei der Konstruktion des Tableaus immer kleiner als S bleiben, trifft man irgendwann auf eine erfolgreiches (wiederholtes) Goal (siehe Lemma 53). Wird hingegen durch die Anwendung einer Regel E oder F größer als S, dann wird versucht, die Konfiguration zu kürzen, sodass E und F wieder kleiner als S sind. Dafür werden insgesamt drei verschiedene Regeln benötigt: **UNF**old (Abschnitt 5.2) und **BAL**ance (Abschnitt 5.3) werden angewendet, solange E und F kleiner als S sind, und andernfalls wird versucht, **CUT** (Abschnitt 5.4) anzuwenden, um die Konfigurationen zu kürzen. Die Tableaukonstruktion erzeugt einen endlichen Baum, den *Tableau*, mit der zu prüfenden Aussage an der Wurzel und erfolgreichen beziehungsweise erfolglosen Goals als Blätter. Die Wurzel ist wahr genau dann, wenn alle Blätter erfolgreich sind.

5.2 UNF

UNF ist eine einfache Regel, die ein Goal $E \stackrel{?}{=} F$ auf die Subgoals $E \cdot a \stackrel{?}{=} F \cdot a$ für alle $a \in \Sigma$ reduziert. UNF ist complete und sound.

Lemma 40 (UNF ist complete)**.** Falls $E \sim F$, dann ist $E \cdot a \sim F \cdot a$ für $a \in \Sigma$ beliebig.

Beweis. Folgt direkt aus Lemma 37.1. □

Lemma 41 (UNF ist sound und mindert den Kontravalenzindex um 1)**.** Falls $E \not\sim_{m+1} F$, dann existiert ein $a \in \Sigma$ sodass $E \cdot a \not\sim_m F \cdot a$.

Beweis. Angenommen $E \not\sim_{m+1} F$ und für jedes $a \in \Sigma$ gilt $E \cdot a \sim_m F \cdot a$. Daraus folgt, dass $\forall a, \forall w$ mit $|w| \leq m : E \cdot aw = \emptyset \iff F \cdot aw = \emptyset \wedge E \cdot aw = V_i \iff F \cdot aw = V_i$. Vor jedes Wort der Länge m jedes Terminalzeichen anzuhängen, geniert jedes Wort der Länge $m+1$. Damit gilt, dass $\forall w$ mit $|w| \leq m+1 : E \cdot w = \emptyset \iff F \cdot w = \emptyset \wedge E \cdot w = V_i \iff F \cdot w = V_i$ und somit gilt nach Definition 30, dass $E \sim_{m+1} F$, ein Widerspruch in der Annahme. □

Definition 42. Sei $E' \stackrel{?}{=} F'$ ein Subgoal, das durch mehrere konsekutive Anwendungen von UNF von $E \stackrel{?}{=} F$ erreicht wird, dann heißt u das *zugehörige* Wort falls $E' = E \cdot u$ und $F' = F \cdot u$.

$$\frac{E \stackrel{?}{=} F}{E \cdot a_1 \stackrel{?}{=} F \cdot a_1 \quad \dots \quad E \cdot a_n \stackrel{?}{=} F \cdot a_n} \text{ UNF } (\{a_1, \dots, a_n\} = \Sigma)$$

Abbildung 3: UNF

Beispiel 43. Seien die Produktionen $A \xrightarrow{a} AA + C, A \xrightarrow{b} \varepsilon, B \xrightarrow{a} BB, B \xrightarrow{b} \varepsilon, C \xrightarrow{a} \varepsilon$ und $\Sigma = \{a, b\}$ gegeben, dann könnte eine Applikation von UNF wie folgt aussehen:

$$\frac{AA + C \stackrel{?}{=} BB}{AAA + CA + \varepsilon \stackrel{?}{=} BBB \qquad A \stackrel{?}{=} B} \text{ UNF}$$

5.3 BAL

Mithilfe von den BAL Regeln (dargestellt in Abbildung 4) können Goals auf „balancierte" Subgoals reduziert werden.

Definition 44. Ein Goal der Form $E_1G_1 + \dots + E_nG_n \stackrel{?}{=} F_1G_1 + \dots + F_nG_n$ heißt *balanciert* mit dem *Ungleichgewicht* $\max\{|E_i|, |F_i| \mid 1 \leq i \leq n\}$.

BAL(R)

$$\frac{\begin{array}{l} F \stackrel{?}{=} X_1H_1 + \dots + X_kH_k \\ \vdots \\ F' \stackrel{?}{=} E_1H_1 + \dots + E_kH_k \end{array}}{F' \stackrel{?}{=} E_1(F \cdot w(X_1)) + \dots + E_k(F \cdot w(X_k))}$$

BAL(L)

$$\frac{\begin{array}{l} X_1H_1 + \dots + X_kH_k \stackrel{?}{=} F \\ \vdots \\ E_1H_1 + \dots + E_kH_k \stackrel{?}{=} F' \end{array}}{E_1(F \cdot w(X_1)) + \dots + E_k(F \cdot w(X_k)) \stackrel{?}{=} F'}$$

Konditionen:

1. Zwischen dem oberen und unteren Goal liegen exakt $\max\{|w(X_i)| \mid E_i \neq \emptyset, 1 \leq i \leq k\}$ Anwendungen von ausschließlich UNF (mit dem zugehörigen Wort u)
2. $\forall i \in \{1, \dots, k\}$: $E_i = X_i \cdot u$
3. $X_1H_1 + \dots + X_kH_k$ ist in 1-HNF
4. Kein E_i ist ε

Abbildung 4: BAL(L) und BAL(R)

Lemma 45 (BAL ist complete)**.** Falls $X_1H_1 + \dots + X_kH_k \sim F$ und $E_1H_1 + \dots + E_kH_k \sim F'$, dann $E_1(F \cdot w(X_1)) + \dots + E_k(F \cdot w(X_k)) \sim F'$

Beweis. Wenn $X_1H_1+\cdots+X_kH_k \sim F$, dann auch $(X_1H_1+\cdots+X_kH_k)\cdot w(X_i) \sim F\cdot w(X_i)$ und somit $H_i \sim F \cdot w(X_i)$. Nach Lemma 37.2 und Lemma 37.3 gilt also, dass die H_i in $E_1H_1 + \cdots + E_kH_k \sim F'$ jeweils durch $F \cdot w(X_i)$ ersetzt werden können: $E_1(F \cdot w(X_1)) + \cdots + E_k(F \cdot w(X_k)) \sim F'$. □

Lemma 46 (BAL ist sound und behält den Kontravalenzindex)**.**
Sei $X_1H_1 + \cdots + X_kH_k \sim_{n+m} F$ und $E_1H_1 + \cdots + E_kH_k \not\sim_{n+1} F'$ mit $E_i \neq \varepsilon$ und $m = \max\{|w(X_i)| \mid E_i \neq \emptyset\}$, dann ist $E_1(F \cdot w(X_1)) + \cdots + E_k(F \cdot w(X_k)) \not\sim_{n+1} F'$

Beweis. Angenommen $X_1H_1 + \cdots + X_kH_k \sim_{n+m} F$ und $E_1H_1 + \cdots + E_kH_k \not\sim_{n+1} F'$, dann gibt es zwei Fälle zu betrachten:

- Sei $E_i = \emptyset$: offenbar ist $E_i(F \cdot w(X_i)) \sim_{n+1} E_iH_i$.
- Sei $E_i \neq \emptyset$: gemäß der Annahme ist also $|E_i| > 0$ und $|w(X_i)| \leq m$. Da $(X_1H_1 + \cdots + X_kH_k) \cdot w(X_i) = H_i$ ist $F \cdot w(X_i) \sim_n H_i$. Nach Lemma 37.6 gilt damit $E_i(F \cdot w(X_i)) \sim_{n+1} E_iH_i$.

Dadurch ist nach Lemma 37.4 $E_1(F_1\cdot w(X_1))+\cdots+E_k(F\cdot w(X_k)) \sim_{n+1} E_1H_1+\cdots+E_kH_k$ und damit $E_1(F \cdot w(X_1)) + \cdots + E_k(F \cdot w(X_k)) \not\sim_{n+1} F'$ (Lemma 37.5). □

Definition 47. Die Konfiguration, die bei der Anwendung von BAL F aus Abbildung 4 entspricht, heißt die *verwendete* Konfiguration, die E_i heißen *Heads* und die $(F \cdot w(X_i))$ *Tails*. Die Anwendung von BAL hat die *Größe* $|F|$.

Lemma 48. Das Ungleichgewicht (Definition 44) nach einer Anwendung von BAL ist maximal $3M + 2$ (M kommt aus Definition 10).

Beweis. Sei eine Applikation von BAL(R) nach Abbildung 4 gegeben mit dem zu den Anwendungen von UNF zugehörigen Wort u. Offenbar ist $|u| \leq M$ und für alle i ist $|E_i| \leq M + 1$, da $E_i = X_i \cdot u$ und $|X_i| = 1$ und durch Anwendung von u kann der Keller um maximal M wachsen. Sei $\beta_1G_1+\cdots+\beta_nG_n$ die $(M+1)$-HNF von F, dann gilt aufgrund von Lemma 28:

(i) $F \cdot u = (\beta_1 \cdot u)G_1 + \cdots + (\beta_n \cdot u)G_n$ und
(ii) $F \cdot w(X_i) = (\beta_1 \cdot w(X_i))G_1 + \cdots + (\beta_n \cdot w(X_i))G_n$.

Da $F' = F \cdot u$, kann das Subgoal geschrieben werden als $(\beta_1 \cdot u)G_1 + \cdots + (\beta_n \cdot u)G_n \stackrel{?}{=} E_1(F \cdot w(X_1)) + \cdots + E_k(F \cdot w(X_k))$. (ii) kann ebenso für die $F \cdot w(X_i)$ eingesetzt werden. Die resultierende Konfiguration kann also als $E'_1G_1 + \cdots + E'_nG_n$ dargestellt werden mit $E'_i = E_1(\beta_i \cdot w(X_1)) + \cdots + E_n(\beta_i \cdot w(X_n))$. Dadurch gilt, dass

$$|E'_i| \leq \underbrace{M+1}_{|E_j|\leq M+1} + \underbrace{M+1}_{|\beta_j|\leq M+1} + \underbrace{M}_{w(X_j)\leq M} = 3M + 2$$

und damit ist das maximale Ungleichgewicht des Subgoals unabhängig von den Tails durch $3M + 2$ beschränkt. □

Beispiel 49 ([Sti01, S. 15])**.** Gegeben seien die folgenden Produktionen:

$$\begin{array}{lll} A \xrightarrow{a} A & A \xrightarrow{b} \varepsilon & A \xrightarrow{c} A \\ B \xrightarrow{a} \varepsilon & B \xrightarrow{b} \emptyset & B \xrightarrow{c} BB \\ C \xrightarrow{a} \emptyset & C \xrightarrow{b} \varepsilon & C \xrightarrow{c} BC + C \end{array}$$

Damit ist $w(A) = b, w(B) = a$ und $w(C) = b$. BAL(L) sieht dann wie folgt aus (um das Beispiel klein zu halten, wird nur der c-Zweig gezeigt).

$$\dfrac{\dfrac{B \qquad A+ \qquad C\varepsilon \stackrel{?}{=} \quad B(BA+C)+ \qquad C\varepsilon}{BB \qquad A+(BC+C)\varepsilon \stackrel{?}{=} BB(BA+C)+(BC+C)\varepsilon}\ \text{UNF}}{BB(BA+C)+(BC+C)\varepsilon \stackrel{?}{=} BB(BA+C)+(BC+C)\varepsilon}\ \text{BAL(L)}$$

Die Lücken sind dabei der Nachvollziehbarkeit wegen in der Darstellung so gesetzt, dass die einzelnen Heads und Tails mitverfolgt werden können.

5.4 CUT

UNF und BAL reichen nicht aus, um zu garantieren, dass die Konstruktion des Tableaus endlich ist, da die Konfigurationen durch Anwendungen von UNF und BAL weiter wachsen können. CUT erlaubt hingegen, die Goals zu kürzen, indem die gemeinsamen Tails eines balancierten Goals durch rekursive Nichtterminale ersetzt werden.

$$\frac{\begin{array}{c} E_1^1G_1+\cdots+E_n^1G_n \stackrel{?}{=} F_1^1G_1+\cdots+F_n^1G_n \\ \vdots \\ E_1^kG_1+\cdots+E_n^kG_n \stackrel{?}{=} F_1^kG_1+\cdots+F_n^kG_n \\ \vdots \\ E_1G_1+\cdots+E_nG_n \stackrel{?}{=} F_1G_1+\cdots+F_nG_n \end{array}}{E_1V_1+\cdots+E_nV_n \stackrel{?}{=} F_1V_1+\cdots+F_nV_n}\ \text{CUT}$$

Konditionen:

1. $\forall i \in \{1,\ldots,k\} : E_1^iG_1+\cdots+E_n^iG_n \sim_m F_1^iG_1+\cdots+F_n^iG_n$
2. $1 \leq k \leq n$
3. $(V_1,\ldots,V_n)$ ist ein m-Unifier für alle $(E_1^i+\cdots+E_n^i, F_1^i+\cdots+F_n^i)$
4. Zwischen $E_1^kG_1+\cdots+E_n^kG_n \stackrel{?}{=} F_1^kG_1+\cdots+F_n^kG_n$ und $E_1G_1+\cdots+E_nG_n \stackrel{?}{=} F_1G_1+\cdots+F_nG_n$ sind mindestens m Anwendungen von UNF.

Abbildung 5: CUT

Lemma 50 (Wenn $(V_1,\ldots,V_n)$ kanonisch ist, ist CUT complete)**.** Falls $E_1^iG_1+\cdots+E_n^iG_n \sim F_1^iG_1+\cdots+F_n^iG_n$ für alle $1 \leq i \leq k$ und $E_1G_1+\cdots+E_nG_n \sim F_1G_1+\cdots+F_nG_n$ und $(V_1,\ldots,V_n)$ ist kanonisch für die Prämissen, dann gilt $E_1V_1+\cdots+E_nV_n \sim F_iV_1+\cdots+F_nV_n$.

Beweis. Wenn $(V_1,\ldots,V_n)$ kanonisch ist für alle $E_1^iG_1+\cdots+E_n^iG_n \sim F_1^iG_1+\cdots+F_n^iG_n$ $(1 \leq i \leq k)$ und für $E_1G_1+\cdots+E_nG_n \sim F_1G_1+\cdots+F_nG_n$, dann gilt $E_1V_1+\cdots+E_nV_n \sim F_iV_1+\cdots+F_nV_n$ nach Definition 31.1. □

Lemma 51 (CUT ist sound und behält den Kontravalenzindex)**.** Falls $d \geq m$ und $E_1^iG_1+\cdots+E_n^iG_n \sim_d F_1^iG_1+\cdots+F_n^iG_n$ für $1 \leq i \leq k$ und $(V_1,\ldots,V_n)$ ist ein m-Unifier für die Heads $(E_1^i+\cdots+E_n^i, F_1^i+\cdots+F_n^i)$ und sei $m \leq m' \leq d$, sodass $E_1G_1+\cdots+E_nG_n \not\sim_{d+1-m'} F_1G_1+\cdots+F_nG_n$, dann ist $E_1V_1+\cdots+E_nV_n \not\sim_{d+1-m'} F_1V_1+\cdots+F_nV_n$

Beweis. Der Beweis erfolgt weitestgehend analog zu dem Beweis von Lemma 33. Sei $m \leq m' \leq d$ und $(V_1,\ldots,V_n)$ ist ein m-Unifier für die Heads $(E_1^i+\cdots+E_n^i, F_1^i+\cdots+F_n^i)$ und $E_1^iG_1+\cdots+E_n^iG_n \sim_d F_1^iG_1+\cdots+F_n^iG_n$ für $1 \leq i \leq k$. Definiere

$$\begin{array}{ll} E \text{ als } E_1G_1+\cdots+E_nG_n, & F \text{ als } F_1G_1+\cdots+F_nG_n, \\ E' \text{ als } E_1V_1+\cdots+E_nV_n, & F' \text{ als } F_1V_1+\cdots+F_nV_n. \end{array}$$

Angenommen, $E \not\sim_{d+1-m'} F$ aber $E' \sim_{d+1-m'} F'$. Sei $u = a_1\ldots a_l$ das kürzeste Wort, dass zwischen E und F unterscheidet. Offenbar gilt, dass $E\cdot a_1\ldots a_j \not\sim_{d+1-m'-j} F\cdot a_1\ldots a_j$ und $E'\cdot a_1\ldots a_j \sim_{d+1-m'-j} F'\cdot a_1\ldots a_j$. Betrachte im Folgenden wieder die vier Sequenzen:

$$Z_X^s\colon X\cdot a_1\ldots a_s \text{ für } X \in \{E,F,E',F'\}.$$

Sei i (falls es existiert), sodass $Z^i_{E'} = H_1V_1 + \cdots + H_nV_n$ und $Z^{i-1}_{E'} \xrightarrow{a_i} V_j$. Also ist $V_j \stackrel{\text{def}}{=} H_1V_1+\cdots+H_nV_n$, $Z^i_E = G_j$ und $G_j \sim_{d-m'} H_1G_1+\cdots+H_nG_n$ aber $G_j \not\sim_{d+1-m'-i} Z^i_F$ gemäß der Annahme, dass $E \not\sim_{d+1-m'} F$ und damit $H_1G_1 + \cdots + H_nG_n \not\sim_{d+1-m'-i} Z^i_F$. Die Sequenz Z^s_E wird nun aktualisiert zu

$$Z^s_E = (H_1G_1 + \cdots + H_nG_n) \cdot a_{i+1} \cdot a_s$$

für $s > i$ und das Verfahren wiederholt bis $s = l'$. Analog wird die Sequenz Z^i_F mithilfe von $Z^i_{F'}$ aktualisiert. Seien Z^k_E und Z^k_F nun die finalen Sequenzen, dann entweder Z^k_E oder Z^k_F die leere Menge oder ein G_j. Entsprechend ist im ersten Fall aber auch nur eine der korrespondierenden Sequenzen $Z^k_{E'}$ oder $Z^k_{F'}$ die leere Menge. Dies ist allerdings ein Widerspruch in der Annahme, dass E' und F' $(d+1-m')$-äquivalent sind. Im zweiten Fall gilt, dass (o.B.d.A. ist $Z^k_E = G_j$) $Z^k_{E'}$ ein terminierendes Nichtterminal V_i ist. Damit ist auch $Z^k_{F'}$ V_i und somit ist F ebenfalls G_j, ein Widerspruch zu der Annahme, dass $E \not\sim_{d+1-m'} F$. □

Completeness für CUT ist nicht für jede Anwendung garantiert, sondern nur, wenn $(V_1, \ldots, V_n)$ kanonisch ist. Die Tableaukonstruktion muss daher das m, für das der m-Unifier kanonisch ist, nichtdeterministisch raten. Dieses m existiert nach Fakt 35.

Beispiel 52. Seien die Produktionen $A \xrightarrow{a} AA, A \xrightarrow{b} \varepsilon, B \xrightarrow{a} BB, B \xrightarrow{b} \varepsilon$ gegeben, dann sieht das Tableau ohne CUT wie folgt aus (zur Übersicht wird nur der „a-Zweig“ von UNF dargestellt):

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{AA \stackrel{?}{=} BB}\,\text{UNF}}{AAA \stackrel{?}{=} BBB}\,\text{BAL(R)}}{AAA \stackrel{?}{=} BBA}\,\text{UNF}}{AAAA \stackrel{?}{=} BBBA}\,\text{BAL(R)}}{AAAA \stackrel{?}{=} BBAA}\,\text{UNF}}{\vdots}$$

Offenbar wachsen die Konfigurationen weiter und der Zweig ist unendlich lang. Durch die Einführung von CUT sähe derselbe Zweig wie folgt aus:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{AA \stackrel{?}{=} BB}\,\text{UNF}}{AAA \stackrel{?}{=} BBB}\,\text{BAL(R)}}{AAA \stackrel{?}{=} BBA}\,\text{CUT}}{AAV_1 \stackrel{?}{=} BBV_1}\,\text{UNF}}{AAAV_1 \stackrel{?}{=} BBBV_1}\,\text{BAL(R)}}{AAAV_1 \stackrel{?}{=} BBAV_1}\,\text{CUT}}{AAV_1 \stackrel{?}{=} BBV_1}$$

Für das rekursiven Nichtterminal gilt, dass $V_1 \stackrel{\text{def}}{=} V_1$. (V_1) ist ein 0-Unifier der Heads (AA, BB) und wird zwei Mal eingeführt.

5.5 Die Tableaukonstruktion

Die Tableaukonstruktion ist ein zentraler Teil des Beweises, da es nicht ausreicht, in jedem Schritt eine arbiträre Regel anzuwenden. Die Konstruktion muss endlich und berechenbar sein. Bevor konkret auf die Konstruktion eingegangen werden kann, ist es daher sinnvoll, zu erkennen, wann ein Goal erfolgreich oder erfolglos ist, und damit ein Blatt des Tableaus darstellt.

Lemma 53. Ein Goal ist *erfolgreich*, falls es die Form $E \stackrel{?}{=} E$ hat oder eine Wiederholung eines zuvor im Zweig aufgetretenen Goals ist mit mindestens einer Anwendung von UNF dazwischen ($\star$).

Beweis von ($\star$). Angenommen das Goal $E \stackrel{?}{=} F$ ist falsch, dann existiert ein minimales m, für das $E \not\sim_m F$. Wiederholt sich das Goal nach $k \geq 1$ Anwendungen von UNF, gilt nach Lemma 41 $E \not\sim_{m-k} F$. Das führt zu einem Widerspruch in der Annahme, dass m minimal ist. □

Lemma 54. Ein Goal $E \stackrel{?}{=} F$ ist *erfolglos*, falls entweder $E = \emptyset$ oder $F = \emptyset$.

Beweis. Sei $E \stackrel{?}{=} \emptyset$ ein Goal mit $E \neq \emptyset$. $E \not\sim \emptyset$, da $L(\emptyset) = \emptyset$ aber $L(E)$ enthält mindestens ein Wort, da es keine unproduktiven Nichtterminale gibt. Analog für $\emptyset \stackrel{?}{=} F$ und $F \neq \emptyset$. □

Um die Konstruktion des Tableaus zu erläutern und zu zeigen, dass die Konstruktion endlich ist, werden eingangs neue Definitionen eingeführt und nützliche Beobachtungen angestellt. Der Beweis dieser Beobachtungen folgt später.

Definition 55. Sei F eine Konfiguration mit rekursiven Nichtterminalen aus der Familie $(V_1, \ldots, V_n)$, dann bezeichnet $\mathrm{rec}(F) = \max\{|H| \mid V_i \stackrel{\mathrm{def}}{=} H\}$ die Länge der längsten Definition in dieser Familie. Wenn F keine rekursiven Nichtterminale enthält, ist $\mathrm{rec}(F) = 0$.

Definition 56. Sei G eine Grammatik, dann definiere $S = M_G^2 + 4M_G + 1$.

Definition 57. Eine Konfiguration E heißt *klein*, falls $|E| \leq \mathrm{rec}(E) + S$.

Lemma 58.

1. Für jedes $m \geq 0$ ist die Anzahl von Konfigurationen E und F mit $|E|, |F| \leq m$ und E und F haben rekursive Nichtterminale aus $(V_1, \ldots, V_n)$ begrenzt.
2. $|E \cdot a| \leq \max\{\mathrm{rec}(E), |E| + 1\}$ für beliebige $a \in \Sigma$.
3. Wenn $E' \stackrel{?}{=} F'$ das Resultat einer Anwendung der Größe m von BAL ist, dann gilt $|E'|, |F'| \leq 2M + 1 + \max\{m, \mathrm{rec}(E')\}$.

Definition 59. Ein *Block* an balancierten Goals ist eine Folge von Subblocks. Subblocks bestehen aus Anwendungen immer derselben BAL-Regel und – falls diese nicht möglich ist – UNF.

Ein Block wird initiiert durch eine Anwendung von BAL auf die *Root Konfiguration* F der Größe m. Das aus der Anwendung von BAL resultierende Subgoal heißt das *Root Goal* des Blockes. Der Wechsel zwischen Subblocks ist nur beschränkt möglich und wird später näher beschrieben.

Lemma 60. Sei π ein Pfad in einem Block mit der Root Konfiguration F der Größe m, dann gilt:

1. Wenn BAL nicht angewendet werden kann, ist nach maximal $M^2 + M$ Anwendungen von UNF ein Tail exposed.
2. Für jede Konfiguration G, die von einer Applikation von BAL in π verwendet wird, gibt es ein Wort u, sodass $G = F \cdot u$ und für alle Präfixe v von u gilt: $|F \cdot v| > m - (M^2 + 3M)$.
3. Das Ergebnis jeder Anwendung von BAL in π hat die Form $E_1G_1 + \cdots + E_nG_n \stackrel{?}{=} F_1G_1 + \cdots + F_nG_n$ wobei die G_i die Tails von F in S-HNF sind.
4. Wenn die F wahr ist, kann CUT irgendwann mit einer kanonischen Familie angewendet werden.

Auf Basis dieser Beobachtungen, kann letztendlich die letzte wichtige Beobachtung angestellt werden.

Lemma 61. Sei π ein Pfad mit unendlich vielen Anwendungen von CUT, dann enthält π wiederholte Goals.

Die von Stirling beschriebene Strategie zur Konstruktion eines Tableaus kann grob zweigeteilt werden ($E \stackrel{?}{=} F$ ist das aktuelle Goal):

1. Solange E und F klein sind: wende sofern möglich eine BAL-Regel an und andernfalls UNF
2. Wenn eine Applikation von BAL eine große Konfiguration verwendet, baue einen Block auf
   - Wenn der Tail einer BAL-Applikation offengelegt wird (siehe Abbildung 6), kann ein neuer Subblock mit dem Pendant der aktuellen BAL-Regel aufgebaut werden, darf aber kein Goal vor $G_k \stackrel{?}{=} F'$ verwenden.
   - Wenn eine Applikation von BAL innerhalb des Blockes kleiner als m wird...
     - ...und die Konfigurationen wieder klein sind, wird 1 wieder angewendet.
     - ...und die Konfigurationen nicht klein sind, wird ein neuer Block mit kleinerer Root Konfiguration begonnen.
   - Wähle $m' = 0$ und wende CUT mit dem m'-Unifier an. Wenn sich für das daraus gebildete Subgoal ein erfolgloses Tableau ergibt, erhöhe m' um 1 und wiederhole.

$$\begin{array}{rcll} E & \stackrel{?}{=} & F & \\ & \vdots & & \text{BAL(L)} \\ E_1G_1 + \cdots + E_nG_n & \stackrel{?}{=} & F_1G_1 + \cdots + F_nG_n & \\ & \vdots & & \text{Nur UNFs} \\ G_k & \stackrel{?}{=} & F' & \end{array}$$

Abbildung 6: Offenlegung eines Tails in einem BAL(L) Block

Theorem 62. Wenn ein Tableau existiert, das $E \stackrel{?}{=} F$ beweist, dann gilt $E \sim F$.

Beweis. Gilt induktiv, da jede Regel sound ist. □

Theorem 63. Wenn $E \sim F$, dann ist ein Tableau berechenbar, das $E \stackrel{?}{=} F$ auf erfolgreiche Blätter ableitet.

Beweis. Sei $E \sim F$, dann wird entsprechend der oben beschriebenen Prozedur das Tableau angefangen bei $E \stackrel{?}{=} F$ aufgebaut. Eine unendliche Folge von Goals in diesem Tableau beinhaltet entweder unendlich viele Applikationen von CUT und damit nach Lemma 61 erfolgreiche Goals oder unendlich viele kleine Goals mit rekursiven Nichtterminalen derselben Familie (oder keinen rekursiven Nichterminalen) und damit erfolgreiche Goals nach Lemma 58.1. □

Dadurch ist Berechenbarkeit aber noch nicht gegeben, da der Tableau nur semi-entscheidet, ob $E \sim F$ gilt. Wenn das beschriebene Verfahren ein erfolgloses Tableau konstruiert, könnten die m'-Unifier der Anwendungen von CUT „falsch“ gewählt worden sein, sodass sie nicht kanonisch sind. Da CUT nicht für jede Anwendung complete ist (UNF und BAL schon), kann $E \not\sim F$ nur aus einem Tableau geschlossen werden, in dem ein erfolgloses Goal lediglich durch Anwendungen von UNF und BAL von der Wurzel abgeleitet wird.

Theorem 64. Wenn $E \not\sim F$, dann berechnet dasselbe Verfahren ein erfolgloses Tableau, das in einem widerlegenden Zweig nur Anwendungen von UNF und BAL enthält.

Beweis. Das Verfahren wählt immer weiter wachsende m' für die m'-Unifier. Gemäß der Anforderungen für CUT müssen also immer mehr Anwendungen von UNF und BAL vor der Anwendung von CUT kommen. Mit jeder Anwendung von UNF mindert sich der Kontravalenzindex, sodass nach einer begrenzten Anzahl an Applikationen ein verwerfendes Goal erreicht wird ohne, dass CUT angewendet werden konnte. □

Beweis von Lemma 58.

1. Da Γ endlich ist, ist die Anzahl N_E verschiedener einfacher Konfigurationen der Länge $\leq m$ endlich. Wenn die einfachen Konfigurationen auf rekursive Nichtterminale der Familie $(V_1, \ldots, V_n)$ enden, dann ist $k = n$, sonst ist $k = 1$.

$$N_E = \sum_{i=0}^{m} |\Gamma|^i \cdot k = \frac{|\Gamma|^{m+1} - 1}{|\Gamma| - 1} \cdot k$$

Die Anzahl der verschiedenen zulässigen Konfigurationen ist dann nach oben beschränkt durch die Anzahl verschiedener zusammengesetzter Konfigurationen N_Z. Sei K die Menge einfacher Konfigurationen (mit $|K| = N_E$), dann entspricht die Anzahl verschiedener zusammengesetzter Konfigurationen der Anzahl verschiedener Teilmengen von K. Anschaulich wird für jedes Element von K die Entscheidung getroffen, ob es in der Teilmenge enthalten ist oder nicht (2 Möglichkeiten). Damit gibt es also $N_Z = 2^{N_E}$ verschiedene Teilmengen.

2. Sei $E \xrightarrow{a} F$ und

- F ist kein rekursives Nichtterminal. Dann kann per Definition der hier verwendeten KFG eine Transition das aktuelle oberste Stacksymbol durch maximal 2 Symbole ersetzen: $|F| \leq |E| + 1$.
- F ist ein rekursives Nichtterminal. Dann wird F durch seine Definition H ersetzt, die per Definition kleiner oder gleich $\mathrm{rec}(F)$ ist. Offenbar gilt $\mathrm{rec}(F) = \mathrm{rec}(E)$, da E und F rekursive Nichtterminale derselben Familie haben und damit: $|E \cdot a| = |H| \leq \mathrm{rec}(E)$.

3. Der Beweis wird für BAL(L) gezeigt und gilt aufgrund der Symmetrie auch für BAL(R). Sei eine Applikation von BAL(L) gegeben, dann sieht der entsprechende Teil des Tableaus wie folgt aus:

$$\begin{array}{lll} X_1 H_1 & +\cdots+ X_k H_k & \stackrel{?}{=} F \\ & & \vdots \\ E_1(F \cdot w(X_1)) + \cdots + E_k(F \cdot w(X_k)) & & \stackrel{?}{=} F' \end{array}$$

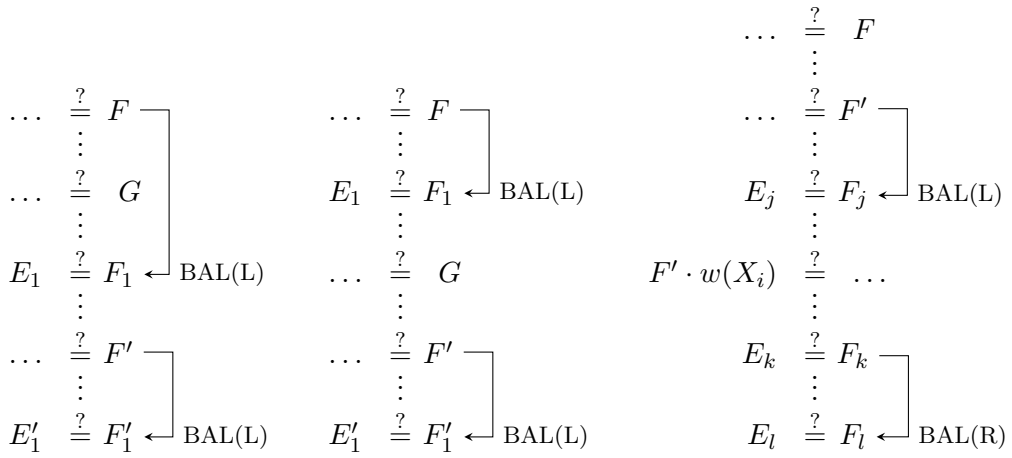
Durch wiederholte Argumentation mit Punkt 2 und, da X_i – und damit auch E_i – keine rekursiven Nichtterminale enthalten, ist $|E_i| \leq M + 1$, da zwischen X_i und E_i maximal M Applikationen von UNF liegen. Die Länge der Tails $F \cdot w(X_i)$ und die Länge von F' kann analog abgeschätzt werden. Für die Gesamtlänge von E' (der linken Seite des Subgoals) kann dann entsprechend abgeschätzt werden mit

$$|E'| \leq \underbrace{M + 1}_{\geq |E_i|} + \underbrace{M + \max\{m, \mathrm{rec}(F)\}}_{\geq |F \cdot w(X_i)|} = 2M + 1 + \max\{|F|, \mathrm{rec}(F)\}.$$

Für F' ergibt sich

$$|F'| \leq M + \max\{\mathrm{rec}(F) - 1, |F|\} \leq 2M + 1 + \max\{|F|, \mathrm{rec}(F)\}.$$

Es gilt, dass $\mathrm{rec}(E') = \mathrm{rec}(F)$, da die Konfigurationen (wenn überhaupt) rekursive Nichtterminale derselben Familie besitzen. □



(a) Fall 1 (b) Fall 2 (c) Wechsel von Subblocks

Abbildung 7: Die Fallunterscheidung für Lemma 60.2

Beweis von Lemma 60.

1. Die Erläuterung erfolgt an BAL(L) und gilt analog für BAL(R). Sei $E_1G_1 + \cdots + E_nG_n = F$ das Ergebnis einer Applikation von BAL(L). Es gilt, dass $E_i \leq M + 1$. Sei $X_1H_1 + \cdots + X_lH_l$ die 1-HNF von E_1. Wenn innerhalb von M Anwendungen von UNF BAL nicht angewendet werden kann, dann muss ein Tail H_i offengelegt worden sein. Dieses Argument kann dann für H_i in 1-HNF wiederholt werden, sodass nach insgesamt $M \cdot (M + 1) = M^2 + M$ Anwendungen ein Tail G_i freigelegt sein muss.

2. Die Aussage gilt für die Root Konfiguration, da $F = F \cdot \varepsilon$ und $|F| = m > m - (M^2 + 3M)$. Für die weiteren Fälle sei im Folgenden von BAL(L) ausgegangen. Aufgrund der Symmetrie von BAL(L) und BAL(R) ist der Beweis aber direkt übertragbar. Sei F eine Konfiguration, die in einer Applikation von BAL verwendet wird. F' folgt aus F durch UNF ($F' = F \cdot u$ mit dem zu UNF zugehörigen Wort u) und ggf. BAL(L) (hier gilt die Aussage induktiv: $F' = F \cdot u$) und $G = F \cdot v$ (für ein beliebiges Präfix v von u) ist eine Konfiguration zwischen F und F'. Angenommen, die Aussage gilt nicht und $|G| \leq m - (M^2 + 3M)$. Dann gilt es, zwei Fälle zu unterscheiden:

Fall 1: G liegt zwischen einer Anwendung von BAL(L) und dessen verwendeten Goal[4]. Da G selbst nicht in der Anwendung von BAL(L) verwendet wird, liegen zwischen G und dem Goal, auf das BAL(L) angewendet wird, maximal $M - 1$ Anwendungen von UNF (Abbildung 7a). Da $|G| \leq m - (M^2 + 3M)$ folgt aus Lemma 58.2, dass $|F_1| \leq m - (M^2 + 2M + 1)$. Da $|F'| \geq m$ aufgrund der Konstruktion des Tableaus, liegen zwischen F_1 und F' also mindestens $M^2 + 2M + 1$ Anwendungen von ausschließlich UNF, die die Konfiguration vergrößern. Das ist aber ein Widerspruch, da nach spätestens $M^2 + M$ Anwendungen von UNF ein Tail exposed sein muss, da BAL(L) zwischen F_1 und F' nicht angewendet werden konnte und BAL(R) somit möglich gewesen sein musste.

Fall 2: G folgt nach einer Anwendung von BAL(L) oder ist das Resultat einer Anwendung von BAL(L) (Abbildung 7b). Laut Annahme gilt $|G| \leq m - (M^2 + 3M)$ und aufgrund der Konstruktion ist $|F'| \geq m$. Es muss also gelten, dass zwischen F' und F' mindestens $M^2 + 3M$ Anwendungen von UNF liegen, die die Konfigurationen vergrößern. Aus demselben Grund wie im vorherigen Fall führt das zu einem Widerspruch.

Es kann also nicht gelten, dass $|G| \leq m - (M^2 + 3M)$, da dies in jedem Fall zu einem Widerspruch führt. Zuletzt gilt es zu zeigen, dass die Aussage weiterhin gilt,

[4]O.B.d.A. liegt zwischen G und dem verwendeten Goal kein weiteres für BAL verwendetes Goal.

wenn der Subblock gewechselt wird (Abbildung 7c). Gemäß der Annahme gibt es ein Wort u, sodass $F' = F \cdot u$. Der offengelegte Tail $F' \cdot w(X_i)$ kann also geschrieben werden als $F \cdot uw(X_i)$. Da zwischen E_k und dem Tail nur UNF liegt, gibt es somit ein Wort v, sodass $E_k = F \cdot uw(X_i)v$ und aufgrund der Konstruktion des Tableaus gilt, dass $|F'|, |E_k| \geq m$. Sei v' ein beliebiges Präfix von $w(X_i)v$ und angenommen $|F' \cdot v'| \leq m - (M^2 + 3M)$. Wenn v' ein Präfix von $w(X_i)$ ist und somit $|v'| \leq M$, ist $|F' \cdot v'| \leq m - (M^2 + 2M)$ und somit müssen zwischen $F' \cdot w(X_i)$ und E_k mindestens $M^2 + 2M$ Anwendungen von UNF liegen, die die Konfigurationen vergrößern, ohne, dass BAL möglich ist. Wenn v' kein Präfix von $w(X_i)$ ist, dann müssen zwischen $F' \cdot w(X_i)$ und E_k mindestens $M^2 + 3M$ derartige Anwendungen von UNF liegen, ohne, dass BAL möglich ist. Dies ist aber wieder ein Widerspruch.

3. Sei $\beta_1 G_1 + \cdots + \beta_n G_n$ die S-HNF von F und sei $\beta_i = X_1^i \dots X_{|\beta_i|}^i$, dann bezeichnet $\beta_i^{(j)}$ das Suffix $X_j^i \dots X_{|\beta_i|}^i$ von β_i. Falls $j > |\beta_i|$, dann ist $\beta_i^{(j)} = \varepsilon$.
Sei G eine von BAL verwendete Konfiguration in π. Nach Lemma 60.2 ist $G = F \cdot u$ für ein u und für alle Präfixe v von u ist $|F \cdot v| > m - (M^2 + 3M)$. Entsprechend kann G in der Form $E_1 \beta_1^{(b_1)} G_1 + \cdots + E_n \beta_n^{(b_n)} G_n$ dargestellt werden mit $b_i = M^2 + 3M$. Das resultierende Subgoal von BAL (hier an BAL(L) gezeigt) ist

$$E_1'(G \cdot w(X_1)) + \cdots + E_k'(G \cdot w(X_k)) \stackrel{?}{=} G \cdot w$$

für ein Wort w mit $|w| \leq M$. Nach Lemma 28 gilt damit auch:

$$\begin{aligned} G \cdot w(X_i) &= (E_1 \beta_1^{(b_1)} \cdot w(X_i)) G_1 + \cdots + (E_n \beta_n^{(b_n)} \cdot w(X_i)) G_n \\ G \cdot w \quad\;\; &= (E_1 \beta_1^{(b_1)} \cdot w \quad\;\;) G_1 + \cdots + (E_n \beta_n^{(b_n)} \cdot w \quad\;\;) G_n \end{aligned}$$

Durch Einsetzen dieser Formen in das resultierende Subgoal, nimmt dieses die geforderte balancierte Form mit den Tails $G_1, \dots, G_n$ an.

4. Solange ein Block nicht klein wird, kann BAL wiederholt angewendet werden. Sei $X_1 E_1 + \cdots + X_k E_k \stackrel{?}{=} Y_1 F_1 + \cdots + Y_k F_k$ ein Goal in 1-HNF. Nach maximal M Anwendungen von UNF wurde auf jeder Seite ein Tail freigelegt. Wenn nicht, dann kann BAL angewendet werden. Falls BAL nicht angewendet werden kann, da die Tails immer freigelegt werden, dann werden die Konfigurationen klein und werden entweder erfolglos, wiederholen sich oder sind trivial wahr.
Sei $\beta_1 G_1 + \cdots + \beta_n G_n$ die S-HNF von F und $E_1^1 G_1 + \cdots + E_n^1 G_n \stackrel{?}{=} F_1^1 G_1 + \cdots + F_n^1 G_n$ ist das Root Goal von dem Block, in dem BAL beliebig häufig angewendet werden kann. Nach Fakt 35 existiert ein minimales m' sodass der m'-Unifier $(V_1, \dots, V_n)$ mit maximal n Verfeinerungen kanonisch ist. Lemma 60.3 garantiert, dass nach $k \leq n$ Anwendungen von BAL, mindestens m' weiteren Anwendungen von UNF und einer weiteren Anwendung von BAL CUT angewendet werden kann. Wenn nicht genügend Anwendungen von BAL auftreten, dann wurden (wie oben beschrieben) die Goals vorher schon klein, sodass CUT nicht notwendig war. □

Beweis von Lemma 61. Ein unendlicher Pfad enthält unendlich viele Root Konfigurationen mit denselben Köpfen, da die Größe der Köpfe nach oben durch $S + 2M + 1$ (Lemma 58.3) beschränkt ist. Seien $E_1 G_1^i + \cdots + E_n G_n^i \stackrel{?}{=} F_1 G_1^i + \cdots + F_n G_n^i$ und $E_1 G_1^j + \cdots + E_n G_n^j \stackrel{?}{=} F_1 G_1^j + \cdots + F_n G_n^j$ zwei Root Goals mit denselben Köpfen, dann ist die Familie rekursiver Nichtterminale, die für $E_1 G_1^i + \cdots + E_n G_n^i \stackrel{?}{=} F_1 G_1^i + \cdots + F_n G_n^i$ kanonisch ist, nach Fakt 34 auch für $E_1 G_1^j + \cdots + E_n G_n^j \stackrel{?}{=} F_1 G_1^j + \cdots + F_n G_n^j$ kanonisch. Die Anwendung von CUT in den Blöcken dieser Root Goals resultiert also beide Male in demselben Goal $E_1 V_1 + \cdots + E_n V_n \stackrel{?}{=} F_1 V_1 + \cdots + F_n V_n$. Also ist das Goal eine Wiederholung und damit erfolgreich. □

6 Beispiele

Zuletzt soll der komplette Entscheidungsprozess exemplarisch an den drei Automaten M_1, M_2 und M_3 mit $L(M_1) = L(M_2) \neq L(M_3)$ vorgeführt werden, indem geprüft wird, ob M_1 und M_2 beziehungsweise M_1 und M_3 äquivalent sind. Das Alphabet, das alle drei Automaten verwenden, ist $\Sigma = \{a\}$. Die Automaten sind dann definiert wie folgt: $M_1 = (\{z_0, z_1\}, \Sigma, \{\#\}, \delta_1, z_0, \#, \{z_0\})$ mit

$$\delta_1\colon \quad z_0a\#\to z_1\# \qquad z_1a\#\to z_0\#$$

$M_2 = (\{z_0, z_1, z_2\}, \Sigma, \{\#\}, \delta_2, z_0, \#, \{z_0, z_2\})$ mit

$$\delta_2\colon \quad z_0a\#\to z_1\# \qquad z_1a\#\to z_2\# \qquad z_2a\#\to z_1\#$$

und $M_3 = (\{z_0, z_1, z_2\}, \Sigma, \{\#\}, \delta_3, z_0, \#, \{z_2\})$ mit

$$\delta_3\colon \quad z_0a\#\to z_1\# \qquad z_1a\#\to z_2\# \qquad z_2a\#\to z_1\#$$

Offenbar ist $L(M_1) = L(M_2) = \{a^{2n} \mid n \in \mathbb{N}_0\}$ verschieden zu $L(M_3) = \{a^{2n} \mid n \in \mathbb{N}^+\}$.

Schritt 1: Übersetzung in DKA nach Stirling Damit die gegebenen DKA, die mithilfe von Endzuständen akzeptieren, in strikt deterministische KFG übersetzt werden können, müssen sie zuerst in DKA mit Akzeptanz durch leeren Stack übersetzt werden. Das erweiterte Alphabet ist für alle Grammatiken wieder gleich: $\Sigma' = \{a, \$\}$. Die übersetzte Variante von M_1 ist $M_1' = (\{z_0, z_1, z_e\}, \Sigma', \{\#, \square\}, \delta_1', z_0\#\square)$ mit den Transitionen

$$\begin{aligned}\delta_1'\colon \quad & z_0\#\xrightarrow{a} z_1\# \quad z_1\#\xrightarrow{a} z_0\# \quad z_0\#\xrightarrow{\$} z_e\varepsilon \\ & z_0\square\xrightarrow{\$} z_e\varepsilon \quad z_e\#\xrightarrow{\varepsilon} z_e\varepsilon \quad z_e\square\xrightarrow{\varepsilon} z_e\varepsilon\end{aligned}$$

Der übersetzte Automat M_2 ist $M_2' = (\{z_0, z_1, z_2, z_e\}, \Sigma', \{\#, \square\}, \delta_2', z_0\#\square)$ mit den Transitionen

$$\begin{aligned}\delta_2'\colon \quad & z_0\#\xrightarrow{a} z_1\# \quad z_1\#\xrightarrow{a} z_2\# \quad z_2\#\xrightarrow{a} z_1\# \\ & z_0\#\xrightarrow{\$} z_e\varepsilon \quad z_0\square\xrightarrow{\$} z_e\varepsilon \quad z_2\#\xrightarrow{\$} z_e\varepsilon \\ & z_2\square\xrightarrow{\$} z_e\varepsilon \quad z_e\#\xrightarrow{\varepsilon} z_e\varepsilon \quad z_e\square\xrightarrow{\varepsilon} z_e\varepsilon\end{aligned}$$

Und zuletzt wird M_3 übersetzt zu $M_3' = (\{z_0, z_1, z_2, z_e\}, \Sigma', \{\#, \square\}, \delta_3', z_0\#\square)$ mit

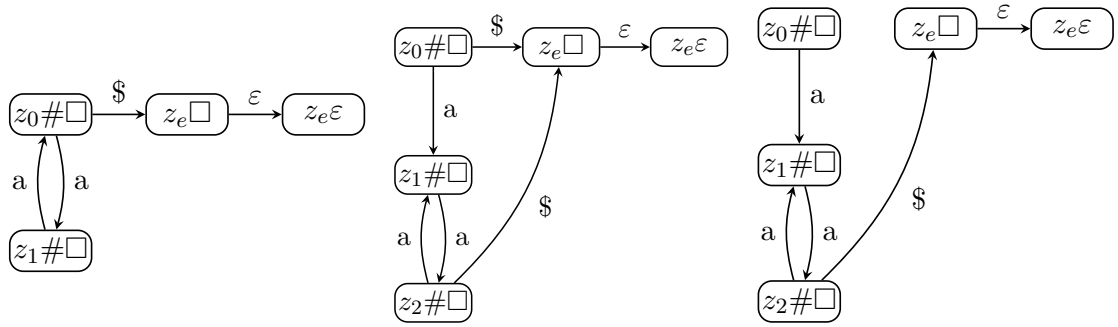
$$\begin{aligned}\delta_3'\colon \quad & z_0\#\xrightarrow{a} z_1\# \quad z_1\#\xrightarrow{a} z_2\# \quad z_2\#\xrightarrow{a} z_1\# \\ & z_2\#\xrightarrow{\$} z_e\varepsilon \quad z_2\square\xrightarrow{\$} z_e\varepsilon \quad z_e\#\xrightarrow{\varepsilon} z_e\varepsilon \\ & z_e\square\xrightarrow{\varepsilon} z_e\varepsilon\end{aligned}$$

Die möglichen Konfiguration und Transitionen der einzelnen Automaten (angefangen bei der jeweiligen Startkonfiguration) sind in Abbildung 8 als Graphen visualisiert.

Schritt 2: Übersetzung in strikt deterministische KFG in 3-GNF Im nächsten Schritt werden die Automaten in entsprechende deterministische KFG in 3-GNF übersetzt. Zuerst wird dafür die übersetzte Produktionsmenge angegeben mit ausgegrauten redundanten Nichtterminalen und anschließend die Produktionsmenge ohne redundante Nichtterminale. Die Nichtterminale werden in der tatsächlichen Produktionsmenge zudem umbenannt, um Lesbarkeit zu gewährleisten.

Zuerst wird wieder M_1' nach $G_1 = (V_1, \Sigma', P_1)$ übersetzt. Die ε-Nichtterminale sind $[z_e\#z_e]$ und $[z_e\square z_e]$ und die Übersetzung der Transitionen resultiert in:

$$\begin{aligned}& [z_0\#z_e] \xrightarrow{\$} \varepsilon && [z_0\square z_e] \xrightarrow{\$} \varepsilon && \\ & [z_0\#z_0] \xrightarrow{a} [z_1\#z_0] && [z_0\#z_1] \xrightarrow{a} [z_1\#z_1] && [z_0\#z_e] \xrightarrow{a} [z_1\#z_e] \\ & [z_1\#z_0] \xrightarrow{a} [z_0\#z_0] && [z_1\#z_1] \xrightarrow{a} [z_0\#z_1] && [z_1\#z_e] \xrightarrow{a} [z_0\#z_e]\end{aligned}$$



(a) Transitionen von M_1' (b) Transitionen von M_2' (c) Transitionen von M_3'

Abbildung 8: Die Transitionen der einzelnen Automaten als Graphen dargestellt

Die ausgegrauten Nichtterminale sind redundant. Entfernt man die entsprechenden Transitionen und benennt die Nichtterminale um, so bekommt man die Produktionen

$$P_1 = \{A \xrightarrow{\$} \varepsilon,\ B \xrightarrow{\$} \varepsilon,\ A \xrightarrow{a} C,\ C \xrightarrow{a} A\}$$

und die entsprechenden Variablen $V_1 = \{A, B, C\}$. Die Startkonfiguration berechnet sich als

$$\begin{aligned} S_1 = \ & \cancel{[z_0\#z_0]\,[z_0\square z_0]} + \cancel{[z_0\#z_0]\,[z_0\square z_1]} + \cancel{[z_0\#z_0]\,[z_0\square z_e]} + \\ & \cancel{[z_0\#z_1]\,[z_1\square z_0]} + \cancel{[z_0\#z_1]\,[z_1\square z_1]} + \cancel{[z_0\#z_1]\,[z_1\square z_e]} + \\ & \cancel{[z_0\#z_e]\,[z_e\square z_0]} + \cancel{[z_0\#z_e]\,[z_e\square z_1]} + [z_0\#z_e]\,[z_e\square z_e] \end{aligned}$$

Die durchgestrichenen einfachen Konfigurationen enthalten redundante Nichtterminale und werden daher entfernt. Übrig bleibt $S_1 = [z_0\#z_e]\,[z_e\square z_e] = [z_0\#z_e] = A$. $[z_e\square z_e]$ wird dabei entfernt, da es ein ε-Nichtterminal ist.

Die Übersetzung von M_2' nach $G_2 = (V_2, \Sigma', P_2)$ resultiert in den Transitionen

$[z_0\#z_e] \xrightarrow{\$} \varepsilon$	$[z_0\square z_e] \xrightarrow{\$} \varepsilon$	$[z_2\#z_e] \xrightarrow{\$} \varepsilon$	$[z_2\square z_e] \xrightarrow{\$} \varepsilon$
$[z_0\#z_0] \xrightarrow{a} [z_1\#z_0]$	$[z_0\#z_1] \xrightarrow{a} [z_1\#z_1]$	$[z_0\#z_2] \xrightarrow{a} [z_1\#z_2]$	$[z_0\#z_e] \xrightarrow{a} [z_1\#z_e]$
$[z_1\#z_0] \xrightarrow{a} [z_2\#z_0]$	$[z_1\#z_1] \xrightarrow{a} [z_2\#z_1]$	$[z_1\#z_2] \xrightarrow{a} [z_2\#z_2]$	$[z_1\#z_e] \xrightarrow{a} [z_2\#z_e]$
$[z_2\#z_0] \xrightarrow{a} [z_1\#z_0]$	$[z_2\#z_1] \xrightarrow{a} [z_1\#z_1]$	$[z_2\#z_2] \xrightarrow{a} [z_1\#z_2]$	$[z_2\#z_e] \xrightarrow{a} [z_1\#z_e]$

Für die Produktionsmenge P_2 werden wieder Transitionen mit redundanten Nichtterminalen entfernt und die Nichtterminale werden umbenannt:

$$\begin{aligned} P_2 = \{ & A \xrightarrow{\$} \varepsilon,\ \ B \xrightarrow{\$} \varepsilon,\ \ C \xrightarrow{\$} \varepsilon,\ D \xrightarrow{\$} \varepsilon,\ \} \\ & A \xrightarrow{a} E,\ E \xrightarrow{a} C,\ C \xrightarrow{a} E \end{aligned}$$

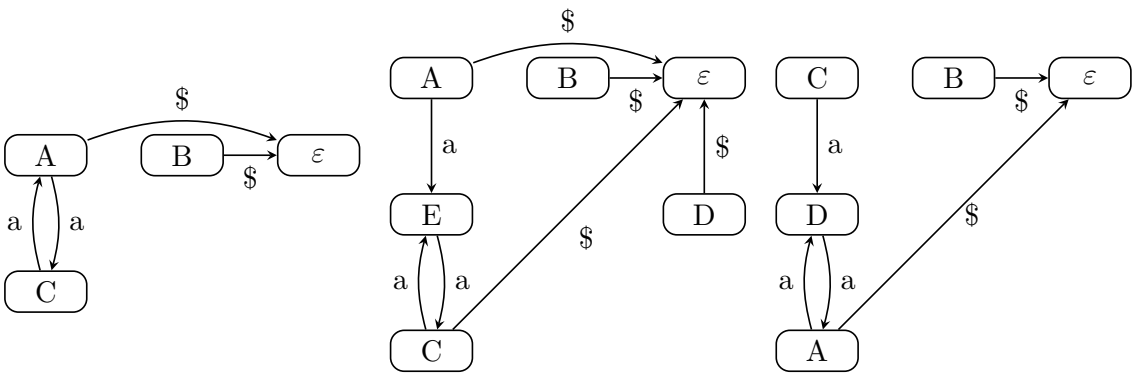
Es ergibt sich, dass $V_2 = \{A, B, C, D, E\}$ und $S_2 = [z_0\#z_e]\cancel{[z_e\square z_e]} = A$. Zuletzt werden die Transitionen von M_3' übersetzt:

$[z_2\#z_e] \xrightarrow{\$} \varepsilon$	$[z_2\square z_e] \xrightarrow{\$} \varepsilon$		
$[z_0\#z_0] \xrightarrow{a} [z_1\#z_0]$	$[z_0\#z_1] \xrightarrow{a} [z_1\#z_1]$	$[z_0\#z_2] \xrightarrow{a} [z_1\#z_2]$	$[z_0\#z_e] \xrightarrow{a} [z_1\#z_e]$
$[z_1\#z_0] \xrightarrow{a} [z_2\#z_0]$	$[z_1\#z_1] \xrightarrow{a} [z_2\#z_1]$	$[z_1\#z_2] \xrightarrow{a} [z_2\#z_2]$	$[z_1\#z_e] \xrightarrow{a} [z_2\#z_e]$
$[z_2\#z_0] \xrightarrow{a} [z_1\#z_0]$	$[z_2\#z_1] \xrightarrow{a} [z_1\#z_1]$	$[z_2\#z_2] \xrightarrow{a} [z_1\#z_2]$	$[z_2\#z_e] \xrightarrow{a} [z_1\#z_e]$

$$P_3 = \{A \xrightarrow{\$} \varepsilon,\ B \xrightarrow{\$} \varepsilon,\ C \xrightarrow{a} D,\ D \xrightarrow{a} A,\ A \xrightarrow{a} D\ \}$$

Die produktiven Nichtterminale sind $V_3 = \{A, B, C, D\}$ und die Startkonfiguration ist $S_3 = [z_0\#z_e]\cancel{[z_e\square z_e]} = C$.

Alle möglichen Transitionen zwischen den Konfigurationen, die von der jeweiligen Startkonfiguration aus erreichbar sind, sind in Abbildung 9 wieder als Graphen dargestellt. Die Graphen sind für diese Kellerautomaten und Grammatiken endlich, da der Keller durch keine Transition weiter wächst.



(a) Transitionen von G_1 (b) Transitionen von G_2 (c) Transitionen von G_3

Abbildung 9: Die möglichen Transitionen zwischen den einzelnen Nichtterminalen

Schritt 3: Vereinigung Im Folgenden sind die Gramatiken $G_{1,2}$ und $G_{1,3}$ die zusammengeführten Grammatiken von G_1 und G_2 beziehungsweise G_1 und G_3.

$G_{1,2} = (\{\underbrace{A, B, C}_{\text{aus } G_1}, \underbrace{V, W, X, Y, Z}_{\text{aus } G_2}\}, \Sigma', P_{1,2})$ mit

$P_{1,2} = \{A \xrightarrow{\$} \varepsilon, \quad B \xrightarrow{\$} \varepsilon, \quad A \xrightarrow{a} C, \quad C \xrightarrow{a} A, \quad$ und $S_{1,2} = A, S'_{1,2} = V$
$V \xrightarrow{\$} \varepsilon, \quad W \xrightarrow{\$} \varepsilon, \quad X \xrightarrow{\$} \varepsilon, \quad Y \xrightarrow{\$} \varepsilon,$
$V \xrightarrow{a} Z, \quad Z \xrightarrow{a} X, \quad X \xrightarrow{a} Z\}$

$G_{1,3} = (\{\underbrace{A, B, C}_{\text{aus } G_1}, \underbrace{W, X, Y, Z}_{\text{aus } G_3}\}, \Sigma', P_{1,3})$ mit

$P_{1,3} = \{A \xrightarrow{\$} \varepsilon, \quad B \xrightarrow{\$} \varepsilon, \quad A \xrightarrow{a} C, \quad C \xrightarrow{a} A, \quad$ und $S_{1,3} = A, S'_{1,3} = Y$
$W \xrightarrow{\$} \varepsilon, \quad X \xrightarrow{\$} \varepsilon, \quad Y \xrightarrow{a} Z \quad Z \xrightarrow{a} W$
$W \xrightarrow{a} Z\}$

Schritt 4: Tableaukonstruktion Für die beiden Grammatiken $G_{1,2}$ und $G_{1,3}$ wird zuletzt getestet, ob $S_{1,2} \sim S'_{1,2}$ beziehungsweise $S_{1,3} \sim S'_{1,3}$ gelten, indem Tableaux angefangen bei $S_{1,2} = S'_{1,2}$ und $S_{1,3} = S'_{1,3}$ konstruiert werden. Bei Anwendungen von UNF ist der linke Zweig die Anwendung von \$ und der rechte Zweig die Anwendung von a.

Das Tableau für $S_{1,2} = S'_{1,2}$ sieht aus wie folgt:

$$\dfrac{\dfrac{A \stackrel{?}{=} V}{\varepsilon \stackrel{?}{=} \varepsilon \quad C \stackrel{?}{=} Z}\ \text{UNF}}{C \stackrel{?}{=} Z}\ \text{BAL(R)}$$

Also gilt, dass $S_{1,2} \sim S'_{1,2}$ und damit sind die Automaten M_1 und M_2 äquivalent, da das Tableau nur erfolgreiche Blätter hat.

Das Tableau für $S_{1,3} = S'_{1,3}$ sieht aus wie folgt:

$$\dfrac{\dfrac{A \stackrel{?}{=} Y}{\varepsilon \stackrel{?}{=} \emptyset \quad C \stackrel{?}{=} Z}\ \text{UNF}}{\emptyset \stackrel{?}{=} Z}\ \text{BAL(L)}$$

Damit gilt, dass $S_{1,3} \not\sim S'_{1,3}$ und damit sind die Automaten M_1 und M_3 nicht äquivalent, da das Tableau mindestens ein erfolgloses Blatt hat.

7 Eine Alternative

In [Sti02] zeigt Stirling, dass CUT durch eine genauere Analyse der Goals komplett ersetzt werden kann, sodass für die Tableautiefe eine obere Schranke berechnet werden kann. Während der Beweis zu Umfangreich ist, um ihn hier anzuführen, seien die Änderungen, die sich für den Tableau ergeben, einmal umrissen.

Definition 65. Sei $E = E_1G_1 + \cdots + E_nG_n$. $F = F_1H_1 + \cdots + F_mH_m$ *erweitert* E *um* die *Extension* $e = (K_1^1 + \cdots + K_n^1, \ldots, K_1^m + \cdots + K_n^m)$, falls für jedes $1 \leq i \leq m$ gilt, dass $H_i = K_1^iG_i + \cdots + K_n^i$.

Definition 66 ([Sti02, S. 28])**.** Sei $d(0), \ldots, d(l)$ ein Zweig von Goals im Tableau. Das Goal $d(l)\colon E_1H_1 + \cdots + E_nH_n \stackrel{?}{=} F_1H_1 + \cdots + F_nH_n$ *gehorcht dem Extension Theorem* falls gilt:

1. Es gibt $g(i), h(i) \in \{d(0), \ldots, d(l)\}$ für $1 \leq i \leq 2^n$ mit
$$g(i)\colon E_1G_1^i + \cdots + E_nG_n^i \stackrel{?}{=} F_1G_1^i + \cdots + F_nG_n^i$$
$$h(i)\colon E_1H_1^i + \cdots + E_nH_n^i \stackrel{?}{=} F_1H_1^i + \cdots + F_nH_n^i$$
2. $d(l) = h(2^n)$ und zwischen $h(2^n - 1)$ und $d(l)$ liegt mindestens eine Anwendung von UNF
3. Es gibt Extensions $e_1, \ldots, e_n$, sodass für jedes e_j und $i \geq 0$
$$g(2^j i + 2^{j-1} + 1) \text{ erweitert } g(2^j j + 2^{j-1}) \text{ um } e_j$$
$$h(2^j i + 2^{j-1} + 1) \text{ erweitert } h(2^j j + 2^{j-1}) \text{ um } e_j$$

Gemäß dem Extension Theorem ist $d(l)$ äquivalent falls alle $g(i)$ und alle $h(j)$ mit $j < 2^n$ äquivalent sind. Analog zum Beweis von Lemma 53 kann gezeigt werden, dass $d(l)$ somit erfolgreich ist, falls es dem Extension Theorem gehorcht. Tatsächlich ist Lemma 53 ein Sonderfall des Extension Theorems mit $n = 1$.

Durch diese aufwendigere Analyse wird CUT nicht mehr benötigt und damit entfallen auch rekursive Nichtterminale und entsprechend kanonische Familien beziehungsweise m-Unifier. Zudem wird die Konstruktion des Tableaus wesentlich vereinfacht und besteht nur noch aus der Konstruktion von Blöcken. Es ist nicht mehr notwendig, die Konfigurationen klein zu halten, da es keinen Mechanismus gibt, sie zu kürzen, wenn sie erst einmal groß werden.

Stirling war dadurch in der Lage, eine Abschätzung für die maximale Tableautiefe in Abhängigkeit von der Grammatik und den zu prüfenden Konfigurationen zu berechnen. Ein Teil dieser Abschätzung $f(n)$ ist die Abschätzung, innerhalb wievieler Goals das Extension Theorem greift, $f_2[d, h, w](w)$ (vgl. [Sti02, S. 35]). Stirling definiert diese Methode rekursiv:

$$f_2[d, h, w](0) = \text{goal}(h)$$
$$f_2[d, h, w](j + 1) = \text{ext}(f_2[d, h, w](j) \cdot d, w) \cdot f_2[d, h, w](j)$$

Mit $d = M^2 + 2M, h = M^2 + 5M + 2, w = \text{width}(h)$. $\text{goal}(h)$ bezeichnet dabei die Anzahl verschiedener Goals $E \stackrel{?}{=} F$ mit $|E|, |F| \leq h$, $\text{width}(h)$ ist die maximale *Breite* n, für die eine zulässige Konfiguration $\beta_1 + \cdots + \beta_n$ existiert mit $|\beta_i| \leq h$ und $\text{ext}(d, w)$ ist die Anzahl verschiedener Extensions e die nicht länger als d sind und eine maximale Breite von w haben.

Die Tatsache, dass für zwei Konfigurationen E und F ein Tableau existiert, dessen Tiefe durch eine Funktion $f(n)$ abgeschätzt werden kann, UNF den Kontravalenzindex reduziert und BAL den Kontravalenzindex behält, führt zu einem interessanten Korollar.

Korollar 67 ([Sti02, S. 30])**.** Für $|E|, |F| < n$ gilt $E \sim F \iff E \sim_{f(n)} F$

Die Abschätzung $f(n)$ ist zwar für einfache Automaten schon riesig, das Korollar zeigt allerdings, dass UNF ausreicht um Äquivalenz zu entscheiden.

8 Schlusswort

Es wurde gezeigt, dass die Äquivalenz von deterministischen Kellerautomaten mit Akzeptanz durch Endzustände, entscheidbar ist, indem das Problem auf die Entscheidbarkeit der Äquivalenz der Sprachen von zwei Startvariablen derselben strikt deterministischen kontextfreien Grammatik reduziert wurde. Zuletzt wurde Stirlings neueres Resultat kurz präsentiert, mit dem die Tableaukonstruktion vereinfacht und CUT durch genauere Analyse des Tableaus ersetzt werden kann. Während für die im Detail präsentierte Variante mit CUT aktuell keine Schranke für die Laufzeit angegeben werden kann, da keine obere Schranke für ein m bekannt ist, sodass der m-Unifier kanonisch ist, erlaubt die Alternative, eine obere Schranke zu bestimmen.

Literatur

[HH73] Michael A. Harrison and Ivan M. Havel. Strict deterministic grammars. *J. Comput. Syst. Sci.*, 7(3):237–277, June 1973.

[Sti01] Colin Stirling. Decidability of dpda equivalence. *Theoretical Computer Science*, 255(1):1 – 31, 2001.

[Sti02] Colin Stirling. Deciding dpda equivalence is primitive recursive. In Peter Widmayer, Stephan Eidenbenz, Francisco Triguero, Rafael Morales, Ricardo Conejo, and Matthew Hennessy, editors, *Automata, Languages and Programming*, pages 821–832, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg.

[Vol19] Heribert Vollmer. Skript zur Vorlesung: Grundlagen der Theoretischen Informatik. unveröffentlicht, WiSe 2018/19.

Erklärung der Selbstständigkeit

Hiermit versichere ich, die vorliegende Bachelorarbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet zu haben. Die Arbeit hat in gleicher oder ähnlicher Form noch keinem anderen Prüfungsamt vorgelegen.

Hannover, 26. August 2020	
(Ort, Datum)	(Unterschrift)