GERT SMOLKA

EINIGE ERGEBNISSE ZUR VOLLSTÄNDIGKEIT DER BEWEISPROZEDUR VON KOWALSKI

DIPLOMARBEIT BETREUT VON P. DEUSSEN UND J.H. SIEKMANN
FAKULTÄT FÜR INFORMATIK, UNIVERSITÄT KARLSRUHE, JULI 1982

I N H A L T

Binet lächelte mit gesenktem Kopf und geweiteten Nasenflügeln und schien versunken in einen jener vollkommenen Glückszustände, die zweifellos nur anspruchslose Beschäftigungen vermitteln können, die den Geist durch leicht zu überwindende Schwierigkeiten anregen und ihn völlig befriedigen mit einem Resultat, das keine weiteren Wünsche mehr schafft.

Aus Gustave FLAUBERTs
"Madame Bovary"

E I N L E I T U N G

Das automatische Beweisen mathematischer Sätze durch einen Computer wird seit etwa zwanzig Jahren intensiv erforscht und ist heute eine eigene Diziplin auf dem Gebiet der Künstlichen Intelligenz. Zwar sind seit HERBRAND /He29/ Algorithmen bekannt, die zu jedem gültigen Satz einer formalisierbaren Theorie einen Beweis berechnen, aber diese Beweisprozeduren benötigen für interessante Theoreme so ungeheuer viele Rechenschritte, daß sie auch auf zukünftigen Rechenanlagen nicht realisierbar sein werden. Nach den ersten enttäuschenden Erfahrungen mit diesen HERBRAND-Prozeduren wurde die Entwicklung ab 1965 durch den von J.A. ROBINSON entwickelten Resolutionskalkül /Ro65/ dominiert. Dieser Kalkül ist einer der ersten Logikkalküle, die speziell für die Implementierung auf einem Rechner entworfen wurden. In der folgenden Dekade wurden Dutzende von Verfeinerungen für den Resolutionskalkül entwickelt und implementiert. Da keiner dieser Versuche zu einem Beweissystem führte, das die euphorischen Erwartungen der frühen sechziger Jahre erfüllte, setzte sich die Erkenntnis durch, daß ein leistungsfähiges Beweisprogramm neben der Deduktionskomponente, die den Logikkalkül implementiert, über eine Wissenskomponente verfügen muß, die inhaltliches Wissen und spezifische Beweismethoden über das mathematische Gebiet einbringt, aus dem das Theorem stammt. Während für die Deduktionskomponente viele Konzepte und Ergebnisse vorliegen (Resolution, Paramodulation, Matrixkalküle usw.), findet man für die Wissenskomponente eines Automatischen Beweisers jedoch bis heute vor allem Absichtserklärungen.

Die Connection Graph Proof Procedure, die der Deduktionskomponente des Karlsruher Beweissystems /BEHSSW81/ zugrunde liegt, wurde 1975 von Robert KOWALSKI publiziert /Ko75/. Diese Beweisprozedur unterscheidet sich von den anderen Verfeinerungen des Resolutionskalküls besonders dadurch, daß mit dem Fortschreiten des Ableitungsprozesses viele der vorher erzeugten Klauseln wieder gelöscht werden können. Trotz der Versuche von BROWN /Br76/, BIBEL /Bi81a, Bi81b/ und von SIEKMANN und STEPHAN /SS76, SS80/ ist das Vollständigkeitsproblem der Con-

nection Graph Proof Procedure bisher nicht gelöst: Es ist offen, ob für den zugrunde liegenden "KOWALSKI-Kalkül" eine Suchstrategie existiert, die für jede unerfüllbare Klauselmenge mit der Erzeugung der leeren Klausel terminiert. Die vorliegende Arbeit untersucht dieses Problem und liefert die folgenden Lösungsbeiträge:

(1) Für die Klasse der unär widerlegbaren Klauselmengen, die im Automatischen Beweisen von großer Bedeutung ist und die die HORN-Mengen mit einschließt, konnten wir alle offenen Fragen klären. Unter anderem geben wir ein Kriterium für die Vollständigkeit von Strategien bei unär widerlegbaren Klauselmengen an. Dieses Kriterium liefert insbesondere, daß jede unär erschöpfende (d.h., jeder Link, der eine unäre Resolution repräsentiert, wird nach endlich vielen Schritten wieder gelöscht) Strategie für unär widerlegbare Klauselmengen vollständig ist. Dies ist auch dann der Fall, wenn die Tautologie-Eliminationsregel uneingeschränkt und die Subsumtions-Eliminationsregel mit der Einschränkung aus /Bi81c/ angewendet werden. Wir zeigen die obigen Ergebnisse mit einer neuen, transparenten Beweistechnik, die sich voraussichtlich so verallgemeinern läßt, daß mit ihr auch das Vollständigkeitsproblem für allgemeine Klauselmengen gelöst werden kann.

(2) Für die Tautologieelimination geben wir Gegenbeispiele an, die zeigen, daß die bisher für den allgemeinen Fall erwarteten und von uns für den unären Fall nachgewiesenen Eigenschaften schon bei aussagenlogischen Klauselmengen nicht mehr erfüllt sind.

(3) BIBEL zeigt in /Bi81c/, daß der KOWALSKI-Kalkül, wenn man bestimmte Restriktionen für die Tautologie- und die Subsumtionselimination vorsieht, für aussagenlogische Klauselmengen konsistent ist, d.h., daß jeder Graph, der aus dem initialen Graphen einer unerfüllbaren aussagenlogischen Klauselmenge abgeleitet werden kann, wieder unerfüllbar ist. Wir zeigen in dieser Arbeit die eigentlich interessante, stärkere Eigenschaft der Konfluenz, d.h., daß jeder Graph, der aus dem initialen Graphen einer unerfüllbaren aussagenlogischen Klauselmenge abgeleitet werden kann, mit der Resolutionsregel für Klauselgraphen widerlegbar ist.

Nach den Vorbereitungen in Kapitel 1 führen wir in Kapitel 2 den KOWALSKI-Kalkül und das Konzept des Filters ein. Dabei haben wir Wert darauf gelegt, alle Konzepte präzise und formal zu definieren und einen breiten begrifflichen Rahmen für unsere und weitere Untersuchungen zu schaffen. In dieser Arbeit betrachten wir eine neue Variante der Klauselgraph-Resolutionsregel, die i.a.

weniger Links erzeugt als die bisher vorgeschlagen Versionen von KOWALSKI /Ko75/, BRUGNOOHE /Br75/, /Ko79/ und von BIBEL /Bi81a, Bi81c/. Natürlich gelten unsere Ergebnisse auch für die bisherigen Versionen der Resolutionsregel. Neben der Tautologieelimination (die Gegenbeispiele (2) finden sich in Abschnitt 2.7) betrachten wir auch ausführlich die Subsumtionselimination. In Kapitel 3 sammeln wir die Ergebnisse für aussagenlogische Klauselmengen und zeigen den Konfluenzsatz (3). Schließlich betrachten wir in Kapitel 4 den Sonderfall der unären Klauselmengen und entwickeln die Lösung des unären Vollständigkeitsproblems. Die Kapitel 3 und 4 sind voneinander unabhängig, so daß der am Hauptergebnis interessierte Leser nach dem Studium von Kapitel 2 sofort zu Kapitel 4 übergehen kann.

K A P I T E L 1

V O R B E R E I T U N G E N

In diesem Kapitel definieren wir die Notationen für die Elemente des Resolutionskalküls und zitieren die für uns wichtigen Ergebnisse. Ausführliche Darstellungen des Resolutionskalküls findet man in /Ni71/, /CL73/ und /Lo78/. Der Resolutionskalkül wurde 1965 von J.A. ROBINSON /Ro65/ formuliert und ist als logischer Rahmen für automatische Beweiser gedacht. Methodisch baut er auf den frühen Arbeiten von SKOLEM /Sk28/ und HERBRAND /He29/ auf. Im letzten Abschnitt stellen wir noch einige Ergebnisse über konfluente Relationen zusammen.

1.1 Die Sprache des Resolutionskalküls

1.1.1 Variablensymbole. Die folgenden Symbole heißen Variablensymbole: x, y, z, u, v, w, $x_1, \ldots, w_1, x_2, \ldots$. Die Menge aller Variablensymbole bezeichnen wir mit VAR und vereinbaren für sie die obige alphabetische Ordnung. Metasymbole für Variablensymbole sind X, Y.

1.1.2 Konstantensymbole. Die folgenden Symbole heißen Konstantensymbole: a, b, c, d, $a_1, \ldots, d_1, a_2, \ldots$. Als Metasymbol benutzen wir a.

1.1.3 Funktionssymbole. Für $n \geq 1$ heißen die folgenden Symbole Funktionssymbole der Stelligkeit n: f^n, g^n, h^n, f_1^n, g_1^n, h_1^n, $f_2^n, \ldots$. Als Metasymbol benutzen wir f.

1.1.4 Prädikatensymbole. Für $n \geq 0$ heißen die folgenden Symbole Prädikatensymbole der Stelligkeit n: P^n, Q^n, R^n, S^n, T^n, $P_1^n, \ldots, T_1^n, P_2^n, \ldots$. Als Metasymbol benutzen wir P.

1.1.5 Terme. Variablensymbole oder Konstantensymbole sind Terme. Wenn f ein n-stelliges Funktionssymbol ist und $t_1, \ldots, t_n$ Terme sind, dann ist die Zeichenreihe $ft_1 \ldots t_n$ ein Term. Die Menge aller Terme bezeichnen wir mit TERME. Metasymbole für Terme sind s und t.

1.1.6 Atome. Wenn P ein n-stelliges Prädikatensymbol ist und $t_1, \ldots, t_n$ Terme sind, dann ist die Zeichenreihe $Pt_1 \ldots t_n$ ein Atom. Metasymbole für Atome sind K, L, M und N.

1.1.7 Literale. Atome sind Literale. Wenn K ein Atom ist, dann ist die Zeichenreihe $\sim K$ ein Literal. Das Symbol $\sim$ heißt Negationssymbol. Die Menge aller Literale bezeichnen wir mit $\mathbb{L}$. Metasymbole für Literale sind K, L, M und N. Wenn L ein Literal ist, bezeichnen wir mit $|L|$ das zu L gehörende Atom. Zwei Literale L und K heißen komplementär, wenn $|L| = |K|$ und $L \neq K$. Wenn L ein Literal ist, bezeichnen wir mit $\bar{L}$ das eindeutig bestimmte komplementäre Literal zu L.

1.1.8 Klauseln. Eine endliche, eventuell leere Menge von Literalen heißt Klau-

sel. Die leere Klausel ist die leere Menge und wird mit $\square$ bezeichnet. Die Menge aller Klauseln bezeichnen wir mit $\mathbb{C}$. Metasymbole für Klauseln sind C, D und E. Mit $|C|$ bezeichnen wir die Anzahl von Literalen in der Klausel C. Eine Klausel heißt Tautologie, wenn sie zwei Literale enthält, die zueinander komplementär sind. Eine Klausel heißt unär, wenn sie genau ein Literal enthält. Als Metasymbol für Mengen von Klauseln benutzen wir S.

1.1.9 Ausdrücke. Ein Ausdruck ist ein Term oder ein Literal oder eine Klausel. Als Metasymbol für Ausdrücke benutzen wir A.

1.1.10 Enthaltene Variablen. Sei A ein Ausdruck. Mit Var(A) bezeichnen wir die Menge der Variablensymbole, die in A vorkommen. Ein Ausdruck, in dem kein Variablensymbol vorkommt, heißt variablenfrei (engl: ground expression). Für eine Klauselmenge S sind Var(S) und Variablenfreiheit entsprechend definiert.

1.1.11 Varianten. Die Varianten einer Klausel C sind rekursiv wie folgt definiert: C ist eine Variante von C; wenn D eine Variante von C ist und X ein Variablensymbol, das in D vorkommt, dann ist die Klausel, die man aus D erhält, indem man alle Auftreten von X durch ein Variablensymbol Y, das in D nicht vorkommt, ersetzt, eine Variante von C.

1.1.12 Aussagenlogische Klauselmengen. Die nullstelligen Prädikatensymbole heißen aussagenlogische Atome. Ein Literal L heißt aussagenlogisch, wenn $|L|$ ein aussagenlogisches Atom ist. Eine Klausel C heißt aussagenlogisch, wenn alle Literale in C aussagenlogisch sind. Eine Klauselmenge S heißt aussagenlogisch, wenn alle Klauseln in S aussagenlogisch sind.

1.1.13 Schreibweise. Bei Funktions- und Prädikatensymbolen lassen wir den die Stelligkeit angebenden Hochindex weg und sorgen dafür, daß er sich aus dem Kontext ergibt. Für ein negatives Literal $\sim Pt_1 \ldots t_n$ schreiben wir $\bar{P}t_1 \ldots t_n$. Gelegentlich bringen wir die Stelligkeit von Funktions- oder Prädikatensymbolen dadurch zum Ausdruck, daß wir die Unterterme in Klammern einschließen und durch Kommata trennen, z.B. f(a,b) statt fab Eine Klausel $\{L_1, \ldots, L_n\}$ schreiben wir kurz als Zeichenreihe $L_1 \ldots L_n$. Wenn C eine Klausel und L ein Literal ist, dann ist $C - L := C - \{L\}$

1.2 Unifikationen

1.2.1 Substitutionen. Eine Abbildung σ: TERME $\longrightarrow$ TERME heißt Substitution, wenn gilt:

(a) Für jeden Term $ft_1 \ldots t_n$ ist $\sigma(ft_1 \ldots t_n) = f\sigma(t_1) \ldots \sigma(t_n)$.

(b) Für jede Konstante a ist $\sigma(a) = a$.

(c) Für höchstens endlich viele Variablen X ist $\sigma(X) \neq X$.

Als Metasymbole für Substitutionen verwenden wir σ, τ, θ und auch andere kleine griechische Buchstaben. Statt $\sigma(t)$ schreiben wir kurz σt. Die Identität TERME $\longrightarrow$ TERME ist eine Substitution und heißt die leere Substitution. Als Symbol für die leere Substitution benutzen wir ε. Wenn σ und τ Substitutionen sind, dann ist die Komposition $\tau\sigma$, $\tau\sigma(t) := \tau(\sigma(t))$ wieder eine Substitution. Die Komposition von Substitutionen ist eine assoziative, aber i.a. nicht kommutative Operation. Die Menge der Substitutionen ist mit der Komposition eine Halbgruppe mit dem Einselement ε.

1.2.2 Darstellung von Substitutionen. Die Zeichenreihe X/t heißt Komponente der Substitution σ, wenn X eine Variable ist mit $\sigma X = t$ und $X \neq t$. Eine Substitution hat nur endlich viele Komponenten. Die leere Substitution ist die einzige Substitution, die keine Komponente hat. Eine Substitution ist durch die Menge ihrer Komponenten eindeutig bestimmt, da sie dies durch ihre Restriktion auf VAR ist. Wie üblich identifizieren wir deshalb eine Substitution mit der Menge ihrer Komponenten. Damit sind für Substitutionen die Mengenoperationen Vereinigung, Durchschnitt und Differenz definiert.

1.2.3 Ber(σ) und Var(σ). Sei σ eine Substitution. Die Menge der Variablen X mit $\sigma X \neq X$ heißt Bereich von σ und wird mit Ber(σ) bezeichnet. Die Menge aller Variablen, die in der Komponentendarstellung von σ auftreten, wird mit Var(σ) bezeichnet. Beispielsweise ist für $\sigma = \{x/fy\}$ Ber(σ)$=\{x\}$ und Var(σ)$= \{x,y\}$.

1.2.4 Normale Substitutionen. Eine Substitution σ heißt normal, wenn sie idempotent (d.h. $\sigma\sigma = \sigma$) ist und für zwei verschiedene Variablen X und Y mit $\sigma X = Y$ stets $X < Y$ im Sinne der alphabetischen Ordnung aus 1.1.1 gilt.

1.2.5 Allgemeinste Substitution. Sei M eine Menge von Substitutionen. Eine Substitution $\sigma \in M$ heißt allgemeinste Substitution in M, wenn für jede Substi-

tution $\tau \in M$ eine Substitution θ existiert mit $\tau = \theta\sigma$.

1.2.6 Restriktionen von Substitutionen. Sei σ eine Substitution und V eine Menge von Variablen. Die Restriktion von σ auf V ist definiert durch $\sigma|_V :=$ $\{X/t \mid X/t \in \sigma \wedge X \in V\}$ und ist offensichtlich wieder eine Substitution.

1.2.7 Kanonische Fortsetzung von Substitutionen. Sei σ eine Substitution. Für ein Atom $Pt_1 \ldots t_n$ ist $\sigma(Pt_1 \ldots t_n) := P\sigma t_1 \ldots \sigma t_n$. Für ein negatives Literal $\sim L$ ist $\sigma(\sim L) := \sim\sigma(L)$. Für eine Klausel $\{L_1, \ldots, L_n\}$ ist $\sigma(\{L_1, \ldots, L_n\}) :=$ $\{\sigma L_1, \ldots, \sigma L_n\}$. Für eine Klauselmenge $\{C_1, \ldots, C_n\}$ ist $\sigma(\{C_1, \ldots, C_n\}) :=$ $\{\sigma C_1, \ldots, \sigma C_n\}$.

1.2.8 U-Substitutionen. Sei V eine Menge von Variablen. Eine Substitution θ mit $Ber(\theta) \subset V$ heißt U-Substitution (Umbenennungssubstitution) für V, wenn θ auf V injektiv ist und θ Variablen in Variablen abbildet. Eine Substitution θ heißt U-Substitution für einen Ausdruck A, wenn θ eine U-Substitution für Var(A) ist. Für eine U-Substitution θ heißt $\theta^{-1} := \{Y/X \mid X/Y \in \theta\}$ die inverse Substitution. Offensichtlich ist eine Klausel D genau dann eine Variante einer Klausel C, wenn es eine U-Substitution θ für C mit $D = \theta C$ gibt. Wenn θ eine U-Substitution für C mit $D = \theta C$ ist, dann gilt $C = \theta^{-1} D$.

1.2.9 Lemma. σ, τ, θ seien Substitutionen und A ein Ausdruck. Dann gilt:

(a) $(\tau\sigma)A = \tau(\sigma A)$.

(b) $(\theta\tau)\sigma = \theta(\tau\sigma)$.

(c) Wenn für jeden Term t $\sigma t = \tau t$ ist, dann ist $\sigma = \tau$.

(d) Wenn C eine Klausel mit $|\sigma C| = |C|$ und $L \in C$ ein Literal ist, dann ist $\sigma(C - L) = \sigma C - \sigma L$.

Beweis. Für (a) - (c) siehe /Ro65/. (d) ist evident. []

1.2.10 Unifikation. Sei M eine Menge von Ausdrücken. M heißt unifizierbar, wenn es eine Substitution σ mit $|\sigma M| \leq 1$ gibt. σ heißt dann Unifikator für M.

1.2.11 Unifikationssatz. Zu jeder unifizierbaren Menge von Ausdrücken existiert genau ein normaler allgemeinster Unifikator. Die Unifizierbarkeit ist mit linearem Aufwand entscheidbar. Ebenso läßt sich der normale allgemeinste Unifikator, wenn er existiert, mit linearem Aufwand berechnen. Wenn σ der normale allgemeinste Unifikator für eine Menge M von Ausdrücken ist, dann gilt

$Var(\sigma) \subset Var(M)$.

Beweis. Für die Existenz und Eindeutigkeit des normalen allgemeinsten Unifikators verweisen wir auf /Ro65/ und /He82/. Ein linearer Unifikationsalgorithmus wird von PATERSON und WEGMAN in /PW78/ angegeben. []

1.1.12 <u>$nau(L_1,\ldots,L_n)$</u>. Seien $L_1,\ldots,L_n$ Literale. Den normalen allgemeinsten Unifikator von $L_1,\ldots,L_n$ bezeichnen wir, falls er existiert, mit $nau(L_1,\ldots,L_n)$.

1.1.13 <u>Potentiell komplementäre Literale</u>. Zwei Literale L und K heißen potentiell komplementär, wenn es für K eine U-Substitution θ gibt, so daß $\bar{L}$ und θK unifizierbar sind.

1.2.14 <u>Lemma</u>. L und K seien Literale und σ sei eine Substitution. Wenn σL und K potentiell komplementär sind, dann sind auch L und K potentiell komplementär.

Beweis. Siehe /He82/. []

1.2.15 <u>Kombinationssubstitutionen</u>. Seien $\sigma = \{X_1/t_1,\ldots,X_n/t_n\}$ und $\tau = \{Y_1/s_1,\ldots,Y_m/s_m\}$ Substitutionen. Dann heißt der Unifikator $\sigma*\tau :=$ $nau(PX_1\ldots X_nY_1\ldots Y_m,\ Pt_1\ldots t_ns_1\ldots s_m)$, wenn er existiert, die Kombinationssubstitution von σ und τ. Die <u>Kombinationsoperation *</u> ist kommutativ und assoziativ. Wenn $M = \{\sigma_1,\ldots,\sigma_n\}$ eine Menge von Substitutionen ist, dann heißt die Substitution $(*\sigma \mid \sigma \in M) := \sigma_1*\ldots*\sigma_n$, falls sie existiert, die Kombinationssubstitution von M. Eine Menge M von Substitutionen heißt <u>konsistent</u>, wenn die Kombinationssubstitution von M existiert.

1.2.16 <u>Lemma</u>.

(a) Wenn σ eine normale Substitution ist, dann ist $\sigma = \sigma*\sigma$.

(b) Für zwei konsistente Substitutionen σ und τ gilt:

$$\sigma*\tau = (\sigma*\tau)\sigma = (\sigma*\tau)\tau = \sigma(\sigma*\tau) = \tau(\sigma*\tau).$$

(c) Für zwei normale Substitutionen σ und τ mit $Ber(\tau) \cap Var(\sigma) = \emptyset$ gilt:

$$\sigma*\tau = \sigma\tau.$$

(d) Wenn L und K variablendisjunkte Literale sind, und σ ein normaler Unifikator von L und K ist, dann gilt $\sigma = \sigma * nau(L,K)$.

Beweis. Siehe /He82/. []

1.3 Semantik von Klauselmengen

In diesem Abschnitt skizzieren wir die Äquivalenz zwischen den üblichen, mit allen Junktoren und Quantoren ausgestatteten prädikatenlogischen Sprachen und der aufs äußerste abgemagerten Sprache des Resolutionskalküls.

1.3.1 Basismodelle. Eine Menge $\mathcal{M}$ von variablenfreien Literalen heißt Basismodell (engl: ground model), wenn $\mathcal{M}$ keine komplementären Literale enthält.

1.3.2 Erfüllbarkeit von variablenfreien Klauselmengen. Eine variablenfreie Klauselmenge S heißt erfüllbar, wenn es ein Basismodell $\mathcal{M}$ gibt, so daß für jede Klausel $C \in S$ gilt: $C \cap \mathcal{M} \neq \emptyset$. Eine variablenfreie Klauselmenge heißt unerfüllbar, wenn sie nicht erfüllbar ist.

1.3.2 Erfüllbarkeit von Klauselmengen. Eine Klauselmenge S heißt unerfüllbar, wenn es eine Menge S' von Varianten von Klauseln in S und eine Substitution σ gibt, so daß $\sigma S'$ variablenfrei und im Sinne von 1.3.2 unerfüllbar ist. Eine Klauselmenge heißt erfüllbar, wenn sie nicht unerfüllbar ist.

1.3.4 Klauselmengen und Formeln in SKOLEM-Normalform. Einer Klauselmenge kann wie folgt eine prädikatenlogische Formel zugeordnet werden: Ein Literal entspricht einer atomaren oder einer negierten atomaren Formel; einer Klausel $C = \{L_1, \ldots, L_n\}$ mit $Var(C) = \{X_1, \ldots, X_m\}$ entspricht eine geschlossene Formel $(\forall X_1 \ldots X_m\colon L_1 \vee \ldots \vee L_n)$; schließlich entspricht einer Klauselmenge $S = \{C_1, \ldots, C_n\}$ eine Formel $C_1 \wedge \ldots \wedge C_n$. Da die Ordnungen für Klauseln, Literale und Variablen bei Klauselmengen irrelevant und bei Formeln relevant sind, entsprechen einer Klauselmenge i.a. mehrere Formeln, die sich aber lediglich in diesen Ordnungen unterscheiden. Für Formeln ist der Begriff erfüllbar (A unerfüllbar gdw ~A allgemeingültig) unabhängig von Klauselmengen definiert. Der Satz von HERBRAND (1930) garantiert, daß eine Klauselmenge genau dann erfüllbar ist, wenn eine ihr entsprechende Formel erfüllbar ist. Eine Formel A heißt in SKOLEM-Normalform (SNF), wenn es eine Klauselmenge gibt, für die A eine entsprechende Formel ist. Wenn A eine Formel in SNF ist, gibt es genau eine Klauselmenge S_A, so daß A eine entsprechende Formel zu S_A ist.

1.3.5 Satz (SKOLEM 1928). Zu jeder prädikatenlogischen Formel A gibt es eine

Formel A* in SNF, so daß A genau dann erfüllbar ist, wenn A* erfüllbar ist. Zu A läßt sich eine Formel A* mit polynomialen Aufwand berechnen.

Beweis. Siehe /Lo78/ und /Sk28/.

1.3.6 Zusammenfassung. Beim Automatischen Beweisen geht es darum, zu allgemeingültigen (engl: valid) Formeln der Struktur $A_1 \wedge \ldots \wedge A_n \rightarrow T$ durch einen Algorithmus einen Beweis zu konstruieren. Dabei heißen die Formeln A_i *Axiome* und die Formel T *Theorem*. Die obigen Sätze zeigen, daß dies geschehen kann, indem man einen Beweis für die Unerfüllbarkeit einer entsprechenden Klauselmenge konstruiert:

$$
\begin{array}{lll}
 & A_1 \wedge \ldots \wedge A_n \rightarrow T & \text{allgemeingültig} \\
\Longleftrightarrow & A_1 \wedge \ldots \wedge A_n \wedge \sim T & \text{unerfüllbar} \\
\Longleftrightarrow & A_1^* \wedge \ldots \wedge A_n^* \wedge (\sim T)^* & \text{unerfüllbar} \\
\Longleftrightarrow & S_{A_1^*} \cup \ldots \cup S_{A_n^*} \cup S_{(\sim T)^*} & \text{unerfüllbar.}
\end{array}
$$

1.4 Der Resolutionskalkül

1.4.1 F-Substitutionen. Eine Substitution σ heißt F-Substitution (Faktorisierungssubstitution) für eine Klausel C, wenn es unifizierbare Literale $L_1,\ldots,L_n \in C$, $n \geq 1$ gibt, so daß $\sigma = nau(L_1,\ldots,L_n)$ ist. Offensichtlich ist jede F-Substitution σ für eine Klausel C normal und erfüllt $Var(\sigma) \subset Var(C)$. Die leere Substitution ε ist eine F-Substitution für jede Klausel.

1.4.2 Faktoren. Eine Klausel D heißt Faktor einer Klausel C, wenn es eine F-Substitution σ für C gibt, so daß $D = \sigma C$. Ein Faktor D einer Klausel C heißt echt, wenn $|D| < |C|$.

1.4.3 Lemma. Sei C eine Klausel und σ eine normale Substitution. Dann existiert eine F-Substitution τ für C mit $|\tau C| = |\sigma C|$ und $\sigma = \sigma * \tau$.

Beweis. Siehe /He82/. []

1.4.4 Resolventen. Seien C und D variablendisjunkte Klauseln und $L \in C$ und $K \in D$ Literale, für die $\sigma := nau(L,\bar{K})$ existiert. Dann heißt die Klausel $Res(C,L;D,K) := (\sigma C - \sigma L) \cup (\sigma D - \sigma K)$ (binäre) Resolvente von C und D. C und D heißen die Elternklauseln der Resolvente.

1.4.5 Lemma. Sei S eine Klauselmenge. Sei D eine Variante oder ein Faktor einer Klausel in S, oder sei D eine Resolvente, deren Elternklauseln Klauseln in S sind. Dann ist S erfüllbar gdw $S \cup \{D\}$ erfüllbar ist.

Beweis. Siehe /Lo78/ oder /CL73/. []

1.4.6 V-Resolventen und FV-Resolventen. Seien C, D und E Klauseln. E heißt

- V-Resolvente von C und D, wenn es Varianten C' bzw. D' von C bzw. D gibt, so daß Literale L und K existieren mit $E = Res(C',L;D',K)$.
- FV-Resolvente von C und D, wenn es Faktoren C' bzw. D' von C bzw. D gibt, so daß E eine V-Resolvente von C' und D' ist.

1.4.7 Resolutionsableitungen. Sei C eine Klausel und S eine Klauselmenge. Eine endliche Folge $C_1 \ldots C_n$, $n \geq 1$ von Klauseln heißt Resolutionsableitung für

C aus S, wenn $C_n = C$ ist und für jedes k mit $1 \leq k \leq n$ gilt: C_k ist Variante einer Klausel in S, oder es existieren $i < k$ und $j < k$, so daß C_k FV-Resolvente von C_i und C_j ist. Eine Resolutionsableitung $C_1 \ldots C_n$ heißt tautologiefrei, wenn kein C_i eine Tautologie ist.

1.4.8 Widerlegungen. Eine (Resolutions) Widerlegung einer Klauselmenge S ist eine Resolutionsableitung für die leere Klausel aus S.

1.4.9 Lemma. Zu jeder unerfüllbaren Klauselmenge S existiert eine tautologiefreie Widerlegung.

Beweis. Siehe /Lo78/ oder /CL73/. []

1.4.10 Satz (Korrektheit und Vollständigkeit des Resolutionskalküls). Eine Klauselmenge S ist genau dann unerfüllbar, wenn eine Widerlegung für S existiert.

Beweis.
"<==" (Korrektheit). Sei S eine Klauselmenge, zu der eine Widerlegung existiert. Angenommen, S ist erfüllbar. Dann liefert wiederholtes Anwenden von Lemma 1.4.5 auf die Widerlegung, daß $S \cup \{\square\}$ erfüllbar ist. W!
"==>" (Vollständigkeit). Lemma 1.4.9. []

1.4.11 Lemma. Sei S eine Klauselmenge und $C \in S$. Dann gilt:
(a) Wenn C eine Tautologie ist, dann ist S erfüllbar gdw $S - \{C\}$ erfüllbar ist.
(b) Wenn C isoliert in S ist, dann ist S erfüllbar gdw $S - \{C\}$ erfüllbar ist.
C heißt isoliert (engl: pure) in S, wenn es in C ein Literal gibt, welches zu keinem Literal in einer von C verschiedenen Klausel in S potentiell komplementär ist.

Beweis. Siehe /Lo78/ oder /CL73/. []

1.5 Konfluente Relationen

Wir sammeln hier einige Notationen und Ergebnisse für binäre Relationen, die wir der Arbeit /Hu80/ entnommen haben.

1.5.1 $\longrightarrow$, $\xrightarrow{+}$, $\xrightarrow{*}$. Sei $\longrightarrow$ eine binäre Relation auf einer Menge M. Statt $\langle a,b\rangle \in \longrightarrow$ schreiben wir wie üblich $a \longrightarrow b$. Mit $\xrightarrow{+}$ bzw. $\xrightarrow{*}$ bezeichnen wir die transitive bzw. die transitive und reflexive Hülle von $\longrightarrow$.

1.5.2 Ableitungen. Sei $\longrightarrow$ eine binäre Relation auf M. Eine Folge $a_1 \ldots a_n$ mit $n \geq 1$ heißt $\longrightarrow$-Ableitung, wenn für alle i $a_i \longrightarrow a_{i+1}$ gilt. Dabei heißt n die Länge der Ableitung. Eine unendliche Folge $a_1 a_2 a_3 \ldots a_n \ldots$ heißt nichtabbrechende $\longrightarrow$-Ableitung, wenn für alle n das Anfangsstück $a_1 \ldots a_n$ eine $\longrightarrow$-Ableitung ist.

1.5.3 Normalformen. Sei $\longrightarrow$ eine binäre Relation auf M. b heißt $\longrightarrow$-Normalform zu a, wenn $a \xrightarrow{*} b$ gilt und kein $c \in M$ mit $b \longrightarrow c$ existiert.

1.5.4 Noethersche Relationen. Eine binäre Relation $\longrightarrow$ auf M heißt noethersch, wenn es keine nichtabbrechende $\longrightarrow$-Ableitung gibt. Wenn $\longrightarrow$ noethersch ist, dann existiert zu jedem $a \in M$ mindestens eine $\longrightarrow$-Normalform.

1.5.5 Konfluenz. Eine binäre Relation $\longrightarrow$ auf M heißt konfluent, wenn für alle $a,b,c \in M$ mit $a \xrightarrow{*} b$ und $a \xrightarrow{*} c$ ein $d \in M$ existiert mit $b \xrightarrow{*} d$ und $c \xrightarrow{*} d$. Wenn $\longrightarrow$ konfluent ist, dann existiert zu jedem $a \in M$ höchstens eine Normalform. Eine binäre Relation $\longrightarrow$ auf M heißt lokal konfluent, wenn für alle a, $b,c \in M$ mit $a \longrightarrow b$ und $a \longrightarrow c$ ein $d \in M$ existiert mit $b \xrightarrow{*} d$ und $c \xrightarrow{*} d$.

1.5.6 Satz. Für eine noethersche binäre Relation $\longrightarrow$ auf M sind die drei folgenden Aussagen äquivalent:

(a) $\longrightarrow$ ist konfluent.

(b) $\longrightarrow$ ist lokal konfluent.

(c) Zu jedem $a \in M$ existiert genau eine $\longrightarrow$-Normalform.

KAPITEL 2

DER KOWALSKI-KALKÜL

Klauselgraphen (clause graphs, connection graphs) sind im Automatischen Beweisen eine verbreitete Datenstruktur, für die viele verschiedene Ableitungssysteme untersucht wurden. Beispiele sind die Arbeiten von ANDREWS /An76, An81/, von BIBEL /Bi82b/, von CHANG und SLAGLE /CS79/ (rewrite rules), von SICKEL /Si76/ (graph-unrolling) und eben die auf dem Resolutionsprinzip basierende Connection Graph Proof Procedure von KOWALSKI /Ko75/, die - im Unterschied zu den anderen Verfeinerungen des Resolutionskalküls - das Löschen vieler der mit

dem Fortschreiten des Ableitungsprozesses erzeugten Klauseln erlaubt.

Der nicht gerade glücklich gewählte Begriff "Connection Graph Proof Procedure" (welcher Graph hat keine Konnektionen?) steht für eine ganze Familie von Beweisprozeduren, denen ein gemeinsamer Kalkül zugrunde liegt. Dieses von uns als KOWALSKI-Kalkül bezeichnete Ableitungssystem für Klauselgraphen besteht aus einer Resolutions- und einer Faktorisationsregel sowie zwei Reduktionsregeln zur Elimination von Tautologien und Subsumtionen. Zusätzlich zum Kalkül führen wir dann das Konzept des Filters ein, das die Formulierung spezieller Beweisprozeduren erlaubt, indem man Restriktionen (z.B. für die Anwendung der Reduktionsregeln) und Suchstrategien (z.B. breadth first, depth first) vorgibt. Das Vollständigkeitsproblem des KOWALSKI-Kalküls besteht nun darin, hinreichende Kriterien für vollständige Filter anzugeben. Dabei heißt ein Filter vollständig, wenn die durch ihn definierte Beweisprozedur für jede unerfüllbare Klauselmenge mit der Erzeugung der leeren Klausel terminiert. Bis heute ist offen, ob ein solcher vollständiger Filter existiert.

Wir legen dieser Arbeit eine neue Variante der Klauselgraph-Resolutionsregel zugrunde. Sie unterscheidet sich von früheren Versionen dadurch, daß sie i.a. stärker reduzierend ist, d.h., daß sie weniger neue Kanten in den Graphen einbringt. Ihre Einführung ist dadurch gerechtfertigt, daß alle bekannten und neuen Vollständigkeitsergebnisse auch für diese Regel gezeigt werden können.

In dieser Arbeit wurde besonderer Wert darauf gelegt, alle Konzepte für den KOWALSKI-Kalkül präzise und formal zu definieren und einen breiten begrifflichen Rahmen für die folgenden Untersuchungen zu schaffen. In diesem und zum Teil im folgenden Kapitel sammeln wir - neben neu gezeigten Ergebnissen - die bisher bekannten Resultate für den KOWALSKI-Kalkül.

2.1 Klauselgraphen

Ein Klauselgraph ist eine graphenähnliche Datenstruktur für eine Menge von Klauseln. Seine Kanten stellen mögliche Resolutionen dar. Die folgenden Beispiele geben einen ersten Eindruck und zeigen die Unterschiede zu gewöhnlichen Graphen:

(a) $\boxed{\bar{P}a}$ — $\boxed{Px \mid Qy}$ — $\boxed{\bar{Q}b}$

(b) $\boxed{Pa}$ — $\boxed{\bar{P}x \mid Pfx}$ — $\boxed{\bar{P}fffy}$

(c) $\boxed{\bar{P}ayx \mid \bar{P}bzx}$ — $\boxed{Puuc \mid Pvbd}$

(d) $\boxed{Pa}$ — $\boxed{\bar{P}x \mid Pfx}$ — $\boxed{\bar{P}y \mid Pfy}$ — $\boxed{\bar{P}fffz \mid Qz}$ $\boxed{\bar{Q}a \mid \bar{P}a}$ — $\boxed{Pa}$

(e) $\boxed{Pa}$ $\boxed{\bar{P}b}$ $\square$

Jeder Knoten repräsentiert eine Klausel und wird als eine Kette nebeneinanderliegender Zellen, die mit den Literalen der Klausel markiert sind, gezeichnet. Eine Zelle zusammen mit dem enthaltenen Literal heißt Literalauftreten. Die Kanten im Klauselgraphen heißen Links. Jeder Link verbindet zwei potentiell komplementäre Literalauftreten. Der Graph (e) hat drei Knoten und keinen Link. Der rechteste Knoten repräsentiert die leere Klausel und ist als leere Zelle gezeichnet. An einen Knoten, der die leere Klausel repräsentiert, kann kein Link führen.

2.1.1 Knoten. Die folgenden Symbole heißen Knoten: $k_1, k_2, k_3, \ldots$ Die Menge aller Knoten bezeichnen wir mit $\mathbb{K}$. Als Metasymbole für Knoten verwenden wir a, b, c, d, e, f

2.1.2 Literalauftreten. Ein Literalauftreten L^a ist ein geordnetes Paar $\langle L,a\rangle \in \mathbb{L} \times \mathbb{K}$. Zwei Literalauftreten L^a und K^b heißen potentiell komplementär,

wenn L und K potentiell komplementär sind.

2.1.3 Links. Ein Link L^aK^b ist ein ungeordnetes Paar $\{L^a, K^b\}$, wobei L^a und K^b potentiell komplementäre Literalauftreten sind. Als Metasymbol für Links benutzen wir ℓ. Für einen Link $\ell = L^aK^b$ heißen a und b die Inzidenzknoten von ℓ. Genauer sprechen wir vom positiven und vom negativen Inzidenzknoten eines Links. Ein Link ℓ inzidiert mit einem Literalauftreten L^a, wenn $L^a \in \ell$. Wenn ℓ mit L^a inzidiert, sagen wir auch, daß ℓ an L^a führt. Ein Link ℓ verbindet die Literalauftreten L^a und K^b, wenn $\ell = L^aK^b$. Ein Link L^aK^b heißt Autolink, wenn $a = b$ ist.

2.1.4 Klauselgraphen. Ein Tupel $\mathcal{G} = \langle$Knoten,C,Links$\rangle$ heißt Klauselgraph wenn gilt:

(a) Knoten ist eine endliche, eventuell leere Teilmenge von $\mathbb{K}$.

(b) C ist eine Abbildung Knoten $\longrightarrow \mathbb{C}$. Wir schreiben C_a für C(a).

(c) Link ist eine Menge von Links. Für jeden Link $L^aK^b \in$ Links gilt: $a \in$ Knoten und $L \in C_a$ und $b \in$ Knoten und $K \in C_b$.

(d) Für zwei verschiedene $a, b \in$ Knoten gilt stets, daß C_a und C_b variablendisjunkt sind.

Ein Knoten a heißt Knoten in $\mathcal{G}$, wenn $a \in$ Knoten. Ein Literalauftreten L^a heißt Literalauftreten in $\mathcal{G}$, wenn a Knoten in $\mathcal{G}$ und $L \in C_a$ ist. Ein Link ℓ heißt Link in $\mathcal{G}$, wenn $\ell \in$ Links.

Die Knoten sind Namen für Klauseln. Sie werden benötigt , weil wir auch Graphen betrachten wollen, in denen die gleiche Klausel mehrfach auftritt. Zwei verschiedene Knoten, die die gleiche Klausel repräsentieren, können sich in den inzidenten Links unterscheiden. Ein Beispiel dafür ist der Graph (d).

Die in Bedingung (d) geforderte Variablendisjunktheit ist nicht prinzipiell erforderlich, erleichtert aber an einigen Stellen die formale Behandlung des Untersuchungsgegenstandes.

2.1.5 Bezeichnungen. Sei $\mathcal{G} = \langle$Knoten,C,Links$\rangle$ ein Klauselgraph. Dann bezeichnet

- $|\mathcal{G}| := |\text{Knoten}|$ die Anzahl der Knoten in $\mathcal{G}$.
- $S(\mathcal{G}) := \{C_a \mid a \in \text{Knoten}\}$ die Klauselmenge von $\mathcal{G}$.
- $\text{Var}(\mathcal{G}) := \text{Var}(S(\mathcal{G}))$ die Variablenmenge von $\mathcal{G}$.

Die Anzahl der Knoten des Graphen (d) ist sechs, die des Graphen (e) ist drei.

Die Klauselmenge des Graphen (e) ist $\{Pa, \bar{P}b, \square\}$. Die Variablenmenge des Graphen (c) ist $\{x,y,z,u,v\}$.

2.1.6 Eigenschaften. Sei $\mathcal{G}$ ein Klauselgraph. $\mathcal{G}$ heißt

- total, wenn für zwei komplementäre Literalauftreten L^a und K^b in $\mathcal{G}$ stets gilt, daß L^aK^b Link in $\mathcal{G}$ ist.
- erfüllbar, wenn $S(\mathcal{G})$ erfüllbar ist.
- unerfüllbar, wenn $S(\mathcal{G})$ unerfüllbar ist.
- tautologiefrei, wenn $S(\mathcal{G})$ keine Tautologie enthält.
- aussagenlogisch, wenn $S(\mathcal{G})$ aussagenlogisch ist.

Die Graphen (a), (b) und (c) sind total, (d) und (e) sind nicht total. Alle Graphen (a) - (e) sind unerfüllbar und tautologiefrei. Keiner der Graphen (a) - (e) ist aussagenlogisch.

2.1.7 Der leere Klauselgraph (). Der leere Klauselgraph ist der eindeutig bestimmte Klauselgraph $\mathcal{G}$ mit $|\mathcal{G}| = 0$. Wir bezeichnen ihn mit (). Der leere Klauselgraph ist total, erfüllbar, tautologiefrei und aussagenlogisch.

2.1.8 Die Klauselgraphen ($\square$). Jeden Klauselgraphen $\mathcal{G}$ mit $|\mathcal{G}| = 1$ und $S(\mathcal{G}) = \{\square\}$ bezeichnen wir mit ($\square$). Die Graphen ($\square$) sind total, unerfüllbar, tautologiefrei und aussagenlogisch.

2.1.9 Subklauselgraphen. Seien $\mathcal{G}$ =<Knoten,C,Links> und $\mathcal{G}'$ =<Knoten',C',Links'> Klauselgraphen. $\mathcal{G}$ heißt Subklauselgraph von $\mathcal{G}'$, wenn Knoten $\subset$ Knoten', Links $\subset$ Links' und C die Restriktion von C' auf Knoten ist. Die Inklusionsrelation $\subset$,

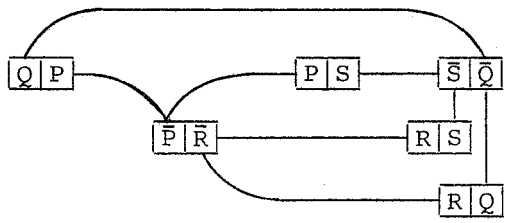
$$\mathcal{G} \subset \mathcal{G}' \quad :\Longleftrightarrow \quad \mathcal{G} \text{ ist Subgraph von } \mathcal{G}'$$

ist eine Halbordnung auf der Menge aller Klauselgraphen. Der *kleinste Klauselgraph* ist der leere Klauselgraph. Der größte Subgraph eines Klauselgraphen $\mathcal{G}$ ist $\mathcal{G}$ selbst.

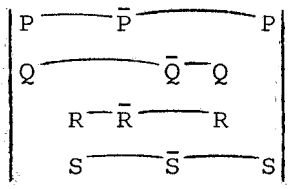
2.1.10 Unwesentlich verschiedene Klauselgraphen. Die Gleichheitsrelation für Klauselgraphen ist sehr fein. Zum Beispiel gibt es für ($\square$) unendlich viele verschiedene Varianten, die sich nur in der Knotenbenennung unterscheiden. Zwei Klauselgraphen heißen unwesentlich verschieden, wenn sie sich nur in den

Knoten- und Variablenbenennungen unterscheiden. (□) bezeichnet also eine ganze Äquivalenzklasse von Klauselgraphen.

2.1.11 Darstellung von AT-Klauselgraphen. Ein aussagenlogischer, tautologiefreier Klauselgraph heißt AT-Klauselgraph. Ein AT-Klauselgraph kann keine Autolinks enthalten, da bei aussagenlogischen Klauseln nur Tautologien potentiell komplementäre Literale enthalten. Für AT-Klauselgraphen verwenden wir eine Verfeinerung der in /Bi81c/ benutzten Matrixdarstellung. Der AT-Klauselgraph



hat beispielsweise die folgende Matrixdarstellung:



Jedem Knoten des Graphen entspricht dabei eine Spalte der Matrix. Da wir Tautologien ausgeschlossen haben, ist es möglich, jedem Atom eine separate Zeile zuzuordnen. Dadurch verlaufen Links immer nur innerhalb einer Zeile. Da in den von uns betrachteten AT-Klauselgraphen stets nur sehr wenige Links fehlen, läßt sich eine weitere Verbesserung der Lesbarkeit und Schreibbarkeit durch eine Negativdarstellung der Links erreichen: Nur die Links, die fehlen, werden gepunktet eingezeichnet. Der obige AT-Klauselgraph hat in Negativdarstellung überhaupt keine Kante mehr, da er total ist.

Um die Links in der Matrixdarstellung eines AT-Klauselgraphen bezeichnen zu können, denken wir uns die Spalten von links nach rechts mit eins beginnend durchnumeriert. $L^n K^m$ bezeichnet dann den Link, der das Literal L in der n-ten Spalte mit dem Literal K in der m-ten Spalte verbindet.

2.1.12 Elementaroperationen. Seien $\mathcal{G}$ ein Klauselgraph, ℓ ein Link, a ein Knoten und C eine Klausel. Dann sind die Elementaroperationen für das Hinzufügen und Löschen von Links und Knoten in Klauselgraphen wie folgt definiert:

- $\mathcal{G} - \ell$ bezeichnet den Klauselgraphen, den man aus $\mathcal{G}$ erhält, wenn man in $\mathcal{G}$ den Link ℓ löscht.
- $\mathcal{G} + \ell$ bezeichnet den Klauselgraphen, den man aus $\mathcal{G}$ erhält, wenn man in $\mathcal{G}$ den Link ℓ einfügt. Die Operation ist *nur dann* definiert, wenn die durch ℓ verbundenen Literalauftreten bereits in $\mathcal{G}$ enthalten sind.
- $\mathcal{G} - a$ bezeichnet den Klauselgraphen, den man aus $\mathcal{G}$ erhält, wenn man in $\mathcal{G}$ den Knoten a und alle zu a inzidenten Links löscht.
- $\mathcal{G} + \langle a,C\rangle$ bezeichnet den Klauselgraphen, den man aus $\mathcal{G}$ erhält, wenn man in $\mathcal{G}$ den Knoten a mit $C_a := C$ einfügt. Die Operation ist *nur dann* definiert, wenn a bisher nicht in $\mathcal{G}$ enthalten ist und wenn $\mathcal{G}$ und C variablendisjunkt sind.

Der folgende Algorithmus bildet zu einer Klauselmenge einen entsprechenden Graphen. Die Definitionen für "$\pi(\mathcal{G})$" und "π-reduziert" finden sich in Abschnitt 2.2.

2.1.13 Algorithmus Initialer Klauselgraph

Eingabe: S : Klauselmenge;

Variablen: $\mathcal{G}$: Klauselgraph := ();
C, D : $\mathbb{C}$; L, M : $\mathbb{L}$; a, e : $\mathbb{K}$;

1: Für jede Klausel $C \in S$, die keine Tautologie ist
1.1: a := Knoten, der nicht in $\mathcal{G}$ vorkommt;
1.2: D := Variante von C, die variablendisjunkt zu $\mathcal{G}$ ist;
1.3: $\mathcal{G} := \mathcal{G} + \langle a,D\rangle$;
1.4: Für jedes Literal $L \in C_a$
1.4.1: Für jedes Literalauftreten M^e in $\mathcal{G}$, das potentiell komplementär zu L^a ist
1.4.1.1: $\mathcal{G} := \mathcal{G} + M^e L^a$;

2: $\mathcal{G} := \pi(\mathcal{G})$;

Ausgabe: $\mathcal{G}$;

Nachbedingung: $\mathcal{G}$ tautologiefrei, total, π-reduziert und $S(\mathcal{G}) \subset S$ und $\mathcal{G}$ erfüllbar gdw S erfüllbar.

Verifikation: Die Nachbedingung ist evident bis auf die Äquivalenz der Erfüllbarkeit von $\mathcal{G}$ und S. Diese ergibt sich aber leicht aus Lemma 1.4.11. []

Da die mit dem obigen Algorithmus zu einer Klauselmenge S erhaltbaren Graphen im Sinne von 2.1.10 nur unwesentlich verschieden sind, ist es legitim, von *dem* initialen Klauselgraphen zu einer Klauselmenge S zu sprechen. Wir bezeichnen ihn mit $\underline{\mathcal{G}_S}$.

Der Graph (a) ist der initiale Graph zu der Klauselmenge $\{\bar{P}a, PxQy, \bar{Q}b\}$. Er ist aber auch der initiale Graph zu $\{\bar{P}a, PxQy, \bar{Q}b, QbRz, \bar{R}uPu\bar{P}u\}$. Die Graphen (d) und (e) können zu keiner Klauselmenge initial sein, da sie nicht total sind.

2.2 Die π-Reduktion

Ein Hauptmerkmal des KOWALSKI-Kalküls ist das sogenannte *π-Prinzip für Klauselgraphen*. Dieses verlangt, daß nach einem Ableitungsschritt stets solange Knoten mit den inzidenten Links gelöscht werden, bis an jedes Literalauftreten im Graphen wenigstens ein Link führt, der kein Autolink ist.

2.2.1 π-reduzierte Klauselgraphen. Ein Literalauftreten in einem Klauselgraphen heißt isoliert (engl: pure), wenn es mit keinem oder nur mit Autolinks inzidiert. Ein Knoten heißt isoliert, wenn er ein isoliertes Literalauftreten enthält. Ein Klauselgraph heißt π-reduziert, wenn er keinen isolierten Knoten enthält.

Die Graphen (a), (b) und (c) aus 2.1 sind π-reduziert. In Graph (d) sind die Literalauftreten Qz und $\bar{Q}$a isoliert. Im Graphen (e) ist der Knoten $\square$ nicht isoliert. Die Graphen ($\square$) sind π-reduziert.

Wir werden nun zeigen, daß zu jedem Klauselgraphen *der* größte π-reduzierte Subgraph existiert, und daß er sich berechnen läßt, indem man in einer beliebigen Reihenfolge solange isolierte Knoten löscht, bis keiner mehr vorhanden ist.

2.2.2 Die π-Relation. Sei $\mathcal{G}$ ein Klauselgraph. Wir schreiben $\mathcal{G} \xrightarrow[\pi]{a} \mathcal{G}'$, wenn $\mathcal{G}' = \mathcal{G} - a$ und a ein isolierter Knoten in $\mathcal{G}$ ist. Wir schreiben $\mathcal{G} \xrightarrow[\pi]{} \mathcal{G}'$, wenn es einen Knoten a mit $\mathcal{G} \xrightarrow[\pi]{a} \mathcal{G}'$ gibt. Die entsprechende Relation $\xrightarrow[\pi]{}$ ist eine binäre Relation auf der Menge aller Klauselgraphen.

2.2.3 Satz. Die π-Relation ist noethersch und konfluent.

Beweis. Daß die π-Relation noethersch ist, ist evident, da für $\mathcal{G} \xrightarrow[\pi]{} \mathcal{G}'$ stets $|\mathcal{G}'| = |\mathcal{G}| - 1$ gilt. Es genügt also zu zeigen, daß die π-Relation lokal konfluent ist. Seien also $\mathcal{G}$, $\mathcal{G}_a$, $\mathcal{G}_b$ Klauselgraphen und a, b Knoten mit $\mathcal{G} \xrightarrow[\pi]{a} \mathcal{G}_a$ und $\mathcal{G} \xrightarrow[\pi]{b} \mathcal{G}_b$. Wir haben zu zeigen, daß es einen Graphen $\mathcal{G}'$ gibt mit $\mathcal{G}_a \xrightarrow[\pi]{*} \mathcal{G}'$ und $\mathcal{G}_b \xrightarrow[\pi]{*} \mathcal{G}'$.

Fall 1: $a = b$. $\mathcal{G}' := \mathcal{G}_a = \mathcal{G}_b$.

Fall 2: $a \neq b$. Sei $\mathcal{G}'$ der Graph, den man aus $\mathcal{G}$ erhält, indem man zuerst alle mit a oder b inzidenten Links löscht und dann a und b selbst. Man überlegt sich leicht, daß gilt: $\mathcal{G}_a \xrightarrow[\pi]{b} \mathcal{G}'$ und $\mathcal{G}_b \xrightarrow[\pi]{a} \mathcal{G}'$. [ ]

Wir bezeichnen die π-Normalform eines Klauselgraphen $\mathcal{G}$, deren Existenz und Eindeutigkeit durch den obigen Satz gewährleistet ist, mit $\underline{\pi(\mathcal{G})}$. Wir zeigen nun, daß die *konstruktiv* definierte π-Normalform $\pi(\mathcal{G})$ stets der *algebraisch* definierte, größte π-freie Subgraph von $\mathcal{G}$ ist.

2.2.4 <u>Satz</u>. Sei $\mathcal{G}$ ein Klauselgraph. Dann ist $\pi(\mathcal{G})$ der größte π-freie Subgraph von $\mathcal{G}$.

Beweis: Sei $\mathcal{G}$ ein Klauselgraph. $\pi(\mathcal{G})$ ist offensichtlich ein π-reduzierter Subgraph von $\mathcal{G}$. Wenn $\mathcal{G}$ π-reduziert ist, dann ist $\pi(\mathcal{G})=\mathcal{G}$ trivialerweise der größte π-reduzierte Subgraph von $\mathcal{G}$. Sei $\mathcal{G}$ also nicht π-reduziert und sei $\mathcal{G}'$ ein beliebiger π-reduzierter Subgraph von $\mathcal{G}$. Wir haben zu zeigen, daß $\mathcal{G}' \subset \pi(\mathcal{G})$ ist. Es gibt $\mathcal{G}_1,\ldots,\mathcal{G}_n$ und $a_1,\ldots,a_{n-1}$ mit $n \geq 2$, so daß

$$\mathcal{G}=\mathcal{G}_1 \xrightarrow[\pi]{a_1} \mathcal{G}_2 \xrightarrow[\pi]{a_2} \ldots \xrightarrow[\pi]{a_{n-1}} \mathcal{G}_n = \pi(\mathcal{G}).$$

Angenommen, $\mathcal{G}' \not\subset \pi(\mathcal{G})$. Dann gibt es ein i mit $\mathcal{G}' \subset \mathcal{G}_i$ und $\mathcal{G}' \not\subset \mathcal{G}_{i+1} = \mathcal{G}_i - a_i$. Das bedeutet, daß a_i in $\mathcal{G}'$ liegt und in $\mathcal{G}_i$ isoliert ist. Also muß a_i auch in $\mathcal{G}'$ isoliert sein. W! [ ]

Die π-Normalform des Graphen (d) in 2.1 ist (), die des Graphen (e) ist (□). Der Übergang zum größten π-reduzierten Subgraphen kann also eine sehr starke Reduktion sein, die unter Umständen ein völliges kollabieren eines unerfüllbaren Graphen in den erfüllbaren, leeren Graphen () bewirkt.

2.3 Die Resolutionsregel

Jeder Link eines Klauselgraphen verbindet zwei potentiell komplementäre Literale und stellt damit eine Resolutionsmöglichkeit dar. Obwohl man im Prinzip auch auf Autolinks resolvieren könnte (sogenannte *Selbstresolution*), ist die Resolution beim KOWALSKI-Kalkül nur auf *Non-Autolinks* zugelassen. Da die durch einen Non-Autolink $\ell = L^a K^b$ verbundenen Klauseln variablendisjunkt sind, existiert stets die binäre Resolvente $\mathrm{Res}(C_a, L; C_b, K)$. Die *erzeugende Komponente* der Klauselgraphresolutionsregel bildet diese Resolvente zu ℓ und fügt sie nach einer Variablenumbennenung als neuen Knoten in den Graphen ein. Danach werden die Links an die Resolvente durch *Vererbung* ausschließlich aus den Links an die Elternknoten gebildet. Die *reduzierende Komponente* der Klauselgraphresolutionsregel löscht zunächst ℓ und führt dann eine π-Reduktion des Graphen durch. Das Löschen des *Resolutionslinks* ℓ bedeutet weit mehr als die Unterdrückung einer Wiederholung der Resolution auf ℓ. Zum einen trägt das Löschen von ℓ zur Isolierung seiner inzidenten Literalauftreten bei. Da nach jedem Ableitungsschritt π-reduziert wird, können dadurch schließlich weitere Links gelöscht werden, auf denen noch nicht resolviert wurde. Zum anderen verstärkt der Vererbungsmechanismus die Löschwirkung von ℓ: Bei späteren Resolutionen kann ℓ nicht mehr vererbt werden, so daß das Löschen von ℓ das "Löschen", d.h. die Nichtbildung, vieler zukünftiger Links nach sich zieht.

2.3.1 Algorithmus ρ_o

Eingabe: $\mathcal{G}_o$: Klauselgraph;

$\ell = L_o^a K_o^b$ Link in $\mathcal{G}_o$ mit $a \neq b$;

Konstanten: $\sigma := \mathrm{nau}(L_o, \bar{K}_o)$;

θ := U-Substitution für $\mathrm{Res}(C_a, L_o; C_b, K_o)$, so daß $\mathcal{G}_o$ und $\theta\mathrm{Res}(C_a, L_o; C_b, K_o)$ variablendisjunkt sind;

c := Knoten, der in $\mathcal{G}_o$ nicht vorkommt;

Variablen: $\mathcal{G}$: Klauselgraph := $\mathcal{G}_o$;

L, K, M : $\mathbb{L}$; d, e : $\mathbb{K}$;

1: $\mathcal{G} := \mathcal{G} + \langle c, \theta\mathrm{Res}(C_a, L_o; C_b, K_o)\rangle$;

2: Für alle $L \in C_c$

2.1: Wähle K, d mit $d \in \{a,b\} \wedge K \in C_d \wedge L = \theta\sigma K$;

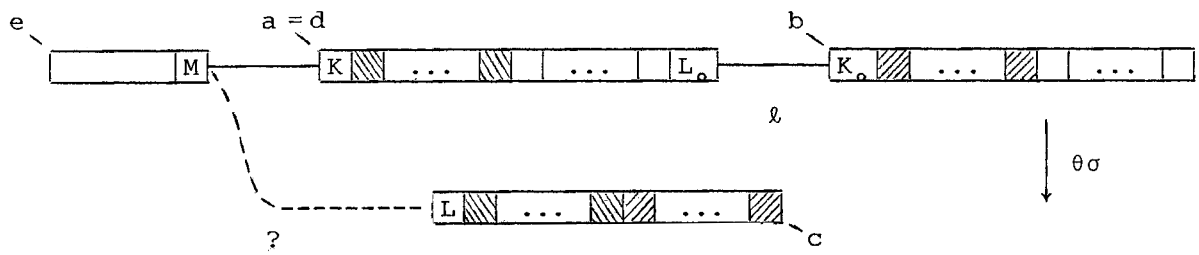
2.2: Für alle M, e mit $M^e K^d$ ist Link in $\mathcal{G}$ und M und L sind potentiell komplementär

2.2.1: $\mathcal{G} := \mathcal{G} + M^e L^c$;

Ausgabe: $\mathcal{G}$;

Nachbedingung: $\mathcal{G}_o = \mathcal{G} - c$.

Die folgende Skizze veranschaulicht die Funktion von ρ_o :

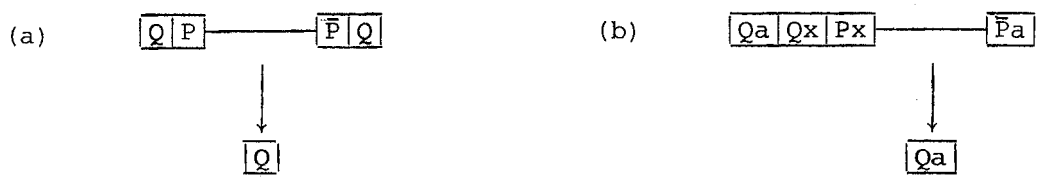


2.3.2 Die Linkvererbung. Der Hauptgedanke bei der Klauselgraphresolutionsregel ist, beim Einbinden der Resolvente die aufwendige Suche nach potentiell komplementären Literalauftreten *irgendwo* im Graphen zu vermeiden.

Jedes Literal der Resolvente wird aus einem oder mehreren Literalen der Elternklauseln gebildet. Wir bezeichnen diese Literale in den Elternklauseln als

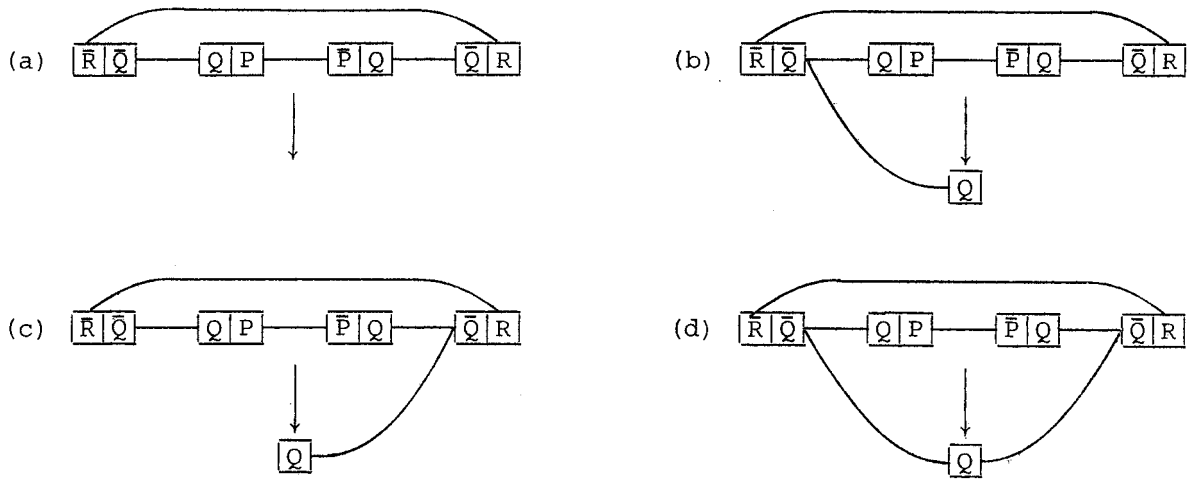
Vorgänger des Literals in der Resolvente. Ein Resolventenliteral mit mehr als einem Vorgänger entsteht durch Verschmelzung (engl: merging) seiner Vorgänger. Das *Einbinden der Resolvente* geht nun wie folgt vor sich: In Schritt 2.1 von ρ_o wird zu jedem Literal L in der Resolvente indeterministisch ein Vorgänger K in einem der Elternknoten, der mit d bezeichnet wird, gewählt. Dieser willkürlich ausgewählte Vorgänger heißt signifikanter Vorgänger. Ein Literal L der Resolvente wird nur dann mit einem potentiell komplementären Literalauftreten M^e durch einen Link verbunden, wenn M^e bereits mit dem signifikanten Vorgänger K^d verbunden ist. Wir sagen, daß der Link M^eL^c an die Resolvente durch Vererbung aus dem Link M^eK^d an den Elternknoten entstanden ist. Wir sagen weiter, daß M^eK^d vererbt wurde zu M^eL^c und daß M^eL^c von M^eK^c vererbt wurde. Vererbt werden kann der Link M^eK^d immer dann, wenn die potentielle Komplementarität durch den Übergang $K \longrightarrow L = \theta\sigma K$ nicht verloren gegangen ist. Da bei aussagenlogischen Klauselgraphen stets $\theta\sigma K = K$ gilt, können hier immer alle Links vererbt werden.

Wir geben nun zwei Beispiele an, in denen ein Literal der Resolvente mehr als zwei Vorgänger hat:



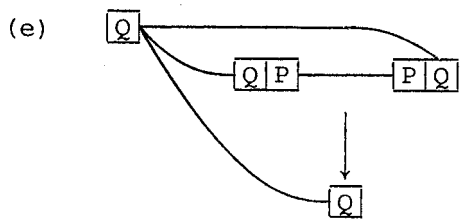
In (b) hat das Resolventenliteral Qa zwei verschiedene Vorgänger in derselben Elternklausel. Dieser Fall kann bei aussagenlogischen Klauselgraphen nicht vorkommen. Dort können nur, wie in (a), zwei Literale verschmelzen, die zu verschiedenen Elternklauseln gehören.

2.3.3 Mono- und Multivererbung. In ρ_o kommen nur Links an den signifikanten Vorgänger eines Resolventenliterals für die Vererbung in Frage. Links an andere Vorgänger bleiben unberücksichtigt. Wir sprechen daher von Monovererbung. Dagegen gibt Robert KOWALSKI in /Ko75/ und /Ko79/ eine Resolutionsregel an, die mit Multivererbung arbeitet. Dabei wird für jeden Vorgänger versucht, seine inzidenten Links an die Resolvente zu vererben. Die Vererbungsmechanismen von Wolfgang BIBEL in /Bi81a/ und /Bi81c/ erzeugen mehr Links als bei Mono- und weniger Links als bei Multivererbung. Das folgende Beispiel zeigt die Unterschiede zwischen Mono- und Multivererbung:



Die Graphen (b) und (c) enstehen durch Anwendung von ρ_o auf den Graphen (a). Bei (b) wurde der signifikante Vorgänger des Resolventenliterals im linken Elternknoten gewählt, während er bei (c) rechts gewählt wurde. Der Indeterminismus bei der Wahl des signifikanten Vorgängers ist also wesentlich, d.h. je nach Wahl erhält man verschiedene Graphen als Ergebnis. Graph (d) entsteht aus (a) durch Anwendung einer Resolutionsregel mit Multivererbung. Die Links beider Vorgänger werden an die Resolvente vererbt.

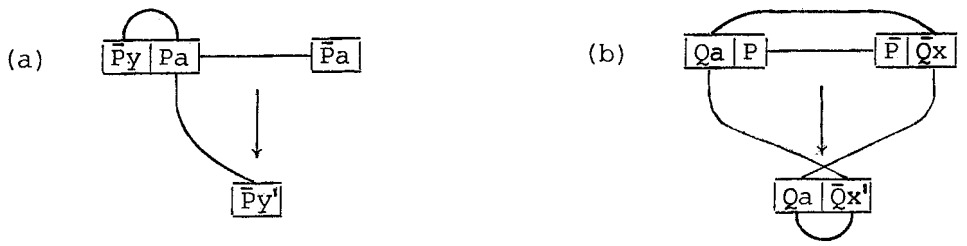
Monovererbung hat gegenüber Multivererbung zwei Vorteile. Zum einen werden weniger Links erzeugt, wie die obigen Beispiele zeigen. Zum anderen erzeugt ρ_o jeden Link nur einmal. Dagegen wird bei Multivererbung ein Link an ein Resolventenliteral mit mehreren Vorgängern u.U. mehrfach gebildet, nämlich für jeden Vorgänger einmal, wie das folgende Beispiel demonstriert:



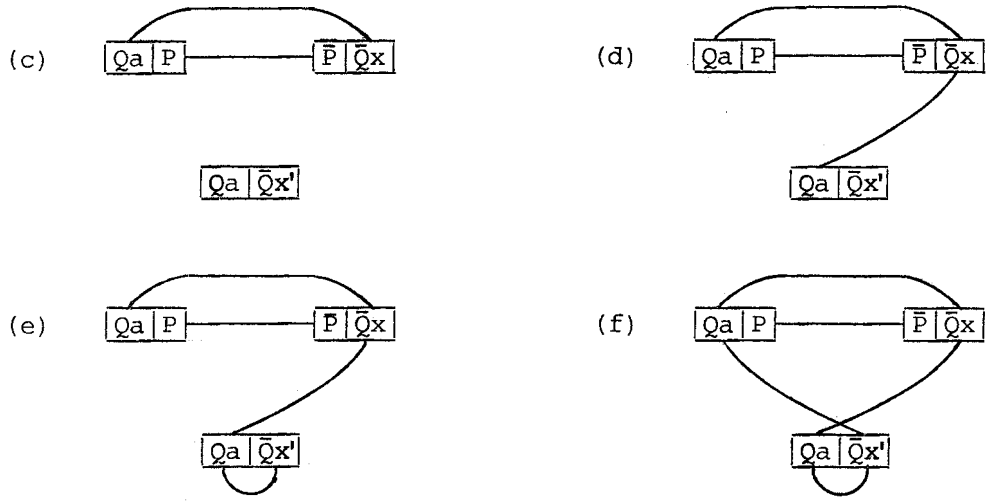
Monovererbung erspart also bei einer Implementierung die unter Umständen aufwendige Abfrage, ob ein Link schon gezogen ist.

2.3.4 Lokale Links. Links, die die Resolvente mit einem Elternknoten verbinden, heißen lokal. Zwei Links heißen parallel, wenn sie dieselben Knoten verbinden. Der Leser überlege sich, daß lokale Links *nur* durch Vererbung von Autolinks oder von Links, die zum Resolutionslink parallel sind, entstehen

können. Die folgenden Beispiele zeigen die beiden Möglichkeiten:



In (a) wird der Autolink des linken Elternknotens an die Resolvente vererbt. Beispiel (b) zeigt, wie subtil der Vererbungsmechanismus arbeitet: Aus dem zum Resolutionslink parallelen Link $Qa\bar{Q}x$ werden insgesamt drei Links an die Resolvente gebildet. Dies geschieht im einzelnen wie folgt: Nehmen wir an, daß in der Schleife 2 von ρ_0 die Resolventenliterale in der Reihenfolge Qa, $\bar{Q}x'$ abgearbeitet werden. Dann ergeben sich die folgenden Schnappschüsse:



Als Vorgänger für Qa in der Resolvente kommt nur Qa im linken Elternknoten in Frage. An dieses Qa führt nur der parallele Link. Wenn man ihn vererbt, erhält man die Situation (d).

Der Vorgänger für das Literal Qx' in der Resolvente ist Qx im rechten Elternknoten. An diesen Vorgänger führen in der aktuellen Situation (d) zwei Links. Die Vererbung des lokalen Links ergibt einen Autolink und die Situation (e). Die Vererbung des parallelen Links ergibt einen weiteren lokalen Link und die finale Situation (f).
Bearbeitet man die Resolventenliterale in der umgekehrten Reihenfolge, bekommt man gespiegelte Zwischenzustände und denselben Finalzustand.

Bei Anwendung von ρ_o auf einen AT-Klauselgraphen können, falls die Resolvente *keine* Tautologie ist, keine lokalen Links entstehen. Das liegt daran, daß es in AT-Klauselgraphen keine Autolinks gibt und die Resolventen zu zwei verschiedenen, parallelen Links stets Tautologien sind.

2.3.5 Die Indeterminismen in ρ_o. Der Algorithmus ρ_o ist an den folgenden Stellen indeterministisch:

(a) Bestimmung von Θ.

(b) Bestimmung von c.

(c) Reihenfolge bei Schritt 2.

(d) Wahl des signifikanten Vorgängers in Schritt 2.1.

(e) Reihenfolge bei Schritt 2.2.

Der Leser überlege sich, daß die Indeterminismen (c) und (e) unwesentlich sind, d.h., daß sie keinen Einfluß auf den finalen Graphen haben. Die Indeterminismen in (a) und (b) beeinflussen den finalen Graphen zwar, aber nur unwesentlich (im Sinne von 2.1.10) durch verschiedene Knoten- und Variablenbenennungen. Der einzige wesentliche Indeterminismus in ρ_o ist (d), also die Wahl des signifikanten Vorgängers. Wenn ein Literal der Resolvente zwei Vorgänger hat, die sich in den inzidenten Links unterscheiden, dann wird der finale Graph je nach Wahl des signifikanten Vorgängers verschieden ausfallen, wie die Graphen (a), (b) und (c) in 2.3.3 zeigen. Man mache sich klar, daß bei einer Wiederholung des Übergangs $\mathcal{G} \xrightarrow[\rho_o]{\ell} \mathcal{G}'$ selbst bei gleichem (!) Resultat $\mathcal{G}'$ die signifikanten Vorgänger anders gewählt sein können.

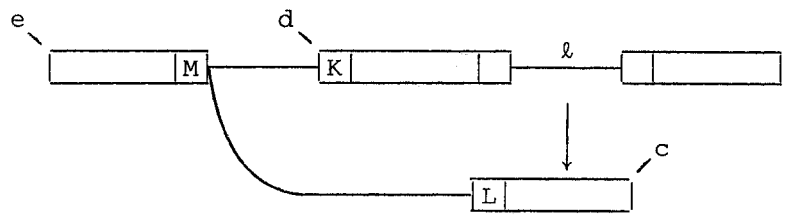
Durch Vorgabe der signifikanten Vorgänger läßt sich ρ_o im wesentlichen deterministisch machen. In einer Implementierung kann man etwa solche Literalauftreten als signifikante Vorgänger wählen, bei denen die Anzahl der inzidenten Links minimal ist. Wir machen hier keine weiteren Aussagen zur Wahl der signifikanten Vorgänger, da diese für die von uns untersuchten Fragen, wie Konfluenz und Vollständigkeit, ohne Bedeutung ist.

Das folgende Lemma gibt eine weitere Nachbedingung für ρ_o an, die die Links an die Resolvente unabhängig von der Vererbungsschleife 2 in ρ_o charakterisiert. In späteren Beweisen werden wir oft auf dieses Lemma zurückgreifen. Wir schreiben $\mathcal{G} \xrightarrow[\rho_o]{\ell} \mathcal{G}'$, wenn $\langle \mathcal{G}, \ell \rangle$ eine korrekte Eingabe für ρ_o ist, und wenn $\mathcal{G}'$ eine mögliche Ausgabe zu dieser Eingabe ist.

2.3.6 <u>Lemma</u>. Sei $\mathcal{G} \xrightarrow[\rho_o]{\ell} \mathcal{G}'$. Sei L^c ein Literalauftreten in der Resolvente zu ℓ und sei K^d der beim obigen Übergang gewählte signifikante Vorgänger zu L^c. Dann gilt für jedes Literalauftreten M^e in $\mathcal{G}'$, das zu L^c potentiell komplementär ist:

$$M^eL^c \text{ ist Link in } \mathcal{G}' \iff M^eK^d \text{ ist Link in } \mathcal{G}'.$$

Beweis. Man mache sich zunächst die Aussagen des Satzes an der folgenden Skizze klar:



Dabei ist zu beachten, daß der Knoten e mit jedem der drei anderen Knoten identisch sein kann.

Die statische Aussage des Satzes muß für den Beweis mit dem dynamischen Ablauf der Vererbungsschleife von ρ_o in Zusammenhang gebracht werden. Im Beweis legen wir einen beliebigen, aber fest gewählten Ablauf $\mathcal{G} \xrightarrow[\rho_o]{\ell} \mathcal{G}'$ zugrunde, bei dem K^d als signifikanter Vorgänger für L^c gewählt wurde.

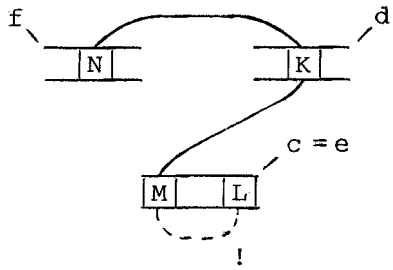
"==>". Sei M^eL^c Link in $\mathcal{G}'$. M^eL^c muß in Schritt 2.2.1 von ρ_o im $\langle L,K^d,M^e\rangle$-Durchlauf eingefügt worden sein. Dafür ist Voraussetzung, daß der Link M^eK^d bereits existierte. Da ρ_o keine Links löscht, existiert M^eK^d auch noch in $\mathcal{G}'$.

"<==". Sei M^eK^d Link in $\mathcal{G}'$.

Fall 1: M^eK^d ist Link in $\mathcal{G}$. Da K^d der signifikante Vorgänger zu L^c ist, und nach Voraussetzung M und L potentiell komplementär sind, wird M^eL^c jedenfalls in Schritt 2.2 von ρ_o erzeugt. Damit ist M^eL^c Link in $\mathcal{G}'$.

Fall 2: M^eK^d ist nicht Link in $\mathcal{G}$. Dann ist $M^{e-d}K$ ein lokaler Link d.h. e = c. Wir haben zu zeigen, daß der Autolink M^cL^c der Resolvente von ρ_o erzeugt wird. Sei N^f der signifikante Vorgänger zu M^c. Da M^cK^d nach Voraussetzung Link in $\mathcal{G}'$ ist, liefert der "==>"-Teil des Beweises, daß auch N^fK^d Link in $\mathcal{G}'$ ist. Da N^f und K^d signifikante Vorgänger sind, sind f und d beide Inzidenzknoten von ℓ. D.h., daß N^fK^d ein zu ℓ paralleler Link oder ein Autolink eines der Inzidenzknoten von ℓ ist. Also existiert der Link N^fK^d bereits in $\mathcal{G}$. Graphisch

läßt sich die vorliegende Situation wie folgt darstellen:



Fall 2a: In Schritt 2 von ρ_o wird zuerst M und dann L bearbeitet. Dann existiert, wenn die Links an L^c erzeugt werden, bereits der Link M^cK^d. Da M und L nach Voraussetzung potentiell komplementär sind und K^d der signifikante Vorgänger zu L^c ist, wird der Autolink M^cL^c erzeugt.
Fall 2b: In Schritt 2 von ρ_o wird zuerst L und dann M bearbeitet. Da M und L nach Voraussetzung potentiell komplementär sind, gilt dies auch für N und L (Lemma 1.2.14). Also wird bei der Bearbeitung von L der lokale Link N^fL^c gebildet. Bei der nachfolgenden Bearbeitung von M wird dieser lokale Link zum Autolink M^cL^c vererbt. []

Der Vererbungsmechanismus von ρ_o soll die Suche nach potentiell komplementären Literalauftreten ersparen. Wir zeigen nun, daß er *vollständig* ist, d.h., daß er totale Graphen in totale Graphen überführt.

2.3.7 <u>Satz</u>. Sei $\mathcal{G} \xrightarrow[\rho_o]{\ell} \mathcal{G}'$. Dann ist, wenn $\mathcal{G}$ total ist, auch $\mathcal{G}'$ total.

Beweis. Sei $\mathcal{G} \xrightarrow[\rho_o]{\ell} \mathcal{G}'$ und sei $\mathcal{G}$ total. Sei L^c ein Literalauftreten in der Resolvente zu ℓ und M^e ein Literalauftreten in $\mathcal{G}'$, welches potentiell komplementär zu L^c ist. Wir müssen zeigen, daß M^eL^c ein Link in $\mathcal{G}'$ ist.
Zu jedem Literal der Resolvente sei der signifikante Vorgänger durch einen Übergang wie oben festgelegt. Sei K^d der signifikante Vorgänger von L^c. Wegen Lemma 2.3.6 genügt es zu zeigen, daß M^eK^d Link in $\mathcal{G}'$ ist. Lemma 1.2.14 liefert, daß M und K potentiell komplementär sind, da M und L dies nach Voraussetzung sind.

Fall 1: $e \neq c$. Dann sind M^e und K^d potentiell komplementäre Literalauftreten in $\mathcal{G}$. Da $\mathcal{G}$ total ist, ist M^eK^d Link in $\mathcal{G}$. Da $\mathcal{G}$ ein Subgraph von $\mathcal{G}'$ ist, ist M^eK^d, wie zu zeigen war, ein Link in $\mathcal{G}'$.

Fall 2: e = c. Sei N^f der signifikante Vorgänger zu M^e Da M und K potentiell komplementär sind, sind auch N und K potentiell komplementär. Als signifikante Vorgänger sind N^f und K^d beide Literalauftreten in $\mathcal{G}$. Da $\mathcal{G}$ total ist, existiert der Link $N^f K^d$ in $\mathcal{G}$ und damit, da $\mathcal{G} \subset \mathcal{G}'$, auch in $\mathcal{G}'$. Lemma 2.3.6 liefert, daß $M^e K^d$ Link in $\mathcal{G}'$ ist, was zu zeigen war. []

Die Klauselgraphresolutionsregel ist der folgende Algorithmus ρ:

2.3.8 <u>Algorithmus ρ</u>

Eingabe:	$\mathcal{G}_0$	: Klauselgraph;
	ℓ	: Non-Autolink in $\mathcal{G}_0$;
Variable:	$\mathcal{G}$	: Klauselgraph := $\mathcal{G}_0$;
1:	$\mathcal{G}$	:= $\rho_0(\mathcal{G}, \ell)$;
2:	$\mathcal{G}$	:= $\pi(\mathcal{G} - \ell)$;
Ausgabe:	$\mathcal{G}$.	

Man beachte, daß $\rho_0(\mathcal{G}, \ell)$ nicht die Anwendung einer Funktion ρ_0 auf $\mathcal{G}$ und ℓ darstellt. Vielmehr handelt es sich um den Aufruf eines indeterministischen Unteralgorithmus. $\rho_0(\mathcal{G}, \ell)$ ist also kein eindeutig bestimmter Klauselgraph. Dagegen ist $\pi(\mathcal{G} - \ell)$ in Schritt 2 eindeutig bestimmt.

Das Löschen des Resolutionslinks und die abschließende π-Reduktion können zu erheblichen Reduktionen führen. Als Beispiel betrachte man den unerfüllbaren, π-reduzierten Graphen (a) in 2.3.3 . Wenn man die ρ-Regel auf den mittleren Link $P\bar{P}$ anwendet, erhält man, unabhängig von der Wahl des signifikanten Vorgängers, den leeren (!) Klauselgraphen. Die ρ-Regel kann also schon im aussagenlogischen Fall, durch den Schneeballeffekt bei der π-Reduktion, große Teile des Graphen löschen. Das obige Beispiel zeigt auch eindrucksvoll, *daß die Reduktionswirkung bei Monovererbung erheblich größer als bei Multivererbung ist:* Da bei Multivererbung der erzeugende Teil der Resolutionsregel den Graphen (d) liefert, löscht der reduzierende Teil lediglich die beiden Elternknoten. Man kann zeigen, daß bei AT-Klauselgraphen und bei Multivererbung generell höchstens die beiden Elternknoten gelöscht werden können, d.h., daß grundsätzlich kein Schneeballeffekt auftreten kann.

Das Löschen des Resolutionslinks hat viel größere Konsequenzen als das bloße Unterdrücken einer Wiederholung der Resolution. Insbesondere kann ein gelöschter Link bei späteren Resolutionen nicht mehr vererbt werden. Der Vererbungsmechanismus *potenziert* also die Löschwirkung eines einzelnen Links.

2.4 Die Faktorisierungsregel

Mit der Faktorisierungsregel für Klauselgraphen können zu einem Knoten die echten Faktoren gebildet und in den Graphen eingebunden werden. Dabei werden die Links an den Faktor mit der gleichen Vererbungsmethode wie bei der Resolutionsregel erzeugt.

2.4.1 Algorithmus ϕ

Eingabe: $\mathcal{G}_o$: Klauselgraph;
a : Knoten in $\mathcal{G}_o$;
σ : F-Substitution für C_a mit $\sigma \neq \varepsilon$;

Konstanten: θ := U-Substitution für σC_a, so daß $\mathcal{G}_o$ und $\theta\sigma C_a$ variablendisjunkt sind;
c := Knoten, der in $\mathcal{G}_o$ nicht vorkommt;

Variablen: $\mathcal{G}$: Klauselgraph := $\mathcal{G}_o$;
L, K, M : $\mathbb{L}$; e : $\mathbb{K}$;

1: $\mathcal{G} := \mathcal{G} + \langle c, \theta\sigma C_a \rangle$;

2: Für alle $L \in C_a$

2.1: Wähle K mit $K \in C_a$ und $L = \theta\sigma K$;

2.2: Für alle M, e
mit $M^e K^a$ ist Link in $\mathcal{G}$
und M und L sind potentiell komplementär

2.2.1: $\mathcal{G} := \mathcal{G} + M^e L^c$;

Ausgabe: $\pi(\mathcal{G})$.

Wir schreiben $\mathcal{G} \xrightarrow[\phi]{a,\sigma} \mathcal{G}'$, wenn $\langle \mathcal{G}, a, \sigma \rangle$ eine korrekte Eingabe für ϕ ist, und wenn $\mathcal{G}'$ eine mögliche Ausgabe zu dieser Eingabe ist. Wenn $\mathcal{G} \xrightarrow[\phi]{a,\sigma} \mathcal{G}'$ und $\mathcal{G}$ π-reduziert ist, dann ist offensichtlich $\mathcal{G} \subset \mathcal{G}'$, d.h., bei der abschließenden π-Reduktion kann dann höchstens der neugebildete Faktorknoten gelöscht werden.

2.4.2 Lemma. Sei $\mathcal{G} \xrightarrow[\phi]{a,\sigma} \mathcal{G}'$, $\mathcal{G} \neq \mathcal{G}'$ und sei $\mathcal{G}$ π-reduziert. Sei L^c ein Literalauftreten im Faktor zu a,σ und sei K^a der beim obigen Übergang gewählte sig-

nifikante Vorgänger zu L^c. Dann gilt für jedes Literalauftreten M^e in $\mathcal{G}'$, das zu L^c potentiell komplementär ist:

$$M^e L^c \text{ ist Link in } \mathcal{G}' \iff M^e K^a \text{ ist Link in } \mathcal{G}'.$$

Beweis. Analog zu Lemma 2.3.6. []

2.4.2 <u>Satz</u>. Sei $\mathcal{G} \xrightarrow[\phi]{a,\sigma} \mathcal{G}'$ Dann ist, wenn $\mathcal{G}$ total ist, auch $\mathcal{G}'$ total.

Beweis. Analog zu Satz 2.3.7. []

Die Faktorisierungsregel läßt sich nicht auf aussagenlogische Graphen anwenden, da zwei variablenfreie Literale nur dann unifizierbar sind, wenn sie gleich sind.
In den von uns untersuchten Spezialfällen (aussagenlogische und unäre Klauselmengen) ist die Faktorisierungsregel ohne Bedeutung. Für den allgemeinen Fall wissen wir vom Resolutionskalkül, daß die Faktorisierungsregel erforderlich ist. Da es jedoch bisher kein einziges Vollständigkeitsresultat für diesen allgemeinen Fall gibt, sind alle Aussagen über die Form und die Anwendungsbedingungen der Faktorisierungsregel rein spekulativ. So stellt auch unsere Version nur einen Vorschlag dar.

2.5 Ableitungen

2.5.1 Ableitungsregeln. Eine Ableitungsregel für Klauselgraphen ist ein indeterministischer Algorithmus, der Klauselgraphen in Klauselgraphen überführt. Der KOWALSKI-Kalkül besteht aus insgesamt vier Ableitungsregeln, nämlich der ρ-, der ϕ-, der τ- und der σ-Regel. Die τ- und die σ-Regel führen wir erst in den Abschnitten 2.7 und 2.8 ein, da wir zur Diskussion der mit ihnen verbundenen Probleme auf Begriffe zurückgreifen wollen, die wir in diesem und im nächsten Abschnitt definieren. Während ρ und ϕ primär erzeugende Regeln sind, d.h. neue Elemente in den Graphen einfügen, sind τ und σ rein reduzierende Regeln, die lediglich Knoten löschen und damit den Graphen in einen seiner Subgraphen überführen. Alle vier Regeln des KOWALSKI-Kalküls erzeugen π-reduzierte Graphen. Wir schreiben:

- $\mathcal{G} \xrightarrow[\alpha]{x_1,\ldots,x_n} \mathcal{G}'$, wenn $\alpha \in \{\rho,\phi,\sigma,\tau\}$ und $\langle\mathcal{G},x_1,\ldots,x_n\rangle$ eine korrekte Eingabe für α ist, für die $\mathcal{G}'$ eine mögliche Ausgabe ist.
- $\mathcal{G} \xrightarrow[\alpha]{} \mathcal{G}'$, wenn es $x_1,\ldots,x_n$ mit $\mathcal{G} \xrightarrow[\alpha]{x_1,\ldots,x_n} \mathcal{G}'$ gibt.
- $\mathcal{G} \xrightarrow[\alpha,\beta]{} \mathcal{G}'$, wenn $\alpha \neq \beta$ und $\mathcal{G} \xrightarrow[\alpha]{} \mathcal{G}'$ oder $\mathcal{G} \xrightarrow[\beta]{} \mathcal{G}'$ gilt.
- $\mathcal{G} \longrightarrow \mathcal{G}'$, wenn $\mathcal{G} \xrightarrow[\rho,\phi,\sigma,\tau]{} \mathcal{G}'$ gilt.

2.5.2 Ableitungsschritte. Ein Tupel $\Sigma = \langle\mathcal{G},x_1,\ldots,x_n,\alpha,\mathcal{G}'\rangle$ heißt Ableitungsschritt, wenn $\mathcal{G} \xrightarrow[\alpha]{x_1,\ldots,x_n} \mathcal{G}'$ gilt und $\square \notin S(\mathcal{G})$ ist.

2.5.3 Ableitungen. Eine Folge $\Gamma = \Sigma_1 \ldots \Sigma_n$, $n \geq 0$ von Ableitungsschritten Σ_i: $\mathcal{G}_i \xrightarrow[\alpha]{} \mathcal{G}_i'$ heißt Ableitung (im KOWALSKI-Kalkül), wenn gilt:

(a) Für alle i mit $1 < i \leq n$ ist $\mathcal{G}_{i-1}' = \mathcal{G}_i$.

(b) Die ϕ-Regel wird in Γ nicht auf Knoten angewendet, die in Γ selbst durch die ϕ-Regel erzeugt wurden.

(c) Für zwei Faktorisierungsschritte Σ_i: $\mathcal{G}_i \xrightarrow[\phi]{a,\sigma} \mathcal{G}_i'$ und Σ_j: $\mathcal{G}_j \xrightarrow[\phi]{a,\sigma} \mathcal{G}_j'$ in Γ mit $i < j$ existiert stets ein k mit $i < k < j$, so daß a kein Knoten in $\mathcal{G}_k$ ist.

Die leere Ableitung ist die leere Folge von Ableitungsschritten. Die Bedingungen (b) und (c) erzwingen, daß in Γ kein Faktor mehrfach gebildet wird. Damit wird die ϕ-Regel der ρ-Regel gleichgestellt, bei der eine Wiederholung einer Resolution, da der Resolutionslink gelöscht wird, von vornherein unmög-

lich ist. In Bedingung (c) müssen wir die Existenz eines Graphen $\mathcal{G}_k$, der d nicht enthält, verlangen, weil im Verlauf von Γ ein Knoten erzeugt werden kann, der dann wieder gelöscht wird und schließlich erneut, i.a. mit einer anderen Klausel markiert, durch die ρ- oder die ϕ-Regel eingebracht wird.

2.5.4 Bezeichnungen. Sei Γ: $\mathcal{G}_0 \longrightarrow \mathcal{G}_1 \longrightarrow \ldots \longrightarrow \mathcal{G}_n$ eine nichtleere Ableitung. Dann heißt

- $\mathcal{G}_0$ der Initialgraph von Γ.
- $\mathcal{G}_n$ der Finalgraph von Γ.
- $\mathcal{G}_i$ ein Zwischengraph von Γ, wenn $0 \leq i \leq n$ gilt.

Die Anzahl der Ableitungsschritte in einer Ableitung Γ heißt die Länge von Γ und wird mit $|\Gamma|$ bezeichnet. Die leere Ableitung hat also die Länge null. Eine Ableitung Γ heißt fortsetzbar, wenn es eine nichtleere Ableitung Γ' gibt, so daß die Konkatenation $\Gamma\Gamma'$ eine Ableitung ist. Offensichtlich ist eine nichtleere Ableitung Γ genau dann nicht fortsetzbar, wenn der Finalgraph von Γ entweder die leere Klausel enthält, oder der leere Graph ist. Dies folgt sofort aus der Tatsache, daß der Finalgraph einer nichtleeren Ableitung stets π-reduziert ist, da jede Anwendung einer Ableitungsregel des KOWALSKI-Kalküls von einer π-Reduktion begleitet wird.

2.5.5 Schreibweisen. Wir schreiben:

- Γ: $\mathcal{G} \vdash\!\!- \mathcal{G}'$, wenn Γ eine Ableitung mit dem Initialgraphen $\mathcal{G}$ und dem Finalgraphen $\mathcal{G}'$ ist, oder wenn Γ die leere Ableitung und $\mathcal{G} = \mathcal{G}'$ ist.
- Γ: $\mathcal{G} \vdash\!\!- \mathcal{G}' \vdash\!\!- \mathcal{G}''$, wenn Γ: $\mathcal{G} \vdash\!\!- \mathcal{G}''$ gilt und $\mathcal{G}'$ ein Zwischengraph von Γ ist, oder wenn Γ die leere Ableitung und $\mathcal{G} = \mathcal{G}' = \mathcal{G}''$ ist.
- $\mathcal{G} \vdash\!\!- \mathcal{G}'$, wenn eine Ableitung Γ mit Γ: $\mathcal{G} \vdash\!\!- \mathcal{G}'$ existiert.
- $\mathcal{G} \vdash\!\!- \mathcal{G}' \vdash\!\!- \mathcal{G}''$, wenn eine Ableitung Γ mit Γ: $\mathcal{G} \vdash\!\!- \mathcal{G}' \vdash\!\!- \mathcal{G}''$ existiert.

Der Leser mache sich klar, daß die Ableitungsrelation $\vdash\!\!-$ des KOWALSKI-Kalküls zwar reflexiv, aber wegen den Bedingungen (b) und (c) in 2.5.3 nicht transitiv ist. Aus diesem Grund benutzen wir statt $\xrightarrow{*}$ (bezeichnet die *transitive* Hülle von $\longrightarrow$) das neue Symbol $\vdash\!\!-$.

2.5.6 Widerlegungen. Eine Ableitung Γ heißt Widerlegung für einen Klauselgraphen $\mathcal{G}$, wenn $\mathcal{G}$ der Initialgraph zu Γ ist und der Finalgraph von Γ die leere Klausel enthält. Ein Klauselgraph heißt widerlegbar, wenn eine Widerlegung

für ihn existiert. Wir schreiben:

- $\Gamma : \mathcal{G} \vdash \square$, wenn Γ eine Widerlegung für $\mathcal{G}$ ist.

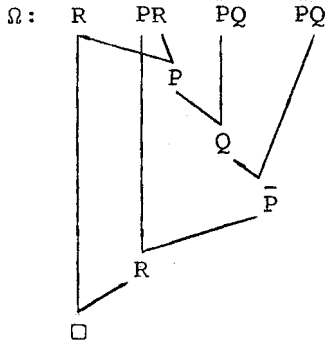
  $\mathcal{G} \vdash \square$, wenn $\mathcal{G}$ widerlegbar ist.

2.5.7 <u>Die induzierten Resolutionsableitungen</u>. Sei Γ: $\mathcal{G} \vdash \mathcal{G}'$ eine Ableitung. Sei $S(\mathcal{G}) = \{C_1,\ldots,C_n\}$ und seien $D_1,\ldots,D_k$ die in Γ erzeugten Resolventen in der Reihenfolge ihrer Erzeugung. Dann gibt es stets passende Varianten $D_1',\ldots,D_k'$ von $D_1,\ldots,D_k$, so daß $\Gamma_R := C_1\ldots C_n D_1'\ldots D_k'$ eine Resolutionsableitung (siehe 1.4.7) ist. Γ_R heißt induzierte Resolutionsableitung zu Γ. Offensichtlich ist Γ genau dann eine Widerlegung für $\mathcal{G}$, wenn Γ_R eine Resolutionswiderlegung für $S(\mathcal{G})$ ist.

2.5.8 <u>Beispiel</u>. Sei $S = \{\bar{R}, PR, \bar{P}Q, \bar{P}\bar{Q}\}$. Γ sei die Widerlegung:

$$\mathcal{G}_S = \left|\begin{array}{ccc} & P & \bar{P} \quad \bar{P} \\ & & Q \quad \bar{Q} \\ \bar{R} & R & \end{array}\right| \xrightarrow[\rho]{\bar{R}^1 R^2} \left|\begin{array}{ccc} \bar{P} & \bar{P} & P \\ Q & \bar{Q} & \\ & & \end{array}\right| \xrightarrow[\rho]{Q^1 \bar{Q}^2} |P \quad \bar{P}| \xrightarrow[\rho]{P^1 \bar{P}^2} (\square).$$

Dann ist $\Gamma_R = \bar{R}, PR, \bar{P}Q, \bar{P}\bar{Q}, P, \bar{P}, \square$ eine induzierte Resolutionsableitung zu Γ. Man überlegt sich leicht, daß die durch den Baum



dargestellte Resolutionswiderlegung Ω für S von keiner Ableitung $\mathcal{G}_S \vdash \mathcal{G}$ induziert werden kann, da nach der einleitenden Resolution auf $\bar{R}^1 R^2$ das Atom R nicht mehr im Graphen vorkommt.

2.5.9 <u>Korrektheit und Vollständigkeit</u>. Die klassischen Forderungen an einen Logikkalkül sind Korrektheit (engl: soundness) und Vollständigkeit (engl: completeness). Der KOWALSKI-Kalkül heißt

- <u>korrekt</u>, wenn jede Klauselmenge S, für die $\mathcal{G}_S$ widerlegbar ist, unerfüllbar ist.
- <u>vollständig</u>, wenn für jede unerfüllbare Klauselmenge S der initiale Graph $\mathcal{G}_S$ widerlegbar ist.

2.5.10 <u>Korrektheitssatz</u>. Der KOWALSKI-Kalkül ist korrekt.

Beweis. Sei S eine Klauselmenge und Γ eine Widerlegung für $\mathcal{G}_S$. Sei Γ_R eine induzierte Resolutionsableitung zu Γ. Dann ist Γ_R eine Resolutionswiderlegung für $S(\mathcal{G}_S)$. Die Korrektheit des Resolutionskalküls (1.4.10) garantiert, daß $S(\mathcal{G}_S)$ unerfüllbar ist. Wegen $S(\mathcal{G}_S) \subset S$ ist auch S unerfüllbar. [ ]

Die Vollständigkeit des KOWALSKI-Kalküls ist bis heute unbewiesen. In den Kapiteln 3 und 4 zeigen wir die Vollständigkeit für spezielle Klassen von Klauselmengen.

2.5.11 <u>Nichtabbrechende Ableitungen</u>. Eine unendliche Folge $\Sigma_1\Sigma_2\ldots\Sigma_n\ldots$ von Ableitungsschritten heißt nichtabbrechende Ableitung, wenn für alle $n \geq 1$ das Anfangsstück $\Sigma_1\ldots\Sigma_n$ eine Ableitung ist.

2.5.12 <u>Beispiel</u> für eine nichtabbrechende Ableitung.

$$\left|\begin{matrix} P & \bar{P} & P & \bar{P} \\ \bar{Q} & Q & & \end{matrix}\right| \xrightarrow[\rho]{\bar{P}^2 P^3} \left|\begin{matrix} P & \bar{P} & P & \bar{P} & & \\ \bar{Q} & Q & & & & Q \end{matrix}\right| \xrightarrow[\rho]{\bar{Q}^1 Q^5} \left|\begin{matrix} P & \bar{P} & P & \bar{P} & P \\ \bar{Q} & Q & & & \end{matrix}\right| =$$

$$= \left|\begin{matrix} P & \bar{P} & P & \bar{P} & P \\ \bar{Q} & Q & & & \end{matrix}\right| \xrightarrow[\rho]{\bar{P}^2 P^3} \ldots$$

2.6 Filter

Robert KOWALSKI nannte seinen Kalkül die "Connection Graph Proof Procedure" In der Tat kann man den KOWALSKI-Kalkül als einen indeterministischen Algorithmus auffassen: Zu einer Klauselmenge S wird zunächst der initiale Graph $\mathcal{G}_S$ gebildet. Danach werden solange indeterministisch gewählte Ableitungsschritte ausgeführt, bis die leere Klausel erzeugt ist. Die Korrektheit des Kalküls gestattet dann die Aussage, daß S unerfüllbar ist, und die erzeugte Widerlegung ist ein Beweis dafür.

Die naheliegende Forderung an diesen Algorithmus ist, daß er *vollständig* ist, d.h., daß er für jede unerfüllbare Klauselmenge S nach endlich vielen Schritten mit der Erzeugung der leeren Klausel terminiert. Diese Forderung ist aus zwei Gründen nicht erfüllt:

Erstens braucht der Ableitungsprozeß, auch für unerfüllbare S, nicht zu terminieren, da es, wie die Beispiele 2.5.12 und 2.7.6 zeigen, nichtabbrechende Ableitungen von $\mathcal{G}_S$ aus gibt.

Zweitens kann die Widerlegbarkeit an sich verloren gehen, wie die Beispiele 2.7.2 und 2.8.6 demonstrieren werden.

Man benötigt also, zusätzlich zum Kalkül, einen *Filter*, der die Wahl der Ableitungsschritte so einschränkt, daß die beiden obigen Insuffizienzen beseitigt werden.

2.6.1 Filter. Ein Filter $\mathcal{F}$ (für den KOWALSKI-Kalkül) ist ein entscheidbares Prädikat auf der Menge aller Ableitungen (des KOWALSKI-Kalküls), so daß für jede Ableitung $\Gamma = \Sigma_1 \ldots \Sigma_n$ gilt: Wenn $\mathcal{F}(\Gamma)$ gilt, dann gilt für jedes k mit $1 \leq k \leq n$: $\mathcal{F}(\Sigma_1 \ldots \Sigma_k)$. Eine Ableitung Γ heißt $\mathcal{F}$-Ableitung, wenn $\mathcal{F}(\Gamma)$ gilt. Eine nichtabbrechende Ableitung $\mathcal{G}_0 \longrightarrow \mathcal{G}_1 \longrightarrow \mathcal{G}_2 \longrightarrow \ldots$ heißt nichtabbrechende $\mathcal{F}$-Ableitung, wenn für alle $n \geq 1$ das Anfangsstück $\mathcal{G}_0 \longrightarrow \ldots \longrightarrow \mathcal{G}_n$ eine $\mathcal{F}$-Ableitung ist. Eine Ableitung Γ heißt $\mathcal{F}$-fortsetzbar, wenn es einen Ableitungsschritt Σ gibt, so daß $\Gamma\Sigma$ eine $\mathcal{F}$-Ableitung ist. Wir schreiben:

- $\Gamma: \mathcal{G} \vdash_{\mathcal{F}} \mathcal{G}'$, wenn Γ eine $\mathcal{F}$-Ableitung ist.
  $\mathcal{G} \vdash_{\mathcal{F}} \mathcal{G}'$, wenn es eine $\mathcal{F}$-Ableitung von $\mathcal{G}$ nach $\mathcal{G}'$ gibt.

Als Synonym für Filter benutzen wir die Begriffe Restriktion und Strategie. Ob wir einen Filter als Restriktion oder als Strategie bezeichnen, hängt vor allem von seiner sprachlichen Formulierung und der damit verbundenen Intuition

ab. Typische Beispiele für Restriktionen sind:

2.6.2 Die NR-Restriktion. Eine Ableitung Γ heißt NR-Ableitung, wenn in Γ nur ρ- und ϕ-Schritte vorkommen. NR steht für "no reduction".

2.6.3 Die FF-Restriktion. Eine Ableitung Γ: $\mathcal{G}_0 \longrightarrow \ldots \longrightarrow \mathcal{G}_n$ heißt FF-Ableitung, wenn die beiden folgenden Bedingungen erfüllt sind:

(a) Vor dem ersten Nicht-ϕ-Schritt in Γ wird zunächst die ϕ-Regel auf jeden Knoten in $\mathcal{G}_0$ so oft wie möglich angewendet.

(b) Nach einem ρ-Schritt in Γ wird die ϕ-Regel, vor dem nächsten Nicht-ϕ-Schritt, so oft wie möglich auf den Resolventenknoten angewendet.

FF steht für "full factoring".

2.6.4 Die T-Restriktion. Eine Ableitung Γ: $\mathcal{G}_0 \longrightarrow \ldots \longrightarrow \mathcal{G}_n$ heißt T-Ableitung, wenn keine der erzeugten Resolventen und Faktoren eine Tautologie ist.

2.6.5 Die U-Restriktion. Eine Ableitung Γ heißt U-Ableitung, wenn Γ keine ϕ-Schritte enthält und die ρ-Regel nur auf unäre Links angewendet wird. Ein Link heißt unär, wenn wenigstens eine seiner Inzidenzklauseln unär ist.

2.6.6 Generationen von Links und Knoten. Sei Γ: $\mathcal{G}_0 \longrightarrow \mathcal{G}_1 \longrightarrow \ldots \longrightarrow \mathcal{G}_n$ eine Ableitung. Wir definieren für jeden Knoten und für jeden Link in Γ die Generation (engl: level) rekursiv wie folgt:

- Für jeden Knoten a in $\mathcal{G}_0$ ist $Gen(a) := 0$.
- Für jeden durch einen ρ-Schritt $\mathcal{G}_i \xrightarrow[\rho]{\ell} \mathcal{G}_{i+1}$ eingebrachten Knoten c ist $Gen(c) := 1 + \max\{Gen(a), Gen(b)\}$, wobei a und b die Inzidenzknoten zu ℓ sind.
- Für jeden durch einen ϕ-Schritt $\mathcal{G}_i \xrightarrow[\phi]{a,\sigma} \mathcal{G}_{i+1}$ eingebrachten Knoten c ist $Gen(c) := Gen(a)$.
- Für jeden Link $\ell = L^a K^b$ in Γ ist $Gen(\ell) := \max\{Gen(a), Gen(b)\}$.

2.6.7 Die M-Restriktion. Eine Ableitung Γ heißt M-Ableitung, wenn für jeden ρ-Schritt $\mathcal{G} \xrightarrow[\rho]{\ell} \mathcal{G}'$ in Γ gilt: $Gen(\ell) = \min\{Gen(\ell') \mid \ell' \text{ ist Link in } \mathcal{G}\}$. M steht für "monotonous".

Bei Restriktionen liegt also die Betonung darauf, anzugeben, *welche Schritte nicht gewählt werden dürfen*. Dagegen sind Strategien positiv formuliert, d.h., es wird angegeben, *welche Schritte gewählt werden sollen*. Ein Beispiel ist:

2.6.8 Die BF-Strategie. Man erhält eine BF-Ableitung $\mathcal{G}_0 \longrightarrow \mathcal{G}_1 \longrightarrow \ldots$ wenn man von $\mathcal{G}_0$ ausgehend wie folgt vorgeht:

(a) Zuerst sind zu jedem Knoten in $\mathcal{G}_0$ mit der ϕ-Regel alle nichtisolierten Faktoren zu bilden.

(b) Die ρ-Regel darf auf einen Link ℓ in $\mathcal{G}_i$ nur dann angewendet werden, wenn seine Generation minimal ist, d.h., wenn für jeden Link ℓ' in $\mathcal{G}_i$ gilt, daß $\mathrm{Gen}(\ell) \leq \mathrm{Gen}(\ell')$.

(c) Nach einem ρ-Schritt, bei dem ein Resolventenknoten c gebildet wurde, müssen mit der ϕ-Regel sofort alle nichtisolierten Faktoren zu c erzeugt werden.

(d) Die τ- und die σ-Regel dürfen nicht angewendet werden.

BF steht für "breadth first search".

Wir haben die BF-Strategie bewußt informell formuliert, denn genau in der informellen Formulierung steckt der Unterschied zwischen Restriktionen und Strategien. Man überlegt sich nämlich leicht, daß die BF-Strategie gerade die *Überlagerung* der M-, FF-, und NR-Restriktion ist, d.h.:

$$\mathrm{BF} \iff \mathrm{M} \wedge \mathrm{FF} \wedge \mathrm{NR}$$

2.6.9 Konfluenz. Ein Filter $\mathcal{F}$ heißt konfluent, wenn für jede unerfüllbare Klauselmenge S gilt: Jede $\mathcal{F}$-Ableitung $\mathcal{G}_S \vdash\!\!- \mathcal{G}$, die keine Widerlegung ist, läßt sich zu einer $\mathcal{F}$-Widerlegung $\mathcal{G}_S \vdash\!\!- \mathcal{G} \vdash\!\!- \square$ fortsetzen.

2.6.10 Die Eigenschaft noethersch. Ein Filter $\mathcal{F}$ heißt noethersch, wenn für keine unerfüllbare Klauselmenge S eine nichtabbrechende $\mathcal{F}$-Ableitung von $\mathcal{G}_S$ aus existiert.

2.6.11 Vollständigkeit. Ein Filter $\mathcal{F}$ heißt vollständig, wenn $\mathcal{F}$ konfluent und noethersch ist.

Wenn $\mathcal{F}$ ein vollständiger Filter ist, dann stellt der KOWALSKI-Kalkül zusammen mit $\mathcal{F}$ einen indeterministischen Algorithmus dar, der für jede unerfüllbare Klauselmenge S nach endlich vielen Ableitungsschritten $\mathcal{G}_S \longrightarrow \mathcal{G}_1 \longrightarrow \mathcal{G}_2 \longrightarrow \ldots$ die leere Klausel erzeugt. Da bisher selbst die Vollständigkeit des KOWALSKI-kalküls an sich unbewiesen ist, ist natürlich auch die Existenz von vollständigen Filtern nicht gesichert. Es wird jedoch allgemein angenommen, daß die BF-Strategie vollständig ist.

In Kapitel 4 untersuchen wir den Spezialfall der unären Klauselmenge. Wir werden ein leicht überprüfbares Kriterium für die Vollständigkeit von Filtern angeben. Insbesondere erfüllt die BF-Strategie dieses Kriterium und ist damit für unäre Klauselmengen vollständig. Wir definieren jetzt einige Begriffe, die zur Formulierung von Vollständigkeitskriterien nützlich sind.

2.6.12 Erschöpfende nichtabbrechende Ableitungen. Eine nichtabbrechende Ableitung Γ: $\mathcal{G}_0 \longrightarrow \mathcal{G}_1 \longrightarrow \mathcal{G}_2 \longrightarrow \ldots$ heißt erschöpfend (engl: exhaustive), wenn gilt:

(a) Jeder Link in Γ wird nach endlich vielen Schritten wieder gelöscht, z.B. durch die ρ-Regel.

(b) Zu jedem Knoten a in Γ, der nicht wieder gelöscht wird, werden mit der ϕ-Regel sämtliche nichtisolierte Faktoren gebildet.

Die nicht abbrechende Ableitung in Beispiel 2.5.11 ist offensichtlich nicht erschöpfend. In Beispiel 2.7.6 geben wir eine erschöpfende nichtabbrechende Ableitung an.

2.6.13 Erschöpfende Filter. Ein Filter $\mathcal{F}$ heißt erschöpfend, wenn jede nichtabbrechende $\mathcal{F}$-Ableitung, die vom initialen Graphen $\mathcal{G}_S$ einer unerfüllbaren Klauselmenge S ausgeht, erschöpfend ist.

2.6.14 Fortsetzbarkeit. Ein Filter $\mathcal{F}$ heißt fortsetzbar, wenn für jede unerfüllbare Klauselmenge S und für jede $\mathcal{F}$-Ableitung Γ: $\mathcal{G}_S \vdash\!\!\!- \mathcal{G}$ gilt: Γ ist eine Widerlegung oder Γ ist $\mathcal{F}$-fortsetzbar.

2.6.15 Satz. Sei $\mathcal{F}$ ein noetherscher Filter. Dann ist $\mathcal{F}$ genau dann konfluent, wenn $\mathcal{F}$ fortsetzbar ist.

Beweis.

"==>". Trivial, da in der Definition 2.6.1 gefordert wird, daß, wenn $\mathcal{F}(\Sigma_1 \ldots \Sigma_n)$ gilt, stets für alle k mit $1 \leq k \leq n$ auch $\mathcal{F}(\Sigma_1 \ldots \Sigma_k)$ gilt.

"<==". Sei $\mathcal{F}$ ein noetherscher und fortsetzbarer Filter. *Angenommen*, $\mathcal{F}$ ist nicht konfluent. Dann gibt es eine unerfüllbare Klauselmenge S und eine $\mathcal{F}$-Ableitung Γ: $\mathcal{G}_S \vdash\!\!\!- \mathcal{G}$, die keine Widerlegung ist, und die sich auch nicht zu einer $\mathcal{F}$-Widerlegung fortsetzen läßt. Da $\mathcal{F}$ fortsetzbar ist, läßt sich Γ jedoch zu $\mathcal{F}$-Ableitungen $\Gamma\Sigma_1, \Gamma\Sigma_1\Sigma_2, \ldots$ fortsetzen. Also exi-

stiert eine nichtabbrechende $\mathcal{F}$-Ableitung $\Gamma\Sigma_1\Sigma_2\ldots$, die von $\mathcal{G}_S$ ausgeht. Da $\mathcal{F}$ noethersch und S unerfüllbar ist, kann dies nicht sein. W! []

2.6.16 <u>Vermutung</u>. Jeder erschöpfende und fortsetzbare Filter $\mathcal{F}$ mit $\mathcal{F}$ ==> NR ist vollständig.

2.7 Tautologien

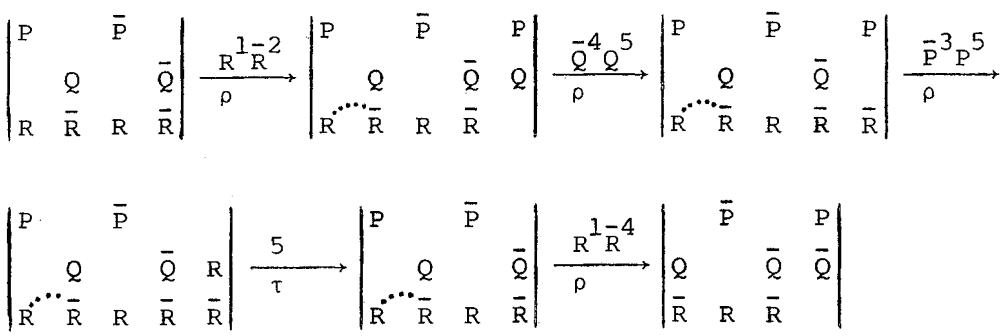
Der initiale Klauselgraph $\mathcal{G}_S$ ist tautologiefrei. Durch die ρ- und die ϕ-Regel können jedoch Resolventen und Faktoren gebildet werden, die Tautologien sind. Die τ-Regel gestattet die Elimination jedes Knotens, dessen Klausel eine Tautologie ist. Die τ-Regel des KOWALSKI-Kalküls ist der folgende Algorithmus:

2.7.1 Algorithmus τ

Eingabe: $\mathcal{G}$ Klauselgraph;
a : Knoten in $\mathcal{G}$, so daß C_a eine Tautologie ist;

Ausgabe: $\pi(\mathcal{G} - a)$.

2.7.2 Beispiel. Die folgende Ableitung zeigt, daß durch die uneingeschränkte Anwendung der τ-Regel die Widerlegbarkeit verloren gehen kann:



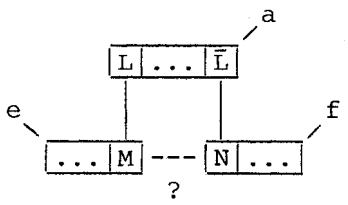
Der initiale Graph ist offensichtlich widerlegbar. Der finale Graph ist erfüllbar und damit nicht widerlegbar.

Ein konfluenter Filter muß also die Anwendbarkeit der τ-Regel einschränken. BIBEL gibt in /Bi81c/ die folgende Bedingung an, auf die wir in Kapitel 3 noch näher eingehen werden.

2.7.3 Die BIBELsche Tautologie-Eliminationsbedingung. Wir sagen, daß a die BIBELsche Tautologie-Eliminationsbedingung (BTEB) bzgl. L in $\mathcal{G}$ erfüllt, wenn gilt: $\mathcal{G}$ ist ein Klauselgraph, a ist ein Knoten in $\mathcal{G}$ und L und $\bar{L}$ sind Literale in C_a, so daß

für alle Links $M^e L^a$ und $N^{f-a} L^a$ in $\mathcal{G}$
gilt: $M^e N^f$ ist Link in $\mathcal{G}$.

Wir sagen, daß a die BTEB in $\mathcal{G}$ erfüllt, wenn ein Literal L existiert, so daß a die BTEB bzgl. L in $\mathcal{G}$ erfüllt. Die folgende Skizze veranschaulicht die BTEB:



Neben der Eigenschaft, daß C_a eine Tautologie ist, fordert die BTEB, daß zu zwei *Rampen* $M^e L^a$ und $N^{f-a} L^a$ stets die *Brücke* $M^e N^f$ existiert. Die in Beispiel 2.7.2 eliminierte Tautologie erfüllt die BTEB nicht, da die Brücke $R^1 R^{2-}$ fehlt.

2.7.4 Die BTE-Restriktion. Eine Ableitung Γ heißt BTE-Ableitung, wenn in Γ die τ-Regel nur auf Knoten angewendet wird, die die BTEB erfüllen.

2.7.5 Die BTELA-Restriktion. Eine Ableitung Γ heißt BTELA-Ableitung, wenn Γ eine BTE-Ableitung ist und jede erzeugte Tautologie sofort durch die τ-Regel eliminiert wird. LA steht für "look ahead", da nur solche tautologischen Resolventen und Faktoren erzeugt werden dürfen, die die BTEB erfüllen.

Die BTE-, BTELA- und auch die T-Restriktion (2.6.4) verhindern den in Beispiel 2.7.2 den Verlust der Widerlegbarkeit. Die T-Restriktion unterdrückt die Anwendung der τ-Regel völlig, da keine Tautologien mehr auftreten können. Sie ist verhältnismäßig effizient zu implementieren. Dies gilt nicht für die BTE-Restriktion, da unter Umständen sehr viele Rampenpaare existieren können. Die BTELA-Restriktion ist wegen der Vorausschau noch zeitintensiver als die BTE-Restriktion. Dafür ist sie, was die Suchraumreduktion anbetrifft, den beiden anderen Restriktionen überlegen. Die BTE-Restriktion ist bezüglich der Suchraumreduktion sogar der effizienteren T-Restriktion unterlegen, da nur ein Teil der erzeugten Tautologien, die unter Umständen sehr viele inzidente Links haben können, wieder gelöscht werden kann.

2.7.6 <u>Beispiel</u> für eine erschöpfende, nichtabbrechende BTELA-Ableitung:

$$\left|\begin{matrix} P & & \bar{P} & \\ & Q & & \bar{Q} \\ R & \bar{R} & R & \bar{R} \end{matrix}\right| \xrightarrow[\rho]{R^1\bar{R}^2} \left|\begin{matrix} P & & \bar{P} & & P \\ & Q & & \bar{Q} & Q \\ R & \bar{R} & R & \bar{R} & \end{matrix}\right| \xrightarrow[\rho]{R^1\bar{R}^4} \left|\begin{matrix} & \bar{P} & & P & P \\ Q & & \bar{Q} & Q & \bar{Q} \\ \bar{R} & R & \bar{R} & & \end{matrix}\right| \xrightarrow[\rho]{\bar{R}^1R^2}$$

$$\left|\begin{matrix} \bar{P} & & P & P & \bar{P} \\ & \bar{Q} & Q & \bar{Q} & Q \\ R & \bar{R} & & & \end{matrix}\right| \xrightarrow[\rho,\tau]{P^4\bar{P}^5}{}^{+} \left|\begin{matrix} \bar{P} & & P & P & \bar{P} \\ & \bar{Q} & Q & \bar{Q} & Q \\ R & \bar{R} & & & \end{matrix}\right| \xrightarrow[\rho,\tau]{\bar{Q}^4Q^5}{}^{+} \left|\begin{matrix} \bar{P} & & P & P & \bar{P} \\ & \bar{Q} & Q & \bar{Q} & Q \\ R & \bar{R} & & & \end{matrix}\right| \xrightarrow[\rho]{\bar{P}^1P^4}$$

$$\left|\begin{matrix} \bar{P} & & P & \bar{P} & \\ & \bar{Q} & Q & Q & \bar{Q} \\ R & \bar{R} & & & R \end{matrix}\right| \xrightarrow[\rho]{\bar{P}^1P^3} \left|\begin{matrix} & P & \bar{P} & & \\ \bar{Q} & Q & Q & \bar{Q} & Q \\ \bar{R} & & & R & R \end{matrix}\right| \xrightarrow[\rho,\tau]{\bar{Q}^1Q^5}{}^{+} \left|\begin{matrix} & P & \bar{P} & & \\ \bar{Q} & Q & Q & \bar{Q} & Q \\ \bar{R} & & & R & R \end{matrix}\right| \xrightarrow[\rho]{\bar{Q}^1Q^2}$$

$$\left|\begin{matrix} & P & \bar{P} & & & P \\ \bar{Q} & Q & Q & \bar{Q} & Q & \\ \bar{R} & & & R & R & \bar{R} \end{matrix}\right| \xrightarrow[\rho]{\bar{Q}^1Q^3} \left|\begin{matrix} P & & & P & \bar{P} \\ Q & Q & \bar{Q} & & \\ & R & R & \bar{R} & \bar{R} \end{matrix}\right| \xrightarrow[\rho]{Q^1\bar{Q}^2} \left|\begin{matrix} & & P & \bar{P} & P \\ \bar{Q} & Q & & & \\ R & R & \bar{R} & \bar{R} & R \end{matrix}\right| \xrightarrow[\rho,\tau]{\bar{P}^4P^5}{}^{+}$$

$$\left|\begin{matrix} & & P & \bar{P} \\ \bar{Q} & Q & & \\ R & R & \bar{R} & \bar{R} \end{matrix}\right| = \left|\begin{matrix} \bar{P} & & P & \\ & \bar{Q} & & Q \\ \bar{R} & R & \bar{R} & R \end{matrix}\right| .$$

Der initiale Graph ist der gleiche wie in Beispiel 2.7.2. Der finale Graph entspricht dem initialen Graphen bis auf die Tatsache, daß jedes Literalauftreten durch sein Komplement ersetzt ist. Folglich läßt sich der finale Graph mit den analogen Schritten wieder in den initialen überführen. Wir haben also insgesamt die Situation:

S ist eine unerfüllbare, aussagenlogische Klauselmenge.

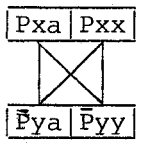
- Γ: $\mathcal{G}_S \xrightarrow[\rho,\tau]{+} \overline{\mathcal{G}_S} \xrightarrow[\rho,\tau]{+} \mathcal{G}_S$ ist eine BTELA-Ableitung.
- $\overline{\mathcal{G}_S}$ und $\mathcal{G}_S$ haben keine Klausel gemeinsam.

Also ist $\Gamma\Gamma\Gamma\ldots$ eine erschöpfende, nichtabbrechende BTELA-Ableitung, die vom initialen Graphen einer unerfüllbaren aussagenlogischen Klauselmenge ausgeht.

2.7.7 <u>τ-Links</u>. Ein Link in einem Klauselgraphen heißt τ-Link, wenn er kein Autolink ist, und wenn seine Resolvente eine Tautologie ist.

Das folgende interessante Beispiel wurde kürzlich von N. EISINGER gefunden:

2.7.8 Beispiel für einen initialen unerfüllbaren Graphen, in dem nur τ-Links vorkommen:



Jeder der vier Links ist ein τ-Link. Keine der tautologischen Resolventen erfüllt die BTEB. Die T- und die BTELA-Restriktion lassen also als Ableitungsschritte nur die Faktorisierung zu. Die BTE-Restriktion hingegen läßt die Resolution auf allen Links zu.

2.8 Subsumtion

2.8.1 S-Substitutionen. Seien C und D Klauseln. Eine Substitution θ mit $Ber(\theta) \subset Var(D)$ heißt S-Substitution von D nach C, wenn $\theta D \subset C$ und $|\theta D| = |D|$ ist.

2.8.2 Subsumierte Klauseln. Seien C und D Klauseln. Wir sagen:

- D subsumiert C, wenn eine S-Substitution von D nach C existiert.
- D subsumiert C reduzierend, wenn C von D subsumiert wird und C keine Variante von D ist.

Unsere Definition der Subsumtion weicht von der θ-Subsumtion in /Lo78, S.97/ ab. Dort wird statt $|\theta D| = |D|$ schwächer $|D| \leq |C|$ gefordert. In /Ro65/ wird auf die Forderung $|\theta D| = |D|$ völlig verzichtet. Sowohl unsere Bedingung als auch LOVELANDs Bedingung stellen Reduktionsforderungen dar. Diese Reduktivität ist im Hinblick auf vollständige Filter erforderlich. In Abschnitt 4.3 werden wir unsere Bedingung $|\theta D| = |D|$ näher begründen.

2.8.3 Lemma. Es gibt keine unendliche Folge $C_1C_2C_3\ldots$ von Klauseln, bei der, für alle $n \geq 1$, C_n von C_{n+1} reduzierend subsumiert wird.

Beweis. Nach Definition 1.1.7 sind Literale Zeichenreihen. Wenn L ein Literal ist, bezeichnen wir mit $|L|$ die Länge von L als Zeichenreihe. Offensichtlich gilt stets $|L| \geq 1$. Für eine Klausel $C = \{L_1, \ldots, L_n\}$ mit $n \geq 1$ definieren wir die Komplexität $[C]$ durch:

(1) $[C] := |L_1| + |L_2| + \ldots + |L_n| - |Var(C)|$.

Diese Klauselkomplexität erfüllt offensichtlich folgende Eigenschaften:

(2) $[C] > 0$; $[C] := 0$ genau dann, wenn $C = \square$.

(3) $C \subset D \Longrightarrow [C] \leq [D]$; $C \subset D \wedge C \neq D \Longrightarrow [C] < [D]$.

(4) $[C] \leq [\theta C]$ für jede Substitution θ.

(5) $[C] < [\theta C]$ für jede Substitution θ mit $Ber(\theta) \subset Var(C)$, die keine U-Substitution für C ist.

Wir zeigen nun, daß für zwei Klauseln C und D stets gilt:

(6) Wenn C von D reduzierend subsumiert wird, dann ist $[C] > [D]$.

Seien also C und D Klauseln, so daß C von D reduzierend subsumiert wird. Dann

gibt es eine Substitution θ mit $\theta D \subset C$, $|\theta D| = |D|$ und $Ber(\theta) \subset Var(D)$.

Fall 1: $|\theta D| < |C|$. Dann gilt: $[D] \leq [\theta D] < [C]$.
Fall 2: $|\theta D| = |C|$. Also ist $\theta D = C$. Da D keine Variante von C ist, kann θ keine U-Substitution für D sein. Also gilt: $[D] < [\theta D] = [C]$.

Angenommen, $C_1C_2C_3\ldots$ ist eine unendliche Folge von Klauseln, bei der, für alle $n \geq 1$, C_n reduziered von C_{n+1} subsumiert wird. Dann gilt $[C_1] > [C_2] > [C_3] > \ldots$ Da für $n \geq 1$ $[C_n] \in \mathbb{N}$ ist, kann dies nicht sein. *W!* []

2.8.4 Subsumierte Knoten. Sei $\mathcal{G}$ ein Klauselgraph und seien a und b zwei verschiedene Knoten in $\mathcal{G}$. Wir sagen, daß a von b [reduzierend] subsumiert wird, wenn C_a von C_b [reduzierend] subsumiert wird.

Die σ-Regel für den KOWALSKI-Kalkül ist der folgende Algorithmus:

2.8.5 Algorithmus σ

Eingabe: $\mathcal{G}$: Klauselgraph;
a : Knoten in $\mathcal{G}$, so daß ein von a verschiedener Knoten in $\mathcal{G}$ existiert, der a subsumiert.

Ausgabe: $\pi(\mathcal{G} - a)$.

2.8.6 Beispiel (N.EISINGER). Die folgende Ableitung zeigt, daß durch uneingeschränktes Anwenden der σ-Regel die Widerlegbarkeit verloren gehen kann.

$$\mathcal{G}_0 := \begin{array}{|ccccccccc|} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ \bar{P} & \bar{P} & \bar{P} & \bar{P} & & & & P & P \\ & & Q & \bar{Q} & Q & Q & \bar{Q} & \bar{Q} & Q \\ R & \bar{R} & & & \bar{R} & R & \bar{R} & & \end{array}$$

$\mathcal{G}_0$ ist ein totaler, π-reduzierter AT-Klauselgraph. Man rechnet leicht nach, daß $\mathcal{G}_0$ widerlegbar ist. Die Links $\bar{Q}^4Q^5$, $\bar{Q}^4Q^6$, $\bar{P}^2P^8$ und $\bar{P}^2P^9$ führen auf Resolventen, die von den Knoten 2, 1, 7 und 5 subsumiert werden. Wenn man die Resolutionen in der angegebenen Reihenfolge durchführt und die subsumierten Resolventen sofort mit der σ-Regel eliminiert, wird bei der letzten π-Reduktion die Spalte 2 gelöscht, und man erhält den (neu numerierten) Graphen $\mathcal{G}_1$:

$$\mathcal{G}_1 = \begin{array}{c} \begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \end{matrix} \\ \left|\begin{matrix} \bar{P} & \bar{P} & \bar{P} & & & & P & P \\ & Q & \bar{Q} & Q & Q & \bar{Q} & \bar{Q} & Q \\ R & & & \bar{R} & R & \bar{R} & & \end{matrix}\right| \end{array}$$

Die Resolventen zu $R^{1}\bar{R}^{4}$ und $R^{1}\bar{R}^{6}$ werden von 2 und 3 subsumiert. Resolution auf diesen Links und Elimination der subsumierten Resolventen liefert den Graphen $\mathcal{G}_2$:

$$\mathcal{G}_2 = \begin{array}{c} \begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{matrix} \\ \left|\begin{matrix} \bar{P} & \bar{P} & & & & P & P \\ Q & \bar{Q} & Q & Q & \bar{Q} & \bar{Q} & Q \\ & & \bar{R} & R & \bar{R} & & \end{matrix}\right| \end{array} \xrightarrow[\rho]{\bar{R}^{3}R^{4}} |\ldots| \xrightarrow[\rho]{\bar{Q}^{6}Q^{7}} \begin{array}{c} \begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \end{matrix} \\ \left|\begin{matrix} \bar{P} & \bar{P} & & & & P & P & P \\ Q & \bar{Q} & Q & Q & \bar{Q} & \bar{Q} & Q & \\ & & R & & \bar{R} & & & \end{matrix}\right| \end{array} =: \mathcal{G}_3$$

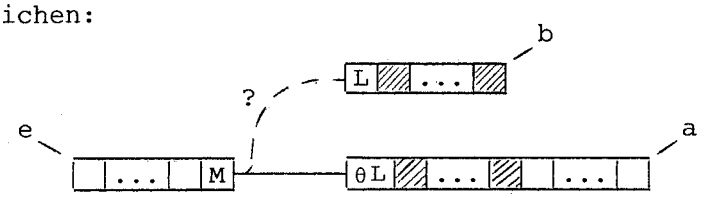
In $\mathcal{G}_3$ sind die Spalten 1 und 3 von Spalte 4 und die Spalten 6 und 7 von Spalte 8 subsumiert. Eliminiert man diese Spalten mit der σ-Regel, so kollabiert der Graph zum leeren Graphen $()$. Insgesamt haben wir also eine Ableitung $\mathcal{G}_o \xrightarrow[\rho,\sigma]{+} ()$, wobei $\mathcal{G}_o$ ein widerlegbarer, π-reduzierter und totaler AT-Klauselgraph ist.

Ein konfluenter Filter muß also, neben der Anwendbarkeit der τ-Regel, auch die Anwendbarkeit der σ-Regel einschränken. Wieder greifen wir auf eine Bedingung aus /Bi81c/ zurück:

2.8.7 <u>Die BIBELsche Subsumtions-Eliminationsbedingung</u>. Wir sagen, daß a die BIBELsche Subsumtionseliminationsbedingung (BSEB) bzgl. b und θ in $\mathcal{G}$ erfüllt, wenn a und b verschiedene Knoten des Klauselgraphen $\mathcal{G}$ sind, und wenn θ eine S-Substitution von C_b nach C_a ist, so daß

für jedes Literal $L \in C_b$
und für jeden Link $M^{e}\theta L^{a}$ in $\mathcal{G}$
gilt: $M^{e}L^{b}$ ist Link in $\mathcal{G}$.

Wir sagen, daß a die BSEB in $\mathcal{G}$ erfüllt, wenn es b und θ gibt, so daß a die BSEB bzgl. b und θ in $\mathcal{G}$ erfüllt. Die folgende Skizze soll die BSEB veranschaulichen:



Die BSEB fordert also, daß b den Knoten a nicht nur klauselmäßig, sondern auch bzgl. der Links subsumiert. Da den Links, wegen der Vererbung und der π-Reduktion, eine primäre Bedeutung zukommt, erscheint uns diese Forderung durchaus natürlich. Die Beweise in den Kapiteln 3 und 4 (Bemerkung 4.3.12) werden weiteres zur Erhellung der BSEB beitragen.

2.8.8 Die BSE-Restriktion. Eine Ableitung Γ heißt BSE-Ableitung, wenn in Γ die σ-Regel nur auf Knoten angewendet wird, die die BSEB erfüllen.

2.8.9 Die BSERF-Restriktion. Eine Ableitung Γ heißt BSERF-Ableitung, wenn Γ eine BSE-Ableitung ist und für jeden σ-Schritt Σ: $\mathcal{G} \xrightarrow[\sigma]{a} \mathcal{G}'$ in Γ eine der folgenden Bedingungen erfüllt ist:

(a) a ist ein Knoten im Initialgraphen von Γ.

(b) Σ folgt in Γ direkt auf einen ρ-Schritt und a wurde in diesem ρ-Schritt als Resolventenknoten erzeugt.

(c) Σ ist ein reduzierender BSE-Schritt, d.h., es gibt b und θ, so daß a die BSEB bzgl. b und θ in $\mathcal{G}$ erfüllt und C_b keine Variante von C_a ist.

RF steht für "reducing or forward".

2.9 Anmerkungen und Verweise

Der KOWALSKI-Kalkül wurde zuerst in /Ko75/ beschrieben und wird dort als die Connection Graph Proof Procedure bezeichnet. Die in /Ko75/ beschriebene Resolutionsregel arbeitet mit Multivererbung und erzeugt stets alle lokalen Links unabhängig vom Vererbungsmechanismus. Von BRUYNOOGHE /Br75/ stammt der Vorschlag, die lokalen Links durch Vererbung aus Autolinks zu erzeugen. Dieses Konzept wurde von KOWALSKI in /Ko79/ übernommen. Die Idee, weniger Links als bei Multivererbung zu erzeugen, geht auf BIBEL /Bi81a, Bi81c/ zurück. Seine Vererbungsmechanismen liegen zwischen der von uns eingeführten Mono- und der Multivererbung von KOWALSKI. Die Kalküle in /Ko75, Ko79/ enthalten keine Subsumtionsregel. Subsumtion beim KOWALSKI-Kalkül wird in den Arbeiten von EISINGER /Ei80, Ei81/, BIBEL /Bi81c/ und CHAMPEAUX /Ch82/ untersucht. EISINGER gibt einen effizienten Test auf Subsumtion an, der die speziellen Gegebenheiten in Klauselgraphen ausnutzt. Von BIBEL stammen die Eliminationsbedingungen für Tautologien und subsumierte Klauseln. In /Bi81c/ zeigt er, daß ein entsprechend eingeschränkter Kalkül zumindest für aussagenlogische Klauselmengen konsistent ist. Wir werden darauf im nächsten Kapitel näher eingehen.

In /Br76/, /SS76, SS80/ und /Bi81a, Bi81b/ wird das Vollständigkeitsproblem des KOWALSKI-Kalküls angegangen. Die Autoren versuchen jeweils, Kriterien für die entsprechenden Eigenschaften von Filtern zu beweisen. Leider sind die Beweise alle falsifiziert oder jedenfalls unvollständig. Bis heute ist selbst die Vollständigkeit des KOWALSKI-Kalküls an sich (siehe 2.5.9) unbewiesen.

WALTHER /Wa81a, Wa81b/ untersucht die Elimination redundanter Links und gibt effiziente Kriterien zu ihrer Erkennung an. Unter redundanten Links werden dabei τ-Links, oder Links, deren Resolvente isoliert ist, verstanden.

SIEKMANN und WRIGHTSON machen in /SW80/ Vorschläge zur Einbeziehung der Paramodulation in den KOWALSKI-Kalkül. Zusätzlich zu den (Resolutions-) Links führen sie P-Links ein, die mögliche Paramodulation repräsentieren, und die ähnlich wie die normalen Links vererbt werden.

Die uns bekannten Implementierungen des KOWALSKI-Kalküls sind das System von AMBLE /Am75/, der Beweiser COGITO von CHAMPEAUX /Ch78/ und die Markgraf Carl Refutation Procedure der Karlsruher Gruppe /BEHSSW81/. Das Karlsruher

System ist weltweit eines der am besten ausgebauten Beweissysteme und wird fortlaufend weiterentwickelt.

Schließlich erwähnen wir noch eine neuere Arbeit /Bi82a/ von BIBEL, in der er mehrere Beweisprozeduren, darunter auch die Connection Graph Proof Procedure, miteinander vergleicht.

KAPITEL 3

KONFLUENZ BEI AUSSAGENLOGISCHEN KLAUSELMENGEN

In diesem Kapitel entwickeln wir den Beweis für die Vollständigkeit der sogenannten AE-Strategie bei aussagenlogischen Klauselmengen. Der Beweis, der in anderer Formulierung in /Ei80/ und implizit auch in /Bi81c/ enthalten ist, baut auf der von BIBEL gezeigten Invarianz der Eigenschaft spannend auf /Bi81c/. Ein spannender Klauselgraph ist unerfüllbar und verfügt zudem über hinreichend viele Links, um auch nach einer π-Reduktion noch unerfüllbar zu sein.

Unser Beitrag ist der bisher fehlende, nichtelementare Beweis dafür, daß jeder spannende Klauselgraph auch widerlegbar ist. Zusammen mit der von BIBEL gezeig-

ten Invarianz der Eigenschaft spannend erhält man damit, daß die Konjunktion der BTE- mit der BSE-Restriktion konfluent ist.

Die Vollständigkeitsbeweise für die klassischen Verfeinerungen des Resolutionskalküls werden üblicherweise in zwei Stufen geführt: Zunächst wird die Vollständigkeit für den aussagenlogischen Fall gezeigt, um danach mit Hilfe des Satzes von HERBRAND durch das sogenannte Lifting auf den allgemeinen Fall übertragen zu werden. Dabei ist das Lifting der unproblematische Teil, so daß der Beweis praktisch schon dann erbracht ist, wenn die Vollständigkeit für aussagenlogische Klauselmengen gezeigt werden kann. Beim KOWALSKI-Kalkül treten aber durch die Linkvererbung und die π-Reduktion Schwierigkeiten auf, die nach unseren Untersuchungen ein Lifting sehr kompliziert oder sogar unmöglich erscheinen lassen.

3.1 Die Invarianz der Eigenschaft spannend

Wenn man den unerfüllbaren Graphen (d) in 2.1 π-reduziert, dann kollabiert er zum erfüllbaren leeren Graphen. Da im KOWALSKI-Kalkül jeder Ableitungsschritt von einer π-Reduktion begleitet wird, muß ein widerlegbarer Graph folglich, neben der Unerfüllbarkeit seiner Klauselmenge, zusätzlich die Eigenschaft haben, daß er über hinreichend viele Links verfügt, die ihn gegen die π-Reduktion stabilisieren. Die von BIBEL eingeführte Eigenschaft spannend charakterisiert diese zum Erhalt der Widerlegbarkeit erforderliche Linkreichhaltigkeit eines unerfüllbaren aussagenlogischen Klauselgraphen. BIBEL zeigt in /Bi81c/, daß der initiale Graph zu einer unerfüllbaren aussagenlogischen Klauselmenge spannend ist, und daß die Resolutionsregel für Klauselgraphen spannende Graphen in spannende Graphen überführt. Auch die τ- und die σ-Regeln erhalten die Eigenschaft spannend, allerdings im allgemeinen nur dann, wenn für den zu eliminierenden Knoten die entsprechende BIBELsche Eliminationsbedingung erfüllt ist. Diese zunächst etwas artifiziell erscheinenden Eliminationsbedingungen werden vor dem Hintergrund der Eigenschaft spannend transparent: Sie sind gerade hinreichende Bedingungen dafür, daß der Graph auch ohne die Links an den tautologischen bzw. subsumierten Knoten noch spannend ist.

Bevor wir näher auf spannende Klauselgraphen eingehen, halten wir noch zwei spezielle Eigenschaften der ρ-Regel für den Fall fest, daß diese auf einen aussagenlogischen Graphen angewendet wird:

3.1.1 Lemma. Sei $\mathcal{G} \xrightarrow[\rho]{\ell} \mathcal{G}'$ und sei $\mathcal{G}$ ein aussagenlogischer Klauselgraph.

(a) Wenn $\mathcal{G}'$ den Resolventenknoten c zu ℓ enthält, $L \in C_c$ ist und K^d ein signifikanter Vorgänger zu L^c ist, dann gilt für jeden Link $M^e K^d$ in $\mathcal{G}'$, daß auch $M^e L^c$ ein Link in $\mathcal{G}'$ ist.

(b) Wenn $\mathcal{G}'$ den Resolventenknoten zu ℓ nicht enthält, dann gibt es einen nichtunären Inzidenzknoten von ℓ, der nicht in $\mathcal{G}'$ enthalten ist.

Beweis. Sei $\mathcal{G}$ ein aussagenlogischer Klauselgraph mit $\mathcal{G} \xrightarrow[\rho]{\ell} \mathcal{G}'$ Sei $\mathcal{G}''$ ein Graph mit:

(1) $\mathcal{G} \xrightarrow[\rho_o]{\ell} \mathcal{G}''$ und $\mathcal{G}' = \pi(\mathcal{G}'' - \ell)$.

Schließlich sei c der Resolventenknoten zu ℓ in $\mathcal{G}'$.

Teil (a). Sei c in $\mathcal{G}'$ enthalten. Sei $L \in C_c$ ein Literal und sei K^d ein signifikanter Vorgänger zu L^c bzgl. des Übergangs (1). Sei $M^e K^d$ ein Link in $\mathcal{G}'$. Wir müssen zeigen, daß $M^e L^c$ ein Link in $\mathcal{G}'$ ist. Zunächst ist $M^e K^d$ auch ein Link in $\mathcal{G}''$. Da $\mathcal{G}$ aussagenlogisch ist, gilt $K = L = \bar{M}$. Lemma 2.3.6 liefert nun, daß $M^e L^c$ ein Link in $\mathcal{G}''$ ist. Da $\mathcal{G}'$ nach Voraussetzung sowohl e als auch c enthält und $\ell \neq M^e L^c$ ist, ist $M^e L^c$ auch ein Link in $\pi(\mathcal{G}'' - \ell)$.

Teil (b). Sei c in $\mathcal{G}'$ nicht enthalten. Seien $\mathcal{G}_0, \ldots, \mathcal{G}_n$ Graphen mit $n \geq 1$ und

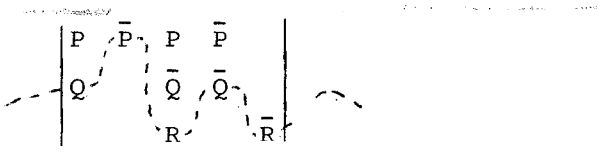
(2) $$\mathcal{G} \xrightarrow[\rho_0]{\ell} \mathcal{G}'' \subset \mathcal{G}'' - \ell = \mathcal{G}_0 \xrightarrow[\pi]{} \mathcal{G}_1 \xrightarrow[\pi]{} \ldots\ldots \xrightarrow[\pi]{} \mathcal{G}_n = \mathcal{G}'.$$

Sei $\mathcal{G}_k$ der erste Graph, in dem c isoliert ist (c ist dann ein Knoten in $\mathcal{G}_k$). Dann gibt es ein Literal $L \in C_c$, so daß L^c in $\mathcal{G}_k$ isoliert ist. Sei K^d der signifikante Vorgänger zu L^c in $\mathcal{G}''$ bzgl. des Übergangs (1). Dann ist d ein nichtunärer (d.h. $|C_d| \geq 2$) Inzidenzknoten von ℓ. Wir werden zeigen, daß d kein Knoten in $\mathcal{G}'$ ist. Da $K = L$ ist ($\mathcal{G}$ ist aussagenlogisch), garantiert Lemma 2.3.6, daß $M^e K^d$ genau dann ein Link in $\mathcal{G}''$ ist, wenn $M^e L^c$ ein Link in $\mathcal{G}''$ ist. Da ℓ weder mit K^d noch mit L^c inzidiert, gilt also auch:

(3) $M^e K^d$ ist genau dann ein Link in $\mathcal{G}_0$, wenn $M^e L^c$ ein Link in $\mathcal{G}_0$ ist.

Da bei der π-Reduktion (2) ein Link $M^e L^c$ genau dann gelöscht wird, wenn der Knoten e gelöscht wird, ist der Knoten d in $\mathcal{G}_k$ entweder gar nicht enthalten oder aber wenigstens isoliert. Also ist d, wie zu zeigen war, kein Knoten in $\mathcal{G}'$. []

3.1.2 <u>Die Eigenschaft spannend</u>. Sei $\mathcal{G}$ = <Knoten,C,Links> ein aussagenlogischer Klauselgraph. Eine Abbildung π: Knoten $\longrightarrow \mathbb{L}$ heißt <u>Pfad in $\mathcal{G}$</u>, wenn für alle $a \in$ Knoten $\pi(a) \in C_a$ ist. Ein Pfad π in $\mathcal{G}$ heißt <u>komplementär in $\mathcal{G}$</u>, wenn es Knoten a und b in $\mathcal{G}$ gibt, so daß $\pi(a)^a \pi(b)^b$ ein Link in $\mathcal{G}$ ist. $\mathcal{G}$ heißt <u>spannend</u>, wenn $\mathcal{G}$ nichtleer ist und wenn jeder Pfad in $\mathcal{G}$ komplementär in $\mathcal{G}$ ist. Ein Graph, der die leere Klausel enthält, ist also stets spannend, da er keine Pfade besitzt. In der Matrixdarstellung lassen sich Pfade besonders gut veranschaulichen:



Wenn man sich die Literale der Matrix als Perlen vorstellt, ist ein Pfad ein Faden, der von links nach rechts durch die Matrix läuft und in jeder Spalte durch genau ein Literal gezogen ist. Komplementär ist ein Pfad genau dann,

wenn zwei seiner Perlen mit einem Link verbunden sind.

3.1.3 Satz (BIBEL).

(a) Ein totaler aussagenlogischer Klauselgraph ist genau dann unerfüllbar, wenn er spannend ist.

(b) Jeder spannende aussagenlogische Klauselgraph ist unerfüllbar.

Beweisskizze. Zu einem aussagenlogischen Klauselgraphen $\mathcal{G}$ läßt sich eine Formel $(\ldots \vee \ldots) \wedge \ldots \wedge (\ldots \vee \ldots)$ in konjunktiver Normalform (KNF) angeben, indem man jedem Knoten a in $\mathcal{G}$ eine Formel $(\ldots \vee \ldots)$ zuordnet, die C_a im Sinne von 1.3.4 entspricht. Diese läßt sich durch vollständiges Ausmultiplizieren in eine Formel $(\ldots \wedge \ldots) \vee \ldots \vee (\ldots \wedge \ldots)$ in disjunktiver Normalform (DNF) überführen. Eine Formel in DNF ist unerfüllbar genau dann, wenn jede Klammer zwei komplementäre Literale enthält. Da die Pfade eines Graphen genau der DNF seiner KNF-Formel entsprechen, ergeben sich nun (a) und (b) leicht. []

Der Leser wird sich an dieser Stelle eventuell fragen, warum wir etwas so kompliziertes wie den KOWALSKI-Kalkül untersuchen, wenn sich die Unerfüllbarkeit einer aussagenlogischen Klauselmenge doch einfach dadurch entscheiden läßt, daß man alle Pfade auf Komplementarität untersucht. Wir nutzen die Gelegenheit, um festzustellen, daß eine Klauselmenge $S = \{C_1, C_2, \ldots, C_n\}$ nicht weniger als $|C_1| \cdot |C_2| \cdot \ldots \cdot |C_n|$ Pfade besitzt, also exponetiell viele relativ zur Anzahl der Klauseln. Bei nur 50 Klauseln mit mindestens je vier Literalen erreichen wir schon die stattliche Anzahl von 10^{10} Pfaden.

Das nächste Lemma zeigt, daß die durch die Eigenschaft spannend charakterisierte Linkreichhaltigkeit eines Graphen durch die π-Reduktion nicht beeinträchtigt wird:

3.1.4 Lemma. Wenn $\mathcal{G}$ ein spannender aussagenlogischer Klauselgraph ist, dann ist auch $\pi(\mathcal{G})$ spannend.

Beweis. In Abschnitt 2.2 haben wir $\pi(\mathcal{G})$ als die Normalform von $\mathcal{G}$ bzgl. der konfluenten und noetherschen π-Relation definiert. Es genügt also, die folgende Behauptung zu zeigen:

(1) Wenn $\mathcal{G} \xrightarrow[\pi]{} \mathcal{G}'$ gilt und $\mathcal{G}$ ein spannender aussagenlogischer Klauselgraph ist, dann ist auch $\mathcal{G}'$ spannend.

Sei also $\mathcal{G}$ ein spannender aussagenlogischer Klauselgraph und sei $\mathcal{G}'$ ein Graph mit $\mathcal{G} \xrightarrow[\pi]{} \mathcal{G}'$. Sei π' ein Pfad in $\mathcal{G}'$. Wir müssen zeigen, daß π' in $\mathcal{G}'$ komplementär ist. Zunächst existiert (siehe 2.2.2) ein isoliertes Literalauftreten L^a in $\mathcal{G}$ mit $\mathcal{G}' = \mathcal{G} - a$. π' ist also ein Pfad in $\mathcal{G} - a$. Sei π die Erweiterung von π' auf $\mathcal{G}$ mit $\pi(a) := L$. Da $\mathcal{G}$ spannend ist, ist π komplementär in $\mathcal{G}$. Es existieren also zwei Knoten e und f in $\mathcal{G}$, so daß $\pi(e)^e\pi(f)^f$ ein Link in $\mathcal{G}$ ist. Da L^a in $\mathcal{G}$ isoliert und $\pi(a) = L$ ist, gilt $e \neq a$ und $f \neq a$. Also sind e und f auch Knoten in $\mathcal{G} - a$. Da in $\mathcal{G} - a$ gegenüber $\mathcal{G}$ nur solche Links fehlen, die mit a inzidieren, ist $\pi(e)^e\pi(f)^f = \pi'(e)^e\pi'(f)^f$ auch ein Link in $\mathcal{G} - a$. Also ist π', wie zu zeigen war, in $\mathcal{G}' = \mathcal{G} - a$ komplementär. []

3.1.5 <u>Die BTSE-Restriktion.</u> Eine Ableitung Γ heißt BTSE-Ableitung, wenn Γ eine BTE- und auch eine BSE-Ableitung ist.

3.1.6 <u>Satz</u> (BIBEL). Sei $\mathcal{G}$ ein spannender aussagenlogischer Klauselgraph. Dann ist jeder Graph $\mathcal{G}'$ mit $\mathcal{G} \vdash_{\text{BTSE}} \mathcal{G}'$ spannend.

Beweis. Sei $\mathcal{G}$ ein spannender aussagenlogischer Klauselgraph. Wir zeigen den Satz in drei Teilen.

Teil 1: Wenn $\mathcal{G}'$ ein Graph mit $\mathcal{G} \xrightarrow[\rho]{} \mathcal{G}'$ ist, dann ist $\mathcal{G}'$ spannend. Sei also $\mathcal{G}'$ ein Graph mit $\mathcal{G} \xrightarrow[\rho]{} \mathcal{G}'$. Dann existiert ein Link ℓ in $\mathcal{G}$ und ein Graph $\mathcal{G}''$ mit:
(1) $\mathcal{G} \xrightarrow[\rho_o]{\ell} \mathcal{G}''$ und $\mathcal{G}' = \pi(\mathcal{G}'' - \ell)$.
Wegen Lemma 3.1.4 genügt es zu zeigen, daß $\mathcal{G}'' - \ell$ spannend ist. Sei also π'' ein Pfad in $\mathcal{G}'' - \ell$.
Angenommen, π'' ist nicht komplementär in $\mathcal{G}'' - \ell$. Sei c der Resolventenknoten zu ℓ in $\mathcal{G}''$. Offensichtlich gilt:
(2) $\mathcal{G} = \mathcal{G}'' - c$.
Sei K^d ein signifikanter Vorgänger zu $\pi''(c)^c$ bzgl. des Übergangs (1). Dann ist K^d ein Literalauftreten in $\mathcal{G}$, das nicht mit ℓ inzidiert. Wir definieren einen Pfad ψ'' in $\mathcal{G}''$ durch

$$\psi''(x) := \begin{cases} K & \text{wenn } x = d \\ \pi''(x) & \text{sonst.} \end{cases}$$

Sei ψ die Einschränkung von ψ'' auf $\mathcal{G}$. Da $\mathcal{G}$ spannend ist, existieren zwei Knoten e und f in $\mathcal{G}$, so daß $\ell_\psi = \psi(e)^e\psi(f)^f$ ein Link in $\mathcal{G}$ ist. Da ℓ zwar mit d, aber nicht mit K^d inzidiert, und $\psi(d) = K$ ist, gilt jedenfalls $\ell_\psi \neq \ell$. Also

ist ℓ_ψ ein Link in $\mathcal{G}'' - \ell$.

Fall 1.1: ℓ_ψ inzidiert nicht mit d. Dann gilt $\ell_\psi = \pi''(e)^e \pi''(f)^f$. Folglich ist π'' komplementär in $\mathcal{G}'' - \ell$. *W!*

Fall 1.2: ℓ_ψ inzidiert mit d. Dann gilt o.B.d.A. $K^d = \psi(f)^f$. Da K^d ein signifikanter Vorgänger zu $\pi''(c)^c$ ist und $\widehat{\psi(e)} = K = \pi''(c)$ gilt, ist $\ell'' := \psi(e)^e \pi''(c)^c$ wegen Lemma 2.3.6 ein Link in $\mathcal{G}''$ und damit auch in $\mathcal{G}'' - \ell$. Wegen $e \neq d$ ist $\ell'' = \pi''(e)^e \pi''(c)^c$. Also ist π'' komplementär in $\mathcal{G}'' - \ell$. *W!*

Teil 2: Wenn $\mathcal{G}'$ ein Graph mit $\mathcal{G} \xrightarrow[\tau]{a} \mathcal{G}'$ ist und a die BTEB in $\mathcal{G}$ erfüllt, dann ist $\mathcal{G}'$ spannend . Sei also $\mathcal{G}'$ ein Graph mit $\mathcal{G} \xrightarrow[\tau]{a} \mathcal{G}'$ und sei a ein Knoten, der die BTEB bzgl. L in $\mathcal{G}$ erfüllt. Dann gilt $\mathcal{G}' = \pi(\mathcal{G} - a)$. Wegen Lemma 3.1.4 genügt es zu zeigen, daß $\mathcal{G} - a$ spannend ist. Sei also π ein Pfad in $\mathcal{G} - a$.

Angenommen, π ist nicht komplementär in $\mathcal{G} - a$. Sei

- π_L der Pfad in $\mathcal{G}$, der durch Erweiterung von π mit $\pi_L(a) = L$ entsteht.
- $\pi_{\bar{L}}$ der Pfad in $\mathcal{G}$, der durch Erweiterung von π mit $\pi_{\bar{L}}(a) = \bar{L}$ entsteht.

Da $\mathcal{G}$ spannend ist, existieren zwei Knoten e und f in $\mathcal{G}$, so daß $\ell_1 := \pi_L(e)^e \pi_L(a)^a = \pi(e)^e L^a$ und $\ell_2 := \pi_{\bar{L}}(f)^f \pi_{\bar{L}}(a)^a = \pi(f)^f \bar{L}^a$ Links in $\mathcal{G}$ sind. Da a die BTEB bzgl. L in $\mathcal{G}$ erfüllt, existiert zu den beiden Rampen ℓ_1 und ℓ_2 die Brücke $\ell_3 = \pi(e)^e \pi(f)^f$ in $\mathcal{G}$. Da ℓ_3 nicht mit a inzidiert, ist ℓ_3 auch ein Link in $\mathcal{G} - a$. Also ist π in $\mathcal{G} - a$ komplementär. *W!*

Teil 3: Wenn $\mathcal{G}'$ ein Graph mit $\mathcal{G} \xrightarrow[\sigma]{a} \mathcal{G}'$ ist und a die BSEB in $\mathcal{G}$ erfüllt, dann ist $\mathcal{G}'$ spannend. Sei also $\mathcal{G}'$ ein Graph mit $\mathcal{G} \xrightarrow[\sigma]{a} \mathcal{G}'$ und sei a ein Knoten, der die BSEB bzgl. b und ε in $\mathcal{G}$ erfüllt. Dann gilt $\mathcal{G}' = \pi(\mathcal{G} - a)$. Wegen Lemma 3.1.4 genügt es wieder zu zeigen, daß $\mathcal{G} - a$ spannend ist. Sei also π ein Pfad in $\mathcal{G} - a$.

Angenommen, π ist nicht komplementär in $\mathcal{G} - a$. Sei ψ die Erweiterung von π auf $\mathcal{G}$ mit $\psi(a) := \pi(b)$ (wohldefiniert, da $\varepsilon C_b = C_b \subset C_a$). Da $\mathcal{G}$ spannend ist, ist ψ in $\mathcal{G}$ komplementär. Da die Einschränkung von ψ auf $\mathcal{G} - a$ in $\mathcal{G} - a$ nicht komplementär ist, existiert ein Knoten e in $\mathcal{G}$, so daß $\psi(e)^e \psi(a)^a$ ein Link in $\mathcal{G}$ ist. Da a die BSEB bzgl. b und ε in $\mathcal{G}$ erfüllt, muß auch $\ell := \psi(e)^e \psi(a)^b = \pi(e)^e \pi(b)^b$ ein Link in $\mathcal{G}$ sein. Da ℓ nicht mit a inzidiert, ist ℓ auch ein Link in $\mathcal{G} - a$. Also ist π in $\mathcal{G} - a$ komplementär. *W!*

Sei nun $\mathcal{G}'$ ein Graph mit $\mathcal{G} \vdash_{\text{BTSE}} \mathcal{G}'$. Da die Faktorisierungsregel auf aussagenlogische Graphen nicht anwendbar ist, entsteht $\mathcal{G}'$ aus $\mathcal{G}$ durch endlich viele Anwendungen der ρ-, τ- und der σ-Regel, wobei bei der τ- und der σ-Regel jeweils die BIBELschen Eliminationsbedingungen erfüllt sind. Wie wir in Teil 1

bis Teil 3 gezeigt haben, bleibt bei jedem dieser Ableitungsschritte die Eigenschaft spannend erhalten. Also ist $\mathcal{G}'$, wie zu zeigen war, spannend. []

3.1.7 <u>Korollar</u>. Sei S eine aussagenlogische Klauselmenge.

(a) Wenn $\mathcal{G}$ ein Graph mit $\mathcal{G}_S \vdash_{\overline{BTSE}} \mathcal{G}$ ist, dann ist $\mathcal{G}$ genau dann erfüllbar, wenn S erfüllbar ist.

(b) Wenn $\mathcal{G}_S \vdash_{\overline{BTSE}} ()$ gilt, dann ist S erfüllbar.

Beweis. Sei S eine aussagenlogische Klauselmenge.

Teil (a). Sei $\mathcal{G}$ ein Graph mit $\mathcal{G}_S \vdash_{\overline{BTSE}} \mathcal{G}$.

"$\mathcal{G}$ erfüllbar ==> S erfüllbar". Sei also $\mathcal{G}$ erfüllbar. *Angenommen*, S ist unerfüllbar. Dann ist wegen Satz 3.1.3(a) $\mathcal{G}_S$ spannend. Wegen Satz 3.1.6 ist auch $\mathcal{G}$ spannend. Also ist $\mathcal{G}$ wegen Satz 3.1.3(b) auch unerfüllbar. *W!*

"S erfüllbar ==> $\mathcal{G}$ erfüllbar". Sei also S erfüllbar. Wegen $S(\mathcal{G}_S) \subset S$ ist auch $\mathcal{G}_S$ erfüllbar. Da in einer Ableitung $\mathcal{G}_S \vdash_{\overline{BTSE}} \mathcal{G}$ neue Klauseln nur mit der Resolutionsregel erzeugt werden, ist $\mathcal{G}$, wie zu zeigen war, wegen Lemma 1.4.5 erfüllbar.

Teil (b). Folgt sofort aus Teil (a), da der leere Graph erfüllbar ist. []

3.2 Die AE-Entscheidungsprozedur.

Mit Hilfe der im vorhergehenden Abschnitt nachgewiesenen Invarianz der Eigenschaft spannend läßt sich nun zeigen, daß die von der DAVIS-Putnam-Beweisprozedur /DP60/ her bekannte *Atomeliminierungs-Strategie* (AE-Strategie) für den KOWALSKI-Kalkül bei aussagenlogischen Klauselmengen konfluent und noethersch ist. Das Prinzip der AE-Strategie ist, nacheinander die vorkommenden Atome aus dem Graphen zu eliminieren. Dabei wird ein Atom L aus dem Graphen entfernt, indem alle mit L inzidenten Links mit der ρ- oder der τ-Regel gelöscht werden. Die τ-Regel wird benötigt, um zu verhindern, daß bei Resolution auf $L\bar{L}$-Tautologien solche neuen Links gebildet werden, die mit L inzidieren.

In der Einleitung zu Abschnitt 2.6 haben wir ausgeführt, daß der KOWALSKI-Kalkül zusammen mit einer Strategie einen indeterministischen Beweisalgorithmus darstellt. Entsprechend formulieren wir hier die AE-Strategie als einen Algorithmus, der, ausgehend vom initialen Graphen einer Klauselmenge, Ableitungsschritte im KOWALSKI-Kalkül durchführt. In dieser Verpackung als Algorithmus entspricht der Vollständigkeit der AE-Strategie die totale Korrektheit der AE-Beweisprozedur. Es wird sich zeigen, daß die AE-Beweisprozedur sogar eine Entscheidungsprozedur für die Aussagenlogik ist.

3.2.1 Algorithmus AE-Entscheidungsprozedur

Eingabe: S : aussagenlogische Klauselmenge;

Variablen: $\mathcal{G}$: Klauselgraph := $\mathcal{G}_S$;
L : $\mathbb{L}$; a : $\mathbb{K}$; ℓ : Link;

$\mathcal{G}$-Invariante: $\mathcal{G}_S \vdash_{BTSE} \mathcal{G}$;

1: Solange $\mathcal{G} \neq ()$ und $\mathcal{G} \neq (\square)$

1-Invariante: $\mathcal{G}$ total;

1.1: Wähle ein Atom L in $\mathcal{G}$;

1.2: Solange $\mathcal{G}$ einen Knoten a enthält,
so daß C_a eine Tautologie ist

1.2.1: $\mathcal{G} := \pi(\mathcal{G} - a)$; -- τ-Regel

1.3: Solange $\mathcal{G}$ einen Knoten a enthält,
so daß a von einem anderen Knoten in $\mathcal{G}$ subsumiert wird

1.3.1: $\mathcal{G} := \pi(\mathcal{G} - a)$; -- σ-Regel

1.4: Solange $\mathcal{G}$ einen Link ℓ enthält,
so daß ℓ mit den Literalen L und $\bar{L}$ inzidiert

1.4.1: $\mathcal{G} := \rho(\mathcal{G}, \ell)$; -- ρ-Regel

Ausgabe: "erfüllbar", wenn $\mathcal{G} = ()$, sonst "unerfüllbar"

3.2.2 Satz. Die AE-Entscheidungsprozedur ist total korrekt.

Beweis. Für den Beweis benötigen wir zwei neue Begriffe. Sei L ein Literal. Dann heißt

- ein aussagenlogischer Klauselgraph $\mathcal{G}$ total bis auf L, wenn für zwei komplementäre Literalauftreten K^a und $\bar{K}^b$ in $\mathcal{G}$ mit $|K| \neq |L|$ stets gilt, daß $K^a\bar{K}^b$ ein Link in $\mathcal{G}$ ist.
- eine Klausel C Tautologie bzgl. L, wenn C die Literale L und $\bar{L}$ enthält.

Da die π-Reduktion die Totalität ohne L erhält, läßt sich mit Lemma 2.3.6, analog wie bei Satz 2.3.7, leicht zeigen, daß gilt:

(0) Wenn $\mathcal{G}$ ein bis auf L totaler aussagenlogischer Klauselgraph mit $\mathcal{G} \xrightarrow[\rho]{\ell} \mathcal{G}'$ ist, dann ist, wenn ℓ mit L inzidiert, auch $\mathcal{G}'$ total bis auf L.

Wir verifizieren nun die AE-Entscheidungsprozedur, indem wir zunächst für jede der vier Schleifen eine passende Invariante zeigen und nachweisen, daß die Schleife terminiert, wenn ihre Invariante zu Beginn der Abarbeitung erfüllt ist. Mit Hilfe dieser Schleifeninvarianten werden wir dann die $\mathcal{G}$-Invariante und schließlich die Nachbedingung zeigen.

Schleife 1.2. Wir zeigen, daß

(1) $\mathcal{G}$ ist total und $\mathcal{G}_S \vdash_{BTSE} \mathcal{G}$

eine Invariante der Schleife 1.2 ist, und daß Schleife 1.2 stets terminiert.

Terminierung. Trivial.

Invariante. Sei S eine Klauselmenge und sei $\mathcal{G}$ ein totaler Graph mit $\mathcal{G}_S \vdash_{BTSE} \mathcal{G}$. Sei a ein Knoten in $\mathcal{G}$, so daß C_a eine Tautologie ist. Sei $\mathcal{G}' := \pi(\mathcal{G} - a)$. Wir müssen zeigen, daß $\mathcal{G}'$ ein totaler Graph mit $\mathcal{G}_S \vdash_{BTSE} \mathcal{G}'$ ist. Da $\mathcal{G}$ total ist, ist $\mathcal{G} - a$ total und damit auch $\mathcal{G}' = \pi(\mathcal{G} - a)$. Die Totalität von $\mathcal{G}$ liefert weiter, daß a die BTEB in $\mathcal{G}$ erfüllt. Da $\mathcal{G} \xrightarrow[\tau]{a} \mathcal{G}'$, gilt damit auch $\mathcal{G} \vdash_{BTSE} \mathcal{G}'$. Folglich ist $\mathcal{G}_S \vdash_{BTSE} \mathcal{G}'$, wie zu zeigen war.

Schleife 1.3. Analog wie bei Schleife 1.2 zeigt man, daß (1) Invariante von Schleife 1.3 ist, und daß Schleife 1.3 immer terminiert.

Schleife 1.4. Wir zeigen, daß

(2) $\mathcal{G}$ ist total bis auf L und $\mathcal{G}$ enthält keine Tautologie bzgl. L und $\mathcal{G}_S \vdash_{BTSE} \mathcal{G}$

eine Invariante von Schleife 1.4 ist, und daß Schleife 1.4 stets terminiert, wenn vor der Abarbeitung die Bedingung (2) erfüllt ist.

Invariante. Sei S eine aussagenlogische Klauselmenge, sei L ein Literal und sei $\mathcal{G}$ ein Graph mit $\mathcal{G}_S \vdash_{BTSE} \mathcal{G}$, der total bis auf L ist und keine Tautologien bzgl. L enthält. Sei ℓ ein Link in $\mathcal{G}$, der mit dem Literal L inzidiert. Schließlich sei $\mathcal{G}'$ ein Graph mit $\mathcal{G} \xrightarrow[\rho]{\ell} \mathcal{G}'$. Wir müssen zeigen, daß $\mathcal{G}'$ total bis auf L ist, daß $\mathcal{G}'$ keine Tautologie bzgl. L enthält, und daß $\mathcal{G}_S \vdash_{BTSE} \mathcal{G}'$ gilt. Da wegen $\mathcal{G} \xrightarrow[\rho]{\ell} \mathcal{G}'$ auch $\mathcal{G} \vdash_{BTSE} \mathcal{G}'$ ist, gilt folglich auch $\mathcal{G}_S \vdash_{BTSE} \mathcal{G}'$. Da $\mathcal{G}$ total bis auf L ist, $\mathcal{G}'$ aus $\mathcal{G}$ durch Resolution auf einem Link entsteht, der mit L und $\bar{L}$ inzidiert, ist wegen (0) auch $\mathcal{G}'$ total bis auf L. Schließlich kann die Resolvente zu ℓ keine Tautologie bzgl. L sein, da sie weder das Literal L noch das Literal $\bar{L}$ enthält. Dies liegt daran, daß ℓ mit L und $\bar{L}$ inzidiert und die Inzi-

denzklauseln von ℓ keine Tautologien bzgl. L sind. Also enthält $\mathcal{G}'$ keine Tautologie bzgl. L und erfüllt somit alle Bedingungen von (2).

Terminierung. Da, wenn vor der Abarbeitung die Bedingung (2) erfüllt ist, die Resolvente zu ℓ, wie wir oben gezeigt haben, weder das Literal L noch das Literal $\bar{L}$ enthält, wird bei jedem Durchlauf der Schleife 1.4 die Anzahl der Links, die mit L inzidieren, echt kleiner.

Schleife 1. Wir zeigen, daß (1) Invariante von Schleife 1 ist, und daß Schleife 1 stets terminiert, wenn vor der Abarbeitung (1) erfüllt ist.

Invariante. Sei S eine aussagenlogische Klauselmenge, sei $\mathcal{G}$ ein totaler Graph mit $\mathcal{G}_S \vdash_{BTSE} \mathcal{G}$ und sei L ein Atom in $\mathcal{G}$. Sei

- $\mathcal{G}_1$ ein Graph, der aus $\mathcal{G}$ durch Anwendung der Schleife 1.2 entsteht.
- $\mathcal{G}_2$ ein Graph, der aus $\mathcal{G}_1$ durch Anwendung der Schleife 1.3 entsteht.
- $\mathcal{G}_3$ ein Graph, der aus $\mathcal{G}_2$ durch Anwendung der Schleife 1.4 entsteht.

Wir müssen zeigen, daß $\mathcal{G}$ bzw. $\mathcal{G}_1$ bzw. $\mathcal{G}_2$ die Invarianten der Schleifen 1.2 bzw. 1.3 bzw. 1.4 erfüllen, und daß $\mathcal{G}_3$ total mit $\mathcal{G}_S \vdash_{BTSE} \mathcal{G}_3$ ist. Zunächst erfüllt $\mathcal{G}$ die Invariante von Schleife 1.2. Also ist $\mathcal{G}_1$ ein totaler Graph mit $\mathcal{G}_S \vdash_{BTSE} \mathcal{G}_1$, der keine Tautologie enthält. Da $\mathcal{G}_1$ die Invariante von Schleife 1.3 erfüllt und offensichtlich $\mathcal{G}_2 \subset \mathcal{G}_1$ gilt, ist $\mathcal{G}_2$ ebenfalls ein totaler Graph mit $\mathcal{G}_S \vdash_{BTSE} \mathcal{G}_2$, der keine Tautologien enthält. Also erfüllt $\mathcal{G}_2$ die Invariante von Schleife 1.4. Folglich ist $\mathcal{G}_3$ total bis auf L und erfüllt $\mathcal{G}_S \vdash_{BTSE} \mathcal{G}_3$. Aus $\mathcal{G}_S \vdash_{BTSE} \mathcal{G}_3$ folgt, daß $\mathcal{G}_3$ zudem π-reduziert ist. Die Terminierungsbedingung von Schleife 1.4 sichert, daß $\mathcal{G}_3$ keinen Link mehr enthält, der mit L inzidiert. Also enthält $\mathcal{G}_3$ die Literale L und $\bar{L}$ nicht mehr. Da wir schon gezeigt haben, daß $\mathcal{G}_3$ total bis auf L ist, haben wir nun, daß $\mathcal{G}_3$ sogar total ist, wie zu zeigen war.

Terminierung. Wir haben oben gezeigt, daß $\mathcal{G}_3$ das Atom L nicht mehr enthält. Also wird bei jedem Durchlauf der Schleife 1 die Anzahl der in $\mathcal{G}$ vorkommenden Atome echt kleiner.

$\mathcal{G}$-Invariante. Da $\mathcal{G}$ zu $\mathcal{G}_S$ initialisiert wird, ist die $\mathcal{G}$-Invariante nach der Initialisierung erfüllt. Da die $\mathcal{G}$-Invariante von allen Schleifeninvarianten impliziert wird, ist die $\mathcal{G}$-Invariante für jeden Graphen $\mathcal{G}$ erfüllt, den die AE-Entscheidungsprozedur als Zwischenergebnis erzeugt.

Nachbedingung. Sei S eine aussagenlogische Klauselmenge und sei $\mathcal{G}'$ ein Graph, der aus $\mathcal{G}_S$ durch Anwendung der Schleife 1 entsteht. Wir müssen zeigen, daß

$\mathcal{G}' = ()$ genau dann gilt, wenn S erfüllbar ist. Zunächst garantieren die $\mathcal{G}$-Invariante und die Terminierunsbedingung von Schleife 1, daß gilt:

(3) $\mathcal{G}_S \vdash_{BTSE} \mathcal{G}'$ und entweder $\mathcal{G}' = ()$ oder $\mathcal{G}' = (\square)$.

"$\mathcal{G}' = ()$ ==> S erfüllbar". Ergibt sich direkt aus Korollar 3.1.7(b).

"S erfüllbar ==> $\mathcal{G}' = ()$". Sei also S erfüllbar. Mit Korollar 3.1.7(a) folgt, daß $\mathcal{G}'$ erfüllbar ist. Wegen (3) gilt $\mathcal{G}' = ()$ oder $\mathcal{G}' = (\square)$. Da $(\square)$ unerfüllbar ist, muß $\mathcal{G}' = ()$ sein. []

3.2.3 <u>Korollar</u>. Der KOWALSKI-Kalkül ist für aussagenlogische Klauselmengen vollständig.

Beweis. Sei S eine unerfüllbare aussagenlogische Klauselmenge. Wir müssen zeigen, daß eine Widerlegung für $\mathcal{G}_S$ existiert. Zunächst ist S eine korrekte Eingabe für die AE-Entscheidungsprozedur. Satz 3.2.2 und die $\mathcal{G}$-Invariante garantieren, daß die AE-Entscheidungsprozedur für S mit einer Ableitung $\mathcal{G}_S \vdash_{BTSE} (\square)$ terminiert. Also existiert eine Widerlegung für $\mathcal{G}_S$. []

3.2.4 <u>Satz</u>. Die AE-Entscheidungsprozedur hat exponentiellen Aufwand relativ zur Anzahl der Literalauftreten in S.

Beweis. Sei S eine aussagenlogische Klauselmenge. Dann terminiert die AE-Entscheidungsprozedur für S mit einer Ableitung $\Gamma: \mathcal{G}_S \vdash ()$ oder $\Gamma: \mathcal{G}_S \vdash (\square)$. Sei Γ_R eine induzierte Ableitung (siehe 2.5.7) zu Γ. Offensichtlich ist Γ_R eine reguläre Resolutionsableitung (für die Definition siehe /Ga77, S.26/). Theorem 4.2.1 in /Ga77/ besagt nun, daß es ein $c > 0$ gibt, so daß für alle n eine unerfüllbare aussagenlogische Klauselmenge S mit n Literalauftreten existiert, so daß jede reguläre Resolutionswiderlegung für S wenigstens 2^{cn} Resolutionen enthält. Dies bedeutet, daß die AE-Entscheidungsprozedur für ein solches S wenigstens 2^{cn} ρ-Schritte (Schritt 1.4.1) ausführt. Also hat die AE-Entscheidungsprozedur exponentiellen Aufwand relativ zur Anzahl der Literalauftreten in S. Wir bemerken noch, daß die Exponentialität der regulären Resolution zuerst von TSEITIN gezeigt wurde (siehe /Ts68/). []

3.3 Der Konfluenzsatz

3.3.1 Satz. Sei $\mathcal{G}$ ein spannender aussagenlogischer Klauselgraph. Dann existiert eine Widerlegung $\mathcal{G} \xrightarrow[\rho]{*} \square$, bei der nur auf Non-$\tau$-Links resolviert wird.

Wir verschieben den umfangreichen Beweis und stellen zunächst fest:

3.3.2 Korollar. Die BTSE- und die BTSE$\wedge$T-Restriktion sind für aussagenlogische Klauselmengen konfluent.

Beweis. Sei S eine unerfüllbare aussagenlogische Klauselmenge und sei Γ: $\mathcal{G}_S \vdash_{BTSE} \mathcal{G}$. Dann liefern die Sätze 3.1.3 und 3.1.6, daß $\mathcal{G}$ spannend ist. Also existiert wegen Satz 3.3.1 eine Widerlegung Γ': $\mathcal{G} \vdash_{BTSE\wedge T} \square$. Die Konkatenation $\Gamma\Gamma'$ ist eine BTSE-Widerlegung. Also ist die BTSE-Restriktion für aussagenlogische Klauselmengen konfluent. Wenn Γ sogar eine BTSE$\wedge$T-Ableitung ist, dann ist auch $\Gamma\Gamma'$ eine BTSE$\wedge$T-Widerlegung. Also ist auch die BTSE$\wedge$T-Restriktion für aussagenlogische Klauselmengen konfluent. [ ]

Die Konfluenz der BTSE-Restriktion für aussagenlogische Klauselmengen ergibt sich also in drei Teilen:

- Unerfüllbare totale Klauselgraphen sind spannend (Satz 3.1.3).
- BTSE-Ableitungen erhalten die Eigenschaft spannend (Satz 3.1.6).
- Jeder spannende Klauselgraph ist widerlegbar (Satz 3.3.1).

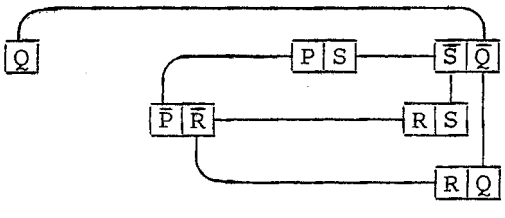
Unser Satz 3.3.1 stellt also das bisher fehlende letzte Drittel des Konfluenzbeweises dar. BIBEL zeigte in /Bi81c, Corollary 2.7/ erstmals, daß der KOWALSKI-Kalkül wenigstens für aussagenlogische Klauselmengen vollständig ist. Sein Beweis benutzt genau die Technik, die auch der AE-Entscheidungsprozedur zugrunde liegt, und ist damit, wie wir gleich näher begründen werden, entscheidend davon abhängig, daß der initiale Graph total ist. Um Satz 3.3.1 zu zeigen, bei dem für den initialen Graphen nur noch vorausgesetzt wird, daß er spannend ist, bedarf es also einer neuen Beweistechnik.

Wenn man den Beweis (siehe 3.2.2) für die totale Korrektheit der AE-Entscheidungsprozedur analysiert, stellt man fest, daß er wesentlich davon abhängt, daß alle auftretenden Tautologien die BTEB erfüllen und damit, unter Erhalt der Eigenschaft spannend, eliminiert werden können. Damit die Tautologien aber

die BTEB erfüllen, muß der initiale Graph total gewesen sein. Wenn er nur noch spannend ist, können nicht mehr alle Tautologien eliminiert werden. Das hat zur Folge, daß man die Terminierung der Schleife 1.4 der AE-Entscheidungsprozedur (siehe 3.2.1) nicht mehr garantieren kann, da dann neue Links, die mit dem Atom L inzidieren, entstehen können. Das Prinzip der AE-Entscheidungsprozedur ist, durch die *kombinierte* Anwendung der ρ- und der τ-Regel, nacheinander die vorkommenden Atome zu eliminieren. Auch in unserem Beweis für Satz 3.3.1 werden wir die Anzahl der vorkommenden Atome reduzieren, diesmal allerdings durch die Kombination der ρ- und der σ-Regel.

Für den Beweis von Satz 3.3.1 benötigen wir eine spezielle Literal-Eliminierungstechnik, die von den Vollständigkeitsbeweisen für die klassischen Restriktionen des Resolutionskalküls her bekannt ist:

3.3.3 <u>$\mathcal{G}$-L^a und $\mathcal{G}$-L</u>. Sei $\mathcal{G}$ ein aussagenlogischer Klauselgraph und sei L^a ein Literalauftreten. Dann bezeichnet $\mathcal{G}$-L^a den Graphen, der aus $\mathcal{G}$ entsteht, wenn man das Literalauftreten L^a (also nicht den ganzen Knoten a) und alle mit L^a inzidenten Links löscht. Mit $\mathcal{G}$-L bezeichnen wir den Graphen, den man aus $\mathcal{G}$ erhält, wenn man alle Auftreten des Literals L löscht. Offensichtlich enthält $\pi(\mathcal{G}\text{-}L)$ das Atom $|L|$ nicht mehr. Wenn $\mathcal{G}$ der Graph aus 2.1.11 und a der linkeste Knoten in $\mathcal{G}$ ist, dann ist $\mathcal{G}$-P^a der folgende Graph:



Der Graph $\pi(\mathcal{G}\text{-}P)$ schließlich hat die Form:

3.3.4 <u>Lemma</u>. Sei $\mathcal{G}$ ein spannender aussagenlogischer Klauselgraph und sei L^a ein Literalauftreten. Dann gilt:

(a) $\mathcal{G}$-L^a ist spannend.

(b) $\mathcal{G}$-L und $\pi(\mathcal{G}\text{-}L)$ sind spannend.

Beweis. Sei $\mathcal{G}$ ein spannender aussagenlogischer Klauselgraph und sei L^a ein Literalauftreten.

Teil (a). Sei π ein Pfad in $\mathcal{G}$-L^a. Wir müssen zeigen, daß π komplementär in $\mathcal{G}$-L^a ist. Zunächst ist π auch ein Pfad in $\mathcal{G}$. Da $\mathcal{G}$ spannend ist, ist π in $\mathcal{G}$ komplementär. Folglich gibt es Knoten e und f in $\mathcal{G}$, so daß $\pi(e)^e\pi(f)^f$ ein Link in $\mathcal{G}$ ist. Da π ein Pfad in $\mathcal{G}$-L^a ist, gilt $\pi(e)^e \neq L^a$ und $\pi(f)^f \neq L^a$. Da in $\mathcal{G}$-L^a gegenüber $\mathcal{G}$ nur Links fehlen, die mit L^a inzidieren, ist $\pi(e)^e\pi(f)^f$ auch ein Link in $\mathcal{G}$-L^a. Also ist π in $\mathcal{G}$-L^a komplementär.

Teil (b). Da die π-Reduktion die Eigenschaft spannend erhält (Lemma 3.1.4), folgt mit Teil (a) sofort, daß mit $\mathcal{G}$ auch $\mathcal{G}$-L und $\pi(\mathcal{G}$-L) spannend sind. [ ]

Um die folgenden Lemmata bequem formulieren zu können, führen wir noch <u>zwei weitere Notationen</u> ein. Wir schreiben $\mathcal{G} \xrightarrow[\rho_T]{} \mathcal{G}'$, wenn es in $\mathcal{G}$ einen Non-τ-Link ℓ mit $\mathcal{G} \xrightarrow[\rho]{\ell} \mathcal{G}'$ gibt. Wenn $\mathcal{G}$ ein Klauselgraph und L ein Literal ist, dann ist <u>Auftr(L,$\mathcal{G}$)</u> := $\{L^a \mid L^a$ ist Literalauftreten in $\mathcal{G}\}$.

3.3.5 <u>Lemma</u>. Seien $\mathcal{G}$ und $\mathcal{H}$ aussagenlogische Klauselgraphen mit $\mathcal{G} \subset \mathcal{H}$. Dann gilt:

(a) $\pi(\mathcal{G}) \subset \pi(\mathcal{H})$.

(b) Wenn $\mathcal{G} \xrightarrow[\rho]{\ell} \mathcal{G}'$ gilt, dann existiert ein Graph $\mathcal{H}'$ mit $\mathcal{H} \xrightarrow[\rho]{\ell} \mathcal{H}'$ und $\mathcal{G}' \subset \mathcal{H}'$.

(c) Wenn $\mathcal{G} \vdash_T \square$ gilt, dann existiert eine Widerlegung $\mathcal{H} \xrightarrow[\rho_T]{*} \square$.

Beweis. Seien $\mathcal{G}$ und $\mathcal{H}$ aussagenlogische Klauselgraphen mit $\mathcal{G} \subset \mathcal{H}$.

Teil (a). Da $\mathcal{G} \subset \mathcal{H}$ ist, gilt auch $\pi(\mathcal{G}) \subset \mathcal{H}$. Da $\pi(\mathcal{G})$ ein π-reduzierter Subgraph von $\mathcal{H}$ und $\pi(\mathcal{H})$ der größte π-reduzierte Subgraph von $\mathcal{H}$ ist, gilt $\pi(\mathcal{G}) \subset \pi(\mathcal{H})$.

Teil (b). Sei $\mathcal{G} \xrightarrow[\rho]{\ell} \mathcal{G}'$. Dann existiert ein Graph $\mathcal{G}''$ mit $\mathcal{G} \xrightarrow[\rho_o]{\ell} \mathcal{G}''$ und $\mathcal{G}' = \pi(\mathcal{G}'' - \ell)$. Wegen Teil (a) genügt es zu zeigen, daß ein Graph $\mathcal{H}''$ mit $\mathcal{H} \xrightarrow[\rho_o]{\ell} \mathcal{H}''$ und $\mathcal{G}'' \subset \mathcal{H}''$ existiert. Dies ist aber evident, da man in ρ_o den Resolventenknoten c, die U-Substitution θ und die signifikanten Vorgänger geeignet wählen kann.

Teil (c). Sei Γ eine T-Widerlegung für $\mathcal{G}$. Wir zeigen durch vollständige Induktion über die Länge von Γ, daß eine Widerlegung $\mathcal{H} \xrightarrow[\rho_T]{*} \square$ existiert.

"$|\Gamma| = 0$". Dann enthält $\mathcal{G}$ die leere Klausel. Also enthält auch $\mathcal{H}$ die leere Klausel.

"$|\Gamma| > 0$". Sei $\Gamma = \Sigma\Gamma'$ und sei Σ der Ableitungsschritt $\mathcal{G} \xrightarrow[\alpha]{} \mathcal{G}_1$. Offensichtlich ist Γ' eine T-Widerlegung für $\mathcal{G}_1$ mit $|\Gamma'| < |\Gamma|$.

Fall 1: $\alpha = \tau$ oder $\alpha = \sigma$. Dann gilt $\mathcal{G}_1 \subset \mathcal{G} \subset \mathcal{H}$. Also ist die Induktionsvoraussetzung auf $\mathcal{G}_1$, $\mathcal{H}$ und Γ' anwendbar. Folglich existiert, wie zu zeigen war, eine

Widerlegung $\mathcal{H} \xrightarrow[\rho T]{*} \square$

Fall 2: $\alpha = \phi$. Dies kann nicht sein, da $\mathcal{G}$ aussagenlogisch ist.

Fall 3: $\alpha = \rho$. Sei ℓ ein Non-τ-Link in $\mathcal{G}$ mit $\mathcal{G} \xrightarrow[\rho T]{\ell} \mathcal{G}_1$. Dann ist ℓ auch ein Non-τ-Link in $\mathcal{H}$. Wegen Teil (b) existiert ein Graph $\mathcal{H}_1$ mit $\mathcal{H} \xrightarrow[\rho T]{\ell} \mathcal{H}_1$ und $\mathcal{G}_1 \subset \mathcal{H}_1$. Die Induktionsvoraussetzung ist auf $\mathcal{G}_1$, $\mathcal{H}_1$ und Γ' anwendbar und liefert $\mathcal{H}_1 \xrightarrow[\rho T]{*} \square$. Also gilt $\mathcal{H} \xrightarrow[\rho T]{\ell} \mathcal{H}_1 \xrightarrow[\rho T]{*} \square$, wie zu zeigen war. []

3.3.6 <u>Lemma</u>. Sei $\mathcal{G}$ ein π-reduzierter aussagenlogischer Klauselgraph und sei L ein Literal. Wenn $\pi(\mathcal{G}-L) \xrightarrow[\rho T]{\ell} \mathcal{S}$ gilt, dann existiert ein Graph $\mathcal{G}'$ mit $\mathcal{G} \xrightarrow[\rho T]{\ell} \mathcal{G}'$, der eine der folgenden zwei Bedingungen erfüllt:

(a) $|\mathrm{Auftr}(\bar{L},\mathcal{G}')| < |\mathrm{Auftr}(\bar{L},\mathcal{G})|$.

(b) $|\mathrm{Auftr}(\bar{L},\mathcal{G}')| = |\mathrm{Auftr}(\bar{L},\mathcal{G})|$ und $\mathcal{S} \subset \pi(\mathcal{G}'-L)$.

Beweis. Sei $\mathcal{G}$ ein π-reduzierter aussagenlogischer Klauselgraph, sei L ein Literal und sei $\pi(\mathcal{G}-L) \xrightarrow[\rho T]{\ell} \mathcal{S}$. Dann existieren wegen Lemma 3.3.5(b) Graphen $\mathcal{S}'$ und $\mathcal{S}''$ mit:

$$\begin{array}{ccccc} \mathcal{G}-L & \xrightarrow[\rho_0]{\ell} & \mathcal{S}'' \supset \pi(\mathcal{S}''-\ell) & = & \mathcal{S}' \\ \cup & & & & \cup \\ \pi(\mathcal{G}-L) & \xrightarrow[\rho T]{\ell} & & & \mathcal{S} \end{array}$$

Da die Inzidenzklauseln (in $\mathcal{G}$) von ℓ das Literal $\bar{L}$ nicht enthalten, ist ℓ auch in $\mathcal{G}$ ein Non-τ-Link. Man überlegt sich leicht, daß ein Graph $\mathcal{G}''$ existiert mit:

(1) $\mathcal{G} \xrightarrow[\rho_0]{\ell} \mathcal{G}''$ und $\mathcal{S}'' = \mathcal{G}''-L$.

Wir definieren

(2) $\mathcal{G}' := \pi(\mathcal{G}''-\ell)$.

Offensichtlich gilt

(3) $\mathcal{G} \xrightarrow[\rho T]{\ell} \mathcal{G}'$ und $\mathrm{Auftr}(\bar{L},\mathcal{G}') \subset \mathrm{Auftr}(\bar{L},\mathcal{G}'') \subset \mathrm{Auftr}(\bar{L},\mathcal{G})$.

Fall 1: $|\mathrm{Auftr}(\bar{L},\mathcal{G}')| < |\mathrm{Auftr}(\bar{L},\mathcal{G})|$. Dann erfüllt $\mathcal{G}'$ die Bedingung (b).

Fall 2: $|\mathrm{Auftr}(\bar{L},\mathcal{G}')| = |\mathrm{Auftr}(\bar{L},\mathcal{G})|$. Dann gilt, wie wir zeigen werden:

(4) $\mathcal{S}' = \pi(\mathcal{G}'-L)$.

Mit (1) folgt sofort, daß

(5) $\mathcal{S}' = \pi(\mathcal{S}''-\ell) = \pi((\mathcal{G}''-\ell) - L)$.

Wegen $\pi(\mathcal{G}''-\ell) - L \subset (\mathcal{G}''-\ell) - L$ gilt

(6) $\pi(\mathcal{G}'-L) = \pi(\pi(\mathcal{G}''-\ell) - L) \subset \pi((\mathcal{G}''-\ell) - L) = \mathcal{S}'$.

Da $\mathrm{Auftr}(\bar{L},\pi(\mathcal{G}''-\ell)) = \mathrm{Auftr}(\bar{L},\mathcal{G}''-\ell)$ und $\mathcal{G}''-\ell$ keine isolierten Auftreten von L enthält (da $\mathcal{G}$ π-reduziert (!) ist und ℓ nicht mit dem Literal L inzidiert), gilt, wie zu zeigen war, die Bedingung (4). Mit $\mathcal{S} \subset \mathcal{S}'$ folgt, daß $\mathcal{G}'$ die Be-

dingung (b) des Lemmas erfüllt. []

3.3.7 Lemma. Sei $\mathcal{G}$ ein π-reduzierter aussagenlogischer Klauselgraph und sei L ein Literal. Sei $\mathcal{H}$ ein Subgraph von $\pi(\mathcal{G}-L)$ mit $\mathcal{H} \xrightarrow[\rho T]{*} \square$. Dann existiert ein Graph $\mathcal{G}'$ mit $\mathcal{G} \xrightarrow[\rho T]{*} \mathcal{G}'$, der eine der folgenden Bedingungen erfüllt:

(a) $\square \in S(\mathcal{G}')$.

(b) $\{L\} \in S(\mathcal{G}')$.

(c) $|\mathrm{Auftr}(\bar{L},\mathcal{G}')| < |\mathrm{Auftr}(\bar{L},\mathcal{G})|$.

Beweis. Sei $\mathcal{G}$ ein π-reduzierter aussagenlogischer Klauselgraph, sei L ein Literal und sei $\mathcal{H}$ ein Subgraph von $\pi(\mathcal{G}-L)$. Sei Γ: $\mathcal{H} \xrightarrow[\rho T]{*} \square$ eine Widerlegung für $\mathcal{H}$. Wir zeigen durch vollständige Induktion über die Länge von Γ, daß ein Graph $\mathcal{G}'$ mit $\mathcal{G} \xrightarrow[\rho T]{*} \mathcal{G}'$ existiert, der eine der Bedingungen (a), (b) oder (c) erfüllt.

"$|\Gamma| = 0$". Dann ist $\square \in S(\mathcal{H})$. Also gilt $\square \in S(\mathcal{G})$ oder $\{L\} \in S(\mathcal{G})$. $\mathcal{G}' := \mathcal{G}$ ist folglich ein Graph mit $\mathcal{G} \xrightarrow[\rho T]{*} \mathcal{G}'$, der die Bedingung (a) oder (b) erfüllt.

"$|\Gamma| > 0$". Sei $\Gamma = \Sigma\Gamma'$ und sei Σ: $\mathcal{H} \xrightarrow[\rho T]{\ell} \mathcal{H}_1$. Wegen Lemma 3.3.5(b) existiert ein Graph $\mathcal{S}$ mit:

(1) $\pi(\mathcal{G}-L) \xrightarrow[\rho T]{\ell} \mathcal{S}$ und $\mathcal{H}_1 \subset \mathcal{S}$.

Wegen Lemma 3.3.6 ($\mathcal{G}$ ist π-reduziert!) existiert nun ein Graph $\mathcal{G}_1$ mit

(2) $\mathcal{G} \xrightarrow[\rho T]{\ell} \mathcal{G}_1$,

auf den einer der beiden folgenden Fälle zutrifft:

Fall 1: $|\mathrm{Auftr}(\bar{L},\mathcal{G}_1)| < |\mathrm{Auftr}(\bar{L},\mathcal{G})|$. Dann erfüllt $\mathcal{G}' := \mathcal{G}_1$ die Bedingung (b).

Fall 2: $|\mathrm{Auftr}(\bar{L},\mathcal{G}_1)| = |\mathrm{Auftr}(\bar{L},\mathcal{G})|$ und $\mathcal{S} \subset \pi(\mathcal{G}_1-L)$. Wegen (1) ist dann $\mathcal{H}_1 \subset \pi(\mathcal{G}_1-L)$. Also ist die Induktionsvoraussetzung auf $\mathcal{G}_1$, L, $\mathcal{H}_1$ und Γ' anwendbar. Folglich existiert ein Graph $\mathcal{G}'$ mit $\mathcal{G} \xrightarrow[\rho T]{\ell} \mathcal{G}_1 \xrightarrow[\rho T]{*} \mathcal{G}'$, der wegen $|\mathrm{Auftr}(\bar{L},\mathcal{G}_1)| = |\mathrm{Auftr}(\bar{L},\mathcal{G})|$ eine der Bedingungen (a), (b) oder (c) des Lemmas erfüllt. []

3.3.8 Beweis für Satz 3.3.1. Wegen Lemma 3.3.5(c) genügt es, die folgende, schwächere Behauptung zu zeigen:

(1) Für jeden π-reduzierten, spannenden aussagenlogischen Klauselgraphen existiert eine T-Widerlegung.

Für einen Klauselgraphen $\mathcal{G}$ definieren wir

(2) $\mathrm{Atome}(\mathcal{G}) := \{|L| \mid$ es gibt einen Knoten a in $\mathcal{G}$ mit $L \in C_a\}$.

Wir beweisen nun die Behauptung (1) mit einer Doppelinduktion.

Sei also $\mathcal{G}$ ein π-reduzierter, spannender aussagenlogischer Klauselgraph. Wir zeigen durch vollständige Induktion über $|\text{Atome}(\mathcal{G})|$, daß eine T-Widerlegung für $\mathcal{G}$ existiert.

"$|\text{Atome}(\mathcal{G})| = 0$" Dann enthält $\mathcal{G}$ die leere Klausel. Also existiert eine T-Widerlegung für $\mathcal{G}$.

"$|\text{Atome}(\mathcal{G})| > 0$". Sei $L \in \text{Atome}(\mathcal{G})$. Wir zeigen zunächst mit Hilfe der Induktionsvoraussetzung die Behauptung:

(3) Wenn $\mathcal{H}$ ein π-reduzierter, spannender aussagenlogischer Klauselgraph mit $\text{Atome}(\mathcal{H}) \subset \text{Atome}(\mathcal{G})$ ist, dann existiert eine T-Widerlegung für $\mathcal{H}$.

Sei also $\mathcal{H}$ ein π-reduzierter, spannender aussagenlogischer Klauselgraph mit $\text{Atome}(\mathcal{H}) \subset \text{Atome}(\mathcal{G})$. Wir zeigen durch vollständige Induktion über $|\text{Auftr}(\bar{L},\mathcal{H})|$, daß eine T-Widerlegung für $\mathcal{H}$ existiert.

"$|\text{Auftr}(\bar{L},\mathcal{H})| = 0$". Dann gilt $|\text{Atome}(\mathcal{H})| < |\text{Atome}(\mathcal{G})|$. Also ist die äußere Induktionsvorausssetzung auf $\mathcal{H}$ anwendbar und liefert, wie zu zeigen war, eine T-Widerlegung für $\mathcal{H}$.

"$|\text{Auftr}(\bar{L},\mathcal{H})| > 0$".

Fall 1: $\mathcal{H}$ enthält eine unäre Klausel $\{L\}$. Sei a ein Knoten in $\mathcal{H}$ mit $C_a = \{L\}$. Da $\mathcal{H}$ π-reduziert ist, existiert ein Link $\ell = L^{a-b}\bar{L}$ in $\mathcal{H}$. Da $\mathcal{H}$ aussagenlogisch ist, kann ℓ kein τ-Link sein. Sei $\mathcal{H}_1$ ein Klauselgraph mit:

(4) $\mathcal{H} \xrightarrow[\rho T]{\ell} \mathcal{H}_1$.

Fall 1.1: $\mathcal{H}_1$ enthält den Knoten b nicht mehr. Dann gilt $|\text{Auftr}(\bar{L},\mathcal{H}_1)| < |\text{Auftr}(\bar{L},\mathcal{H})|$. Also ist die innere Induktionsvoraussetzung auf $\mathcal{H}_1$ anwendbar und liefert $\mathcal{H}_1 \vdash_T \square$. Wegen (4) existiert also , wie zu zeigen war, eine T-Widerlegung für $\mathcal{H}$.

Fall 1.2: $\mathcal{H}_1$ enthält den Knoten b. Wegen Lemma 3.1.1 enthält $\mathcal{H}_1$ auch den Resolventenknoten c zu ℓ. Da der zweite Inzidenzknoten zu ℓ unär ist, erfüllt b die BSEB in $\mathcal{H}_1$ bzgl. c und ε. Also existiert ein spannender (!), π-reduzierter Graph $\mathcal{H}_2$ mit:

(5) $\mathcal{H}_1 \xrightarrow[\sigma]{b} \mathcal{H}_2$ und $|\text{Auftr}(\bar{L},\mathcal{H}_2)| < |\text{Auftr}(\bar{L},\mathcal{H}_1)| \leq |\text{Auftr}(\bar{L},\mathcal{H})|$.

Somit ist die innere Induktionsvoraussetzung auf $\mathcal{H}_2$ anwendbar und liefert $\mathcal{H}_2 \vdash_T \square$. Wegen (4) und (5) existiert, wie zu zeigen war, eine T-Widerlegung für $\mathcal{H}$.

Fall 2: $\mathcal{H}$ enthält keine unäre Klausel $\{L\}$. $\pi(\mathcal{H}-L)$ ist π-reduziert, wegen Lemma 3.3.4(b) auch spannend und erfüllt $|\text{Atome}(\mathcal{H}-L)| < |\text{Atome}(\mathcal{G})|$. Also ist die äußere

Induktionsvoraussetzung auf $\pi(\mathcal{H}\text{-}L)$ anwendbar, und liefert, daß eine T-Widerlegung für $\pi(\mathcal{H}\text{-}L)$ existiert. Wegen Lemma 3.3.5(c) gilt sogar $\pi(\mathcal{H}\text{-}L) \xrightarrow[\rho T]{*} \square$.
Nun existiert wegen Lemma 3.3.7 ein Graph $\mathcal{H}'$ mit:
(6) $\mathcal{H} \xrightarrow[\rho T]{*} \mathcal{H}'$ und $\mathcal{H}'$ ist spannend und π-reduziert,
auf den einer der folgenden Fälle zutrifft:
Fall 2.1: $\square \in S(\mathcal{H}')$. Dann gilt $\mathcal{H} \vdash_T \square$, wie zu zeigen war.
Fall 2.2: $\{L\} \in S(\mathcal{H}')$. Wenn man für $\mathcal{H}'$ so wie in Fall 1 für $\mathcal{H}$ verfährt, erhält man $\mathcal{H}' \vdash_T \square$. Zusammen mit (6) ergibt sich, wie zu zeigen war, daß eine T-Widerlegung für $\mathcal{H}$ existiert.
Fall 2.3: $|\mathrm{Auftr}(\bar{L},\mathcal{H}')| < |\mathrm{Auftr}(\bar{L},\mathcal{H})|$. Dann ist die innere Induktionsvoraussetzung auf $\mathcal{H}'$ anwendbar und liefert $\mathcal{H}' \vdash_T \square$. Wieder erhält man zusammen mit (6), daß, wie zu zeigen war, eine T-Widerlegung für $\mathcal{H}$ existiert.

Damit haben wir die Behauptung (3) gezeigt. Da sie auf $\mathcal{G}$ anwendbar ist, existiert, wie zu zeigen war, eine T-Widerlegung für $\mathcal{G}$. Also haben wir die Behauptung (1) und damit Satz 3.3.1 bewiesen. []

KAPITEL 4

VOLLSTÄNDIGKEIT BEI UNÄREN KLAUSELMENGEN

Für allgemeine Klauselmengen sind alle Fragen, die die Vollständigkeit von Filtern für den KOWALSKI-Kalkül betreffen, noch offen. Selbst die Vollständigkeit des reinen Kalküls (siehe 2.5.9) ist unbewiesen. Für aussagenlogische Klauselmengen sind immerhin zwei wichtige Ergebnisse gezeigt: Zum einen ist eine vollständige Strategie, nämlich die AE-Strategie, bekannt. Zum anderen konnten wir für die BTSE-Restriktion, die lediglich die Anwendbarkeit der Reduktionsregeln einschränkt, die Konfluenz nachweisen. Was im aussagenlogischen Fall noch fehlt, ist ein hinreichend allgemeines Kriterium für die NOETHER-Eigenschaft von Filtern.

Wie bereits in der Einleitung zu Kapitel 3 ausgeführt, ist es beim KOWALSKI-Kalkül bisher nicht gelungen, Vollständigkeitsergebnisse für den aussagenlogischen Fall auf den prädikatenlogischen Fall zu übertragen. Man benötigt also voraussichtlich eine neue Beweistechnik, die von vornherein auf prädikatenlogischer Ebene arbeitet. Wir glauben diese Technik gefunden zu haben, können sie aber im Rahmen dieser Arbeit nur für die immerhin unentscheidbare Teilklasse der unären Klauselmengen (siehe Abschnitt 4.2) vorführen, die im Automatischen Beweisen von großer praktischer Bedeutung ist (siehe /Ch70/, /Ku72/ und /HW74/). Sie schließt die gut untersuchte HORN-Klasse mit ein, deren Mächtigkeit hinreicht, um z.B. Gruppen oder BOOLsche Algebren zu axiomatisieren.

Unsere Beweistechnik basiert auf der folgenden Idee aus /SS76, SS80/: Der initiale Graph zu einer unerfüllbaren Klauselmenge enthält einen unerfüllbaren, π-reduzierten Subgraphen, den sogenannten Kern. Diesem Kern kann eine nichtnegative, ganzzahlige Komplexität zugeordnet werden. Komplexität Null bedeutet, daß der Kern die leere Klausel enthält. Bei einer Anwendung der Resolutionsregel gibt es zwei Möglichkeiten: Entweder liegt der Resolutionslink außerhalb oder innerhalb des Kerns. Resolution außerhalb des Kerns läßt diesen unberührt. Resolution innerhalb des Kerns transformiert den Kern und reduziert seine Komplexität. Zusammen ergeben diese Annahmen über den Kern, daß jede Ableitungsstrategie, die nur die Resolutionsregel und diese erschöpfend (d.h., jeder erzeugte Link wird nach endlich vielen Schritten wieder gelöscht) benutzt, vollständig ist.

Wenn man diese Idee weiter verfolgt, ergeben sich jedoch zwei Schwierigkeiten: Wie transformiert sich der Kern bei Anwendung der Resolutionsregel und wie sieht ein geeignetes Komplexitätsmaß für den Kern aus?

Ein naheliegender Gedanke ist, den Kern als Projektion seiner Resolutionswiderlegung zu definieren und zu untersuchen, wie sich diese Resolutionswiderlegung bei Resolution innerhalb des Kerns transformiert. Dies stößt jedoch auf unüberwindliche Schwierigkeiten, da die Struktur von Resolutionswiderlegungen sehr fein und die Transformationen entsprechend kompliziert sind. Wir beseitigen diese Schwierigkeiten, indem wir die Resolutionswiderlegungen zu Klassen bündeln und die gesamte Klasse durch eine einzige, einfachere Struktur darstellen. Für die Darstellung dieser Klassen eignen sich die Widerlegungsgraphen von SHOSTAK /Sh76/, die insbesondere den Vorteil haben, daß in ihnen nur Varianten von solchen Klauseln vorkommen, die auch im Kern auftreten. Statt der Transformationen des Kerns betrachten wir nun die besser handhabaren Transfor-

mationen des korrespondierenden Widerlegungsgraphen und seiner Projektion in den Kern. Es zeigt sich, daß eine Resolution innerhalb des Kerns einer oder mehreren Resolutionen auf dem korrespondierenden Widerlegungsgraphen entspricht.

Unäre Klauselmengen zeichnen sich nun dadurch aus, daß ihre Resolutionswiderlegungen sogar durch Widerlegungsbäume dargestellt werden können. Bei allgemeinen Klauselmengen sind dagegen Zyklen unvermeidbar, wenn der Widerlegungsgraph nur Varianten von Klauseln des Kerns enthalten soll.

In Abschnitt 4.1 führen wir die für unseren Beweis erforderlichen Widerlegungsbäume ein, die wir reduzierte Klauselbäume oder kurz R-Bäume nennen. Danach betrachten wir in 4.2 unäre Klauselmengen und stellen den Zusammenhang mit den R-Bäumen her. Die Projektion der R-Bäume in den Kern und die Transformationen bei den verschiedenen Ableitungsschritten betrachten wir in Abschnitt 4.3. Nach diesen Vorbereitungen können wir in 4.4 die eigentlich anvisierten Vollständigkeitssätze für unäre Klauselmengen zeigen.

Unsere Untersuchungen schließen die Eliminationsregeln für Tautologien und Subsumtionen mit ein. Ein Hauptvorteil der neuen Beweismethode ist, daß sie die Funktion dieser Reduktionsregeln transparent macht. Insbesondere erweisen sich die BIBELschen Eliminationsbedingungen auch vor diesem Hintergrund als adäquat. Überraschenderweise zeigt sich jedoch, daß die Tautologie-Eliminationsregel bei unären Klauselmengen, die ihre Faktoren enthalten, ohne Einschränkung angewendet werden darf, ohne die Vollständigkeit zu gefährden. Wie die Beispiele 2.7.2 und 2.7.8 zeigen, ist dies bei allgemeinen Klauselmengen nicht der Fall.

4.1 Reduzierte Klauselbäume

Reduzierte Klauselbäume sind spezielle Klauselgraphen, die von der Resolutionsregel wieder in reduzierte Klauselbäume überführt werden. Die Einschränkung der durch die Resolutionsregel erzeugten Relation $\xrightarrow[\rho]{}$ auf reduzierte Klauselbäume ist konfluent und noethersch. Die Normalform jedes reduzierten Klauselbaums bezüglich dieser Relation ist der Graph (□). Reduzierte Klauselbäume enthalten keine Tautologien und keine τ-Links. Am Ende des Abschnitts führen wir eine spezielle Subsumtionsregel für reduzierte Klauselbäume ein. Diese erlaubt, Klauseln des Baumes durch subsumierende Klauseln von außerhalb zu ersetzen.

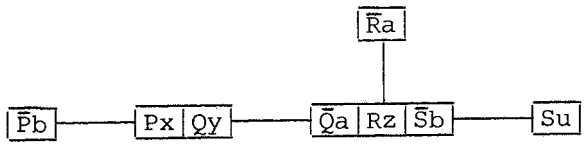
4.1.1 Pfade. Sei $\mathcal{G}$ ein Klauselgraph. Eine Folge $a_1 \ldots a_n$ von Knoten in $\mathcal{G}$ mit $n \geq 1$ heißt Pfad in $\mathcal{G}$, wenn es paarweise verschiedene Links $\ell_1, \ldots, \ell_{n-1}$ in $\mathcal{G}$ gibt, so daß ℓ_i die Knoten a_i und a_{i+1} verbindet. a_1 heißt der Anfang, a_n das Ende und n-1 die Länge des Pfades $a_1 \ldots a_n$. Ein Pfad $a_1 \ldots a_n$ verbindet die Knoten a und b, wenn $\{a_1, a_n\} = \{a, b\}$.

4.1.2 Zusammenhang. Ein Klauselgraph $\mathcal{G}$ heißt zusammenhängend, wenn für je zwei Knoten a und b in $\mathcal{G}$ stets ein Pfad in $\mathcal{G}$ existiert, der a mit b verbindet. Der leere Klauselgraph () und die Graphen (□) sind zusammenhängend.

4.1.3 Komponenten. Sei $\mathcal{G}$ ein Klauselgraph. Eine Komponente von $\mathcal{G}$ ist ein maximaler (bzgl. der Inklusionsordnung) zusammenhängender Subgraph von $\mathcal{G}$. Jeder nichtleere Klauselgraph hat wenigstens eine Komponente. Ein Klauselgraph ist zusammenhängend genau dann, wenn er höchstens eine Komponente hat.

4.1.4 Zyklen. Sei $\mathcal{G}$ ein Klauselgraph und a ein Knoten in $\mathcal{G}$. Ein Pfad in $\mathcal{G}$, der länger als Null ist und der a mit a verbindet, heißt Zyklus in $\mathcal{G}$. $\mathcal{G}$ heißt zyklisch, wenn es einen Zyklus in $\mathcal{G}$ gibt. $\mathcal{G}$ heißt azyklisch, wenn $\mathcal{G}$ nicht zyklisch ist. Offensichtlich kann ein azyklischer Klauselgraph keine Autolinks enthalten.

4.1.5 Klauselbäume. Ein zusammenhängender, azyklischer, nichtleerer (!) Klauselgraph $\mathcal{B}$ heißt Klauselbaum, wenn jedes Literalauftreten in $\mathcal{B}$ mit genau einem Link inzidiert. Ein Beispiel für einen Klauselbaum ist:



Klauselbäume sind offensichtlich stets π-reduziert. Wenn $\mathcal{B}$ ein Klauselbaum mit $|\mathcal{B}| = 1$ ist, dann gilt stets $\mathcal{B} = (\square)$. Ein Klauselbaum mit mehr als einem Knoten enthält mindestens einen unären Link. Dabei heißt ein Link unär, wenn wenigstens eine seiner Inzidenzklauseln unär ist.

4.1.6 Unifizierbare Klauselgraphen. Ein Klauselgraph $\mathcal{G}$ heißt unifizierbar, wenn es eine Substitution ξ gibt, so daß für jeden Link $L^a K^b$ in $\mathcal{G}$ gilt, daß ξ ein Unifikator von L und $\bar{K}$ ist. ξ heißt dann Unifikator für $\mathcal{G}$. Ein Unifikator für den Klauselgraphen in 4.1.5 ist z.B. $\{x/b, y/a, z/a, u/b\}$.

4.1.7 Satz. Sei $\mathcal{G}$ ein unifizierbarer Klauselgraph. Dann existiert genau ein normaler allgemeinster Unifikator für $\mathcal{G}$. Den normalen allgemeinsten Unifikator für $\mathcal{G}$ bezeichnen wir, falls er existiert, mit $\mathrm{nau}(\mathcal{G})$. Es gilt stets:
$\mathrm{Var}(\mathrm{nau}(\mathcal{G})) \subset \mathrm{Var}(\mathcal{G})$.

Beweis. Sei $\mathcal{G}$ ein Klauselgraph. Wenn $\mathcal{G}$ keine Links hat, dann ist offensichtlich ε der normale allgemeinste Unifikator von $\mathcal{G}$. Seien also $\ell_1, \ldots, \ell_n$, wobei $n \geq 1$ ist, die Links von $\mathcal{G}$. Für jeden Link ℓ_i seien $s_{i1}, \ldots, s_{ik_i}$ und $t_{i1}, \ldots, t_{ik_i}$ die eindeutig bestimmten Terme, so daß ein Prädikatensymbol P existiert, für das $Ps_{i1} \ldots s_{ik_i}$ und $\sim Pt_{i1} \ldots t_{ik_i}$ die Inzidenzliterale von ℓ_i sind. Wir definieren $s_i := f^{k_i} s_{i1} \ldots s_{ik_i}$, $t_i := f^{k_i} t_{i1} \ldots t_{ik_i}$, $s := f^n s_1 \ldots s_n$ und $t := f^n t_1 \ldots t_n$. Offensichtlich ist ξ genau dann ein Unifikator von $\mathcal{G}$, wenn ξ ein Unifikator für s und t ist. Der Unifikationssatz 1.2.11 liefert nun alle Aussagen des Satzes. []

4.1.8 Satz. Sei $\mathcal{B}$ ein Klauselbaum. Dann gilt:

(a) Zu zwei Knoten a und b in $\mathcal{B}$ gibt es stets genau einen Pfad in $\mathcal{B}$, der mit a anfängt und der mit b endet.

(b) Zu einem Pfad $\pi = a_1 \ldots a_n$ in $\mathcal{B}$ gibt es stets eindeutig bestimmte Links $\ell_1, \ldots, \ell_{n-1}$ in $\mathcal{B}$, so daß ℓ_i die Knoten a_i und a_{i+1} verbindet. Die Links $\ell_1, \ldots, \ell_{n-1}$ heißen die zu π gehörigen Links.

Beweis. Trivial. []

4.1.9 <u>τ-Pfade in unifizierbaren Klauselbäumen</u>. Sei $\mathcal{B}$ ein Klauselbaum und sei $\pi = a_1 \ldots a_n$ ein Pfad in $\mathcal{B}$. Seien $\ell_1, \ldots, \ell_{n-1}$ die zu π gehörigen Links. Ein Literalauftreten L^a heißt

- <u>Anfang von π</u>, wenn $n \geq 2$ und $a = a_1$ und $L^a \in \ell_1$.
- <u>Ende von π</u>, wenn $n \geq 2$ und $a = a_n$ und $L^a \in \ell_{n-1}$.

Offensichtlich hat π, wenn $n \geq 2$ ist, genau ein Literalauftreten als Anfang und genau ein Literalauftreten als Ende.

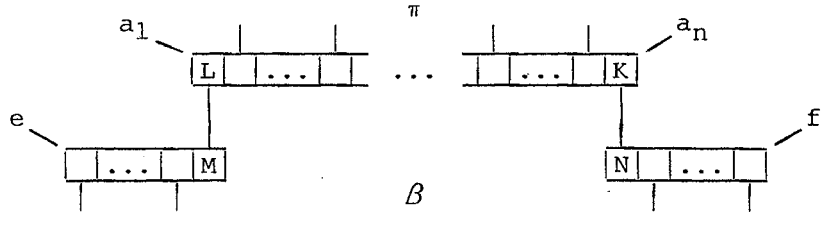
Sei ξ ein Unifikator für $\mathcal{B}$. π heißt <u>τ-Pfad für ξ bzgl. L und K</u>, wenn die folgenden Bedingungen erfüllt sind:

(a) $L \in C_{a_1}$ und $K \in C_{a_n}$ und $\xi L = \xi \bar{K}$.

(b) L^{a_1} ist nicht der Anfang von π.

(c) K^{a_n} ist nicht das Ende von π.

Sei π ein τ-Pfad für ξ bzgl. L und K. Die folgende Skizze gibt eine anschauliche Darstellung von π:



Man überlegt sich leicht, daß die Literalauftreten M^e und N^f in $\mathcal{B}$ eindeutig bestimmt sind, und daß $\xi M = \xi \bar{N}$ gilt. π läßt sich aus $\mathcal{B}$ eliminieren, indem man zu $\mathcal{B}' := \pi((\mathcal{B} + M^e N^f) - M^e L^{a_1})$ übergeht. $\mathcal{B}'$ ist wieder ein Klauselbaum mit ξ als Unifikator. Offensichtlich gilt $S(\mathcal{B}') \subset S(\mathcal{B})$ und $|\mathcal{B}'| < |\mathcal{B}|$.

4.1.10 <u>R-Bäume</u>. Ein Klauselbaum $\mathcal{B}$ heißt <u>reduziert</u> oder R-Baum, wenn $\mathcal{B}$ unifizierbar ist und wenn es in $\mathcal{B}$ keine τ-Pfade für nau($\mathcal{B}$) gibt. Ein Klauselbaum $\mathcal{B}$ heißt <u>reduziert für ξ</u>, wenn ξ ein Unifikator für $\mathcal{B}$ ist und wenn in $\mathcal{B}$ keine τ-Pfade für ξ existieren. Offensichtlich ist jeder Klauselbaum $\mathcal{B}$, für den ein Unifikator ξ existiert, so daß $\mathcal{B}$ für ξ reduziert ist, insbesondere ein R-Baum. Die Graphen () und (□) sind R-Bäume. Auch der Graph in 4.1.5 ist ein R-Baum.

4.1.11 <u>Satz</u>. Zu jedem unifizierbaren KLauselbaum $\mathcal{B}$ existiert ein R-Baum $\mathcal{B}'$ mit $S(\mathcal{B}') \subset S(\mathcal{B})$ und $|\mathcal{B}'| \leq |\mathcal{B}|$.

Beweis. Sei $\mathcal{B}$ ein KLauselbaum und ξ ein Unifikator für $\mathcal{B}$. Mit der in 4.1.9 beschriebenen Reduktion lassen sich alle τ-Pfade für ξ aus $\mathcal{B}$ eliminieren, da bei jeder Elimination die Anzahl der Knoten echt kleiner wird. Man erhält so schließlich einen Klauselbaum $\mathcal{B}'$, der für ξ reduziert ist, und der $S(\mathcal{B}') \subset S(\mathcal{B})$ und $|\mathcal{B}'| \leq |\mathcal{B}|$ erfüllt. []

4.1.12 <u>Proposition</u>. Ein R-Baum enthält keine Tautologie.

Beweis. Sei $\mathcal{B}$ ein Klauselbaum, der für ξ reduziert ist.
Angenommen, $\mathcal{B}$ enthält eine Tautologie. Dann gibt es einen Knoten a in $\mathcal{B}$ und ein Atom L mit $L \in C_a$ und $\bar{L} \in C_a$. Offensichtlich ist a ein τ-Pfad in $\mathcal{B}$ für ξ bzgl. L und $\bar{L}$. Also ist $\mathcal{B}$ nicht für ξ reduziert. *W!* []

4.1.13 <u>Satz</u> (Resolution auf R-Bäumen). Sei $\mathcal{B} \xrightarrow[\rho]{\ell} \mathcal{B}'$ und sei $\mathcal{B}$ ein R-Baum. Dann ist $\mathcal{B}'$ ebenfalls ein R-Baum, der den Resolventenknoten zu ℓ enthält, und der die Inzidenzknoten zu ℓ nicht mehr enthält. Es gilt also $|\mathcal{B}| > |\mathcal{B}'|$.

Beweis. Sei $\mathcal{B} \xrightarrow[\rho]{\ell} \mathcal{B}'$ und sei $\mathcal{B}$ ein R-Baum. Sei

$\ell = L_o^a K_o^b$.

$\mathcal{G}$ ein Klauselgraph mit $\mathcal{B} \xrightarrow[\rho_o]{\ell} \mathcal{G}$ und $\mathcal{B}' = \pi(\mathcal{G} - \ell)$.

- c der Resolventenknoten zu ℓ in $\mathcal{G}$.

$\sigma := \mathrm{nau}(L_o, \bar{K}_o)$.

- θ eine U-Substitution für $\mathrm{Res}(C_a, L_o; C_b, K_o)$ mit $C_c = \theta \mathrm{Res}(C_a, L_o; C_b, K_o)$.

$\xi := \mathrm{nau}(\mathcal{B})$.

$\zeta := \xi\theta^{-1}$.

Dann gilt:

(1) $\mathcal{B}$ ist reduziert für ζ. (da $\zeta|_{\mathrm{Var}(\mathcal{B})} = \xi$)

(2) $\xi = \xi * \sigma = \xi\sigma$. (Lemma 1.2.16(d))

(3) Wenn $L \in C_c$ und K^d der signifikante Vorgänger zu L^c ist, dann gilt $\zeta K = \zeta L$.

Aussage (3) ergibt sich wie folgt:

$$\begin{aligned} \zeta L = (\xi\theta^{-1})(\theta\sigma K) &= \xi\sigma K && \text{, da } \mathrm{Ber}(\theta^{-1}) \cap \mathrm{Var}(\sigma K) = \emptyset. \\ &= \xi K && \text{wegen (2).} \\ &= \zeta K && \text{, da } \zeta|_{\mathrm{Var}(K)} = \xi. \end{aligned}$$

Wir zeigen nun den Satz in vier Teilen:

Teil 1: ζ ist ein Unifikator für $\mathcal{G}$ und für $\mathcal{B}'$. Es genügt zu zeigen, daß ζ ein

Unifikator für $\mathcal{G}$ ist, da $\mathcal{B}'$ ein Subgraph von $\mathcal{G}$ ist. Sei also $M^e N^f$ ein Link in $\mathcal{G}$. Wir müßen zeigen, daß $\zeta M = \zeta\bar{N}$ ist.

Fall 1.1: $e \neq c$ und $f \neq c$. Dann ist $M^e N^f$ Link in $\mathcal{B}$. Da ζ Unifikator für $\mathcal{B}$ ist, gilt $\zeta M = \zeta\bar{N}$.

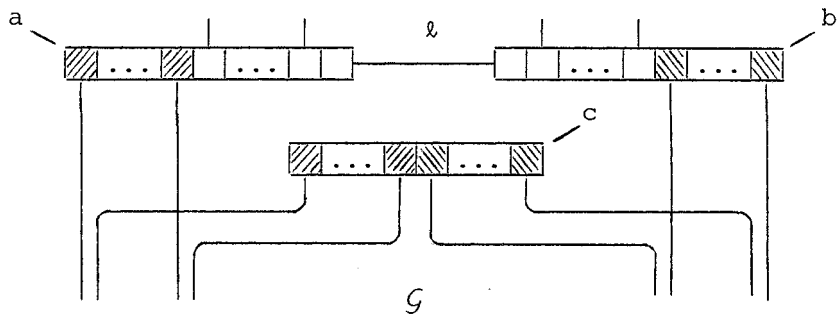
Fall 1.2: $e = c$ und $f \neq c$. Sei K^d der signifikante Vorgänger zu Me Wegen Lemma 2.3.6 ist $K^d N^f$ Link in $\mathcal{G}$. Fall 1.1 liefert, daß $\zeta K = \zeta\bar{N}$ ist. Wegen (3) gilt, wie zu zeigen war, $\zeta M = \zeta\bar{N}$.

Fall 1.3: $e \neq c$ und $f = c$. Analog zu Fall 1.2.

Fall 1.4: $e = f = c$. Sei K^d der signifikante Vorgänger zu Me Wegen Lemma 2.3.6 ist $K^d N^f$ Link in $\mathcal{G}$. Fall 1.3 liefert, daß $\zeta K = \zeta\bar{N}$. Wegen (3) gilt wieder $\zeta M = \zeta\bar{N}$.

Teil 2: Bei $\mathcal{B} \xrightarrow[\rho_0]{\ell} \mathcal{G}$ wird jeder Link, der mit einem signifikanten Vorgänger inzidiert, vererbt. Sei also $L \in C_c$ und sei K^d der signifikante Vorgänger zu L^c Sei $M^e K^d$ ein Link in $\mathcal{G}$. Wir müßen zeigen, daß $M^e L^c$ ein Link in $\mathcal{G}$ ist. Dafür ist, wegen Lemma 2.3.6, hinreichend, daß M und L potentiell komplementär sind. Da ζ ein Unifikator für $\mathcal{G}$ ist, gilt $\zeta M = \zeta\bar{K}$. Wegen (3) gilt dann $\zeta M = \zeta\bar{L}$.

Teil 3: $\mathcal{B}'$ ist ein Klauselbaum, der c enthält und der a und b nicht enthält. Da $\mathcal{B}' = \pi(\mathcal{G} - \ell)$ ist, verifiziert man die Behauptung leicht mit Hilfe der folgenden Skizze, die $\mathcal{G}$ darstellt:



Die signifikanten Vorgänger sind schraffiert eingezeichnet. In Teil 2 haben wir gezeigt, daß alle Links, die mit einem signifikanten Vorgänger inzidieren, an die Resolvente vererbt werden, so daß $\mathcal{G}$ die obige Struktur haben muß.

Teil 4: $\mathcal{B}'$ ist reduziert für ζ.

Angenommen, $\pi' = a_1 \ldots a_n$ ist ein τ-Pfad in $\mathcal{B}'$ für ζ bzgl. L und K.

Fall 4.1: Für alle i ist $a_i \neq c$. Dann ist π' τ-Pfad in $\mathcal{B}$ für ζ bzgl. L und K. Das kann wegen (1) nicht sein. *W!*

Fall 4.2: $n = 1$ und $a_1 = c$. Dann sind $L, K \in C_c$. Seien M^e und N^f die signifikanten

Vorgänger zu L^c und K^c Da $\zeta L = \zeta\bar{K}$ gilt, gilt wegen (3) auch $\zeta M = \zeta\bar{N}$.

Fall 4.2.1: $e = f$. Dann ist e ein τ-Pfad in $\mathcal{B}$ für ζ bzgl. M und N. *W!*

Fall 4.2.2: $e \neq f$. Dann ist ef ein τ-Pfad in $\mathcal{B}$ für ζ bzgl. M und N. *W!*

Fall 4.3: $n > 1$ und es gibt ein i mit $a_i = c$. Seien $\ell_1, \ldots, \ell_{n-1}$ die zu π' gehörigen Links in $\mathcal{B}'$.

Fall 4.3.1: $a_1 = c$. Dann ist $L \in C_c$. Seien

- M^e der signifikante Vorgänger zu L^c
- N^f der signifikante Vorgänger des Literalauftretens in $a_1 = c$, das mit ℓ_1 inzidiert.

Da $\zeta L = \zeta\bar{K}$ ist, gilt wegen (3) auch $\zeta M = \zeta\bar{K}$.

Fall 4.3.1.1: $e = f$. Dann ist $ea_2 \ldots a_n$ τ-Pfad in $\mathcal{B}$ für ζ bzgl. M und K. *W!*

Fall 4.3.1.2: $e \neq f$. Dann ist $efa_2 \ldots a_n$ τ-Pfad in $\mathcal{B}$ für ζ bzgl. M und K. *W!*

Fall 4.3.2: $a_n = c$. Analog zu Fall 4.3.1.

Fall 4.3.3: $a_i = c$ und $1 < i < n$. Seien

- M^e der signifikante Vorgänger des Literalauftretens in $a_i = c$, das mit ℓ_{i-1} inzidiert.
- N^f der signifikante Vorgänger des Literalauftretens in $a_i = c$, das mit ℓ_i inzidiert.

Fall 4.3.3.1: $e = f$. Dann ist $a_1 \ldots a_{i-1} e a_{i+1} \ldots a_n$ τ-Pfad in $\mathcal{B}$ für ζ bzgl. L und K. *W!*

Fall 4.3.3.2: $e \neq f$. Dann ist $a_1 \ldots a_{i-1} e f a_{i+1} \ldots a_n$ ein τ-Pfad in $\mathcal{B}$ für ζ bzgl. L und K. *W!* []

4.1.14 <u>Korollar</u>. Sei $\mathcal{B}$ ein R-Baum. Dann gilt:

(a) $\mathcal{B}$ enthält keine τ-Links.

(b) $\mathcal{B} \xrightarrow[\rho]{*} (\square)$, wobei nur auf unären Non-τ-Links resolviert wird.

(c) Jeder unifizierbare Klauselbaum ist unerfüllbar.

Beweis. Sei $\mathcal{B}$ ein R-Baum.

Teil (a). Angenommen, ℓ ist ein τ-Link in $\mathcal{B}$. Sei $\mathcal{B}'$ ein Klauselgraph mit $\mathcal{B} \xrightarrow[\rho]{\ell} \mathcal{B}'$. Wegen Satz 4.1.13 ist $\mathcal{B}'$ ein R-Baum, der die Resolvente zu ℓ enthält. Also enthält $\mathcal{B}'$ eine Tautologie. Das kann wegen Proposition 4.1.12 nicht sein. *W!*

Tei (b). Jeder Klauselbaum besitzt mindestens einen Knoten. Wenn $\mathcal{B}$ nur einen Knoten enthält, dann ist $\mathcal{B} = (\square)$. Wenn $\mathcal{B}$ mehr als einen Knoten enthält, dann enthält $\mathcal{B}$ auch einen unären Link ℓ (siehe 4.1.5). Wegen (a) ist ℓ sogar ein

unärer Non-τ-Link. Wegen Satz 4.1.13 existiert ein R-Baum $\mathcal{B}'$ mit $\mathcal{B} \xrightarrow[\rho]{\ell} \mathcal{B}'$ und $|\mathcal{B}'| < |\mathcal{B}|$. Also gibt es eine Widerlegung $\mathcal{B} \xrightarrow[\rho]{*} (\square)$, bei der nur auf unären Non-τ-Links resolviert wird.

Teil (c). Wegen Satz 4.1.11 genügt es zu zeigen, daß R-Bäume unerfüllbar sind. Sei also $\mathcal{B}$ ein R-Baum. Wegen (b) gilt, daß $\mathcal{B} \xrightarrow[\rho]{*} (\square)$. Also gilt $\mathcal{B} \vdash \square$ im KOWALSKI-Kalkül. Die Korrektheit des KOWALSKI-Kalküls liefert nun, daß $\mathcal{B}$ unerfüllbar ist. []

4.1.15 <u>Die σ_B-Regel für R-Bäume</u>. Die zweite und letzte Ableitungsoperation für R-Bäume ist die σ_B-Regel. Sie erlaubt, eine Klausel C_a im Baum durch eine neue Klausel, die C_a subsumiert, zu ersetzen.

Algorithmus σ_B;

Eingabe: $\mathcal{B}$: R-Baum;
a : Knoten in $\mathcal{B}$;
D : Klausel, die variablendisjunkt zu $\mathcal{B}$ ist;
θ : S-Substitution von D nach C_a;

Konstante: d : Knoten, der nicht in $\mathcal{B}$ vorkommt;

Variablen: $\mathcal{G}$: Klauselgraph := $\mathcal{B}$;
K, L, M : $\mathbb{L}$; e : $\mathbb{K}$;

1: $\mathcal{G} := \mathcal{G} + \langle d,D\rangle$;

2: Für alle $L \in C_d$

2.1: Wähle $K \in C_a$ mit $K = \theta L$; -- eindeutig bestimmt.

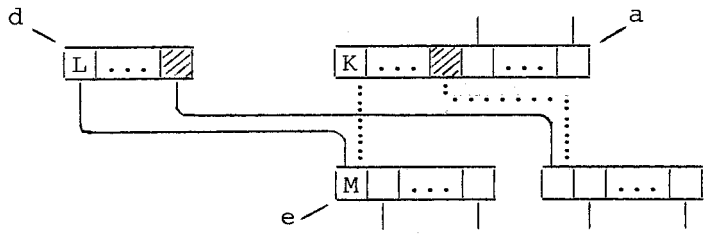
2.2: Wähle M, e mit $M^e K^a$ ist Link in $\mathcal{G}$; -- eindeutig bestimmt.

2.3: $\mathcal{G} := (\mathcal{G} - M^e K^a) + M^e L^d$; -- nau(M,$\bar{K}$) existiert ==> nau(M,$\bar{L}$) existiert.

3: $\mathcal{G} := \pi(\mathcal{G})$;

Ausgabe: $\mathcal{G}$.

Die folgende Skizze zeigt $\mathcal{G}$ direkt vor der π-Reduktion in Schritt 3:



Die gepunkteten Links wurden von σ_B gelöscht. Daß die von σ_B neu erzeugten Links an d auch wirklich potentiell komplementäre Literale verbinden, folgt aus Lemma 1.2.14, da $K = \theta L$ gilt.

Wie bisher schreiben wir:

$\mathcal{B} \xrightarrow[\sigma_B]{a,D,\theta} \mathcal{B}'$, wenn $\langle \mathcal{B}, a, D, \theta \rangle$ eine korrekte Eingabe für σ_B ist, und wenn $\mathcal{B}'$ als Ausgabe zu dieser Eingabe erhalten werden kann.

- $\mathcal{B} \xrightarrow[\sigma_B]{} \mathcal{B}'$, wenn es a, D, θ mit $\mathcal{B} \xrightarrow[\sigma_B]{a,D,\theta} \mathcal{B}'$ gibt.

4.1.16 <u>Satz</u>. Sei $\mathcal{B} \xrightarrow[\sigma_B]{} \mathcal{B}'$ und sei $\mathcal{B}$ ein R-Baum. Dann ist auch $\mathcal{B}'$ ein R-Baum mit $|\mathcal{B}| \leq |\mathcal{B}'|$.

Beweis. Sei $\mathcal{B} \xrightarrow[\sigma_B]{a,D,\theta} \mathcal{B}'$. Sei

- $\xi := \mathrm{nau}(\mathcal{B})$.
- $\zeta := \xi\theta$.

Dann gilt offensichtlich:

(1) $\zeta|_{\mathrm{Var}(\mathcal{B})} = \xi$.

(2) $\mathcal{B}$ ist reduziert für ζ.

(3) Wenn $L \in D$ und $K \in C_a$ mit $K = \theta L$ ist, dann gilt $\zeta K = \zeta L$.

Wir zeigen nun den Satz in drei Teilen.

Teil 1: $\mathcal{B}'$ ist ein Klauselbaum mit $|\mathcal{B}| \leq |\mathcal{B}'|$. Dies ergibt sich sofort aus der Skizze und der Bemerkung in 4.1.15.

Teil 2: ζ ist ein Unifikator für $\mathcal{B}'$. Wir müssen für jeden Link ℓ in $\mathcal{B}'$ zeigen, daß ζ die Literale von ℓ echt komplementär macht. Sei also ℓ ein Link in $\mathcal{B}'$. Wenn ℓ auch ein Link in $\mathcal{B}$ ist, folgt die Behauptung sofort aus (2). Wenn ℓ kein Link in $\mathcal{B}$ ist, gibt es M, e und L, so daß $\ell = L^d M^e$ und $e \notin \{a,d\}$ ist. Sei $K \in C_a$ das eindeutig (wegen $|D| = |\theta D|$) bestimmte Literal mit $K = \theta L$. Da $K^a M^e$ ein Link in $\mathcal{B}$ ist, gilt $\zeta K = \zeta \bar{M}$ wegen (2). Wegen (3) gilt $\zeta L = \zeta \bar{M}$, wie zu zeigen war.

Teil 3: β' ist reduziert für ζ.

Angenommen, $\pi' = a_1 \ldots a_n$ ist ein τ-Pfad in β' für ζ bzgl. L und K.

Fall 3.1: Für alle i ist $a_i \neq d$. Dann ist π' τ-Pfad in β für ζ bzgl. L und K. Das kann wegen (2) nicht sein. *W!*

Fall 3.2: $n = 1$ und $a_1 = d$. Dann sind $L, K \in C_d$. Seien $M, N \in C_a$ (in β) die Literale mit $M = \theta L$ und $N = \theta K$. Da $\zeta L = \zeta \bar{K}$ ist, gilt wegen (3), daß $\zeta M = \zeta \bar{N}$ ist. Also ist a ein τ-Pfad in β für ζ bzgl. M und N. *W!*

Fall 3.3: $n > 1$ und es gibt ein i mit $a_i = d$. Seien $\ell_1, \ldots, \ell_{n-1}$ die zu π' gehörigen Links in β'.

Fall 3.3.1: $a_1 = d$. Dann ist $L \in C_d$. Seien

- M^a das eindeutig bestimmte Literalauftreten in β mit $M = \theta L$.
- N_1^d das Literalauftreten in β', welches mit ℓ_1 inzidiert.
- N^a das Literalauftreten in β mit $N = \theta N_1$.

Da $\zeta L = \zeta \bar{K}$ ist, gilt wegen (3) auch, daß $\zeta M = \zeta \bar{K}$ ist. Also ist $aa_2 \ldots a_n$ ein τ-Pfad in β für ζ bzgl. M und K. *W!*

Fall 3.3.2: $a_n = d$. Analog zu Fall 3.3.1.

Fall 3.3.3: $a_i = d$ und $1 < i < n$. Dann ist $a_1 \ldots a_{i-1} a a_{i+1} \ldots a_n$ ein τ-Pfad in β für ζ bzgl. L und K. *W!* []

4.1.17 <u>Lemma</u>. Es gibt keine unendliche Folge $\beta_0 \beta_1 \beta_2 \ldots$ von R-Bäumen, so daß für alle $i \geq 0$ gilt: Es existieren a, D und θ, so daß $\beta_i \xrightarrow[\sigma_B]{a, D, \theta} \beta_{i+1}$ ein <u>reduzierender σ_B-Schritt</u> ist, d.h., daß C_a von D reduzierend subsumiert wird.

Beweis. Angenommen, $\beta_0 \beta_1 \beta_2 \ldots$ ist eine Folge mit den obigen Eigenschaften. Offensichtlich gilt:

(1) $|\beta_0| \geq |\beta_1| \geq \ldots \geq 1$.

Also existiert ein $n \geq 0$ mit:

(2) $|\beta_n| = |\beta_{n+1}| = |\beta_{n+2}| = \ldots$.

Damit wird in jedem Schritt $\beta_i \xrightarrow[\sigma_B]{} \beta_{i+1}$ mit $i \geq 1$ genau ein Knoten a gelöscht und durch einen neuen Knoten d ersetzt, wobei C_a von C_d reduzierend subsumiert wird. Es gibt folglich eine unendliche Folge $C_1 C_2 C_3 \ldots$ von Klauseln, so daß C_i von C_{i+1} reduzierend subsumiert wird. Das kann nach Lemma 2.8.3 nicht sein. *W!*

[]

4.2 Unäre Klauselmengen

4.2.1 Unäre Resolutionsableitungen. Eine Resolutionsableitung $C_1 \ldots C_n$ für C aus S heißt unär (engl: unit), wenn für jedes k mit $1 \leq k \leq n$ gilt: C_k ist eine Variante einer Klausel in S, oder es existieren $i < k$, $j < k$ und Faktoren C bzw. D von C_i bzw. C_j, so daß C unär und C_k eine V-Resolvente von C und D ist.

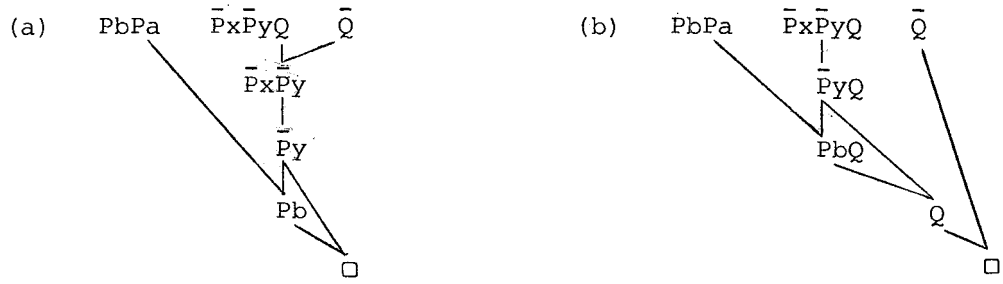
4.2.2 Strikt unäre Resolutionsableitungen. Eine Resolutionsableitung $C_1 \ldots C_n$ für C aus S heißt strikt unär, wenn für jedes k mit $1 \leq k \leq n$ gilt: C_k ist Variante einer Klausel in S, oder es existieren $i < k$ und $j < k$, so daß C_i unär ist und C_k eine V-Resolvente von C_i und C_j ist.

4.2.3 Initiale Resolutionsableitungen. Eine Resolutionsableitung $C_1 \ldots C_n$ für C aus S heißt initial (engl: input), wenn für jedes k mit $1 \leq k \leq n$ gilt: C_k ist eine Variante einer Klausel in S, oder es existieren $i < k$ und $j < k$, so daß C_i eine Variante einer Klausel in S und C_k eine FV-Resolvente von C_i und C_j ist.

4.2.4 Strikt initiale Resolutionsableitungen. Eine Resolutionsableitung $C_1 \ldots C_n$ für C aus S heißt strikt initial, wenn für jedes k mit $1 \leq k \leq n$ gilt: C_k ist eine Variante einer Klausel in S, oder es existieren $i < k$ und $j < k$, so daß C_i eine Variante einer Klausel in S und C_k eine V-Resolvente von C_i und C_j ist.

4.2.5 R-Bäume für S. Sei β ein R-Baum und S eine Klauselmenge. Dann heißt β R-Baum für S, wenn $S(\beta) \subset S$ ist.

4.2.6 Beispiel. Die Klauselmenge $S = \{PbPa, \bar{P}x\bar{P}yQ, \bar{Q}\}$ ist wie folgt unär und initial widerlegbar:



(a) ist eine unäre und (b) ist eine initiale Widerlegung für S. Man sieht unschwer, daß weder eine strikt unäre noch eine strikt initiale Widerlegung noch ein R-Baum für S existieren. Da S keine unären Faktoren besitzt, ist dies auch für S zusammen mit seinen unären Faktoren der Fall. Dagegen ist (b) eine strikt initiale Widerlegung für S zusammen mit seinen Faktoren. Ein R-Baum für S zusammen mit seinen Faktoren ist

(c) $\boxed{\bar{Q}}$ — $\boxed{Q \mid \bar{P}y}$ — $\boxed{Pb \mid Pa}$ — $\boxed{\bar{P}x \mid Q}$ — $\boxed{\bar{Q}}$

oder auch

(d) $\boxed{\bar{Q}}$ — $\boxed{Q \mid \bar{P}u}$ — $\boxed{Pa \mid Pb}$ — $\boxed{\bar{P}x \mid Q \mid \bar{P}y}$ — $\boxed{Pb \mid Pa}$ — $\boxed{\bar{P}z \mid Q}$ — $\boxed{\bar{Q}}$, $\boxed{\bar{Q}}$

4.2.7 Satz (CHANG, HARRISON, RUBIN). Für eine Klauselmenge S sind die folgenden Aussagen äquivalent:

(a) Für S existiert eine unäre Widerlegung.

(b) Für S zusammen mit seinen Faktoren existiert eine tautologiefreie strikt unäre Widerlegung.

(c) Für S existiert eine initiale Widerlegung.

(d) Für S zusammen mit seinen Faktoren existiert eine tautologiefreie strikt initiale Widerlegung.

(e) Für S zusammen mit seinen Faktoren existiert ein R-Baum.

Wir diskutieren zunächst die Aussage und die Implikationen des Satzes und holen dann den Beweis nach.

4.2.8 Unäre Klauselmengen. Eine Klauselmenge S heißt

- UW-Klauselmenge, wenn für S eine unäre Widerlegung existiert.
- UWF-Klauselmenge, wenn S eine UW-Klauselmenge ist, die ihre Faktoren enthält.
- unär, wenn S erfüllbar oder wenn S eine UW-Klauselmenge ist.
- UF-Klauselmenge, wenn S unär ist und seine Faktoren enthält.

Satz 4.2.7 liefert fünf verschiedene Charakterisierungen für UW-Klauselmengen. Die Übereinstimmung zwischen unär und initial widerlegbaren Klauselmengen wurde von CHANG /Ch70/ entdeckt. HARRISON und RUBIN /HR78/ erkannten den Zusammenhang mit den R-Bäumen. Die refutation trees in /HR78/ entsprechen genau unseren unifizierbaren Klauselbäumen. Wir haben die unifizierbaren Klauselbäume und Satz 4.2.7 unabhängig von HARRISON und RUBIN gefunden und

wurden erst später auf deren Arbeit aufmerksam. Während in /HR78/ lediglich die Beweisidee für Satz 4.2.7 skizziert wird, geben wir den Beweis vollständig und detailliert an. Außerdem haben wir den Satz um die Tautologiefreiheit verschärft.

4.2.9 Die repräsentierten Widerlegungen. Sei S eine Klauselmenge und $\mathcal{B}$ ein R-Baum für S. Jede Ableitung $\Gamma: \mathcal{B} \xrightarrow[\rho]{*} (\square)$ erzeugt nach 2.5.7 mindestens eine Widerlegung Γ_R für S. Die Menge

$$\Omega(\mathcal{B}) := \{\Gamma_R \mid \Gamma \text{ ist eine Ableitung } \mathcal{B} \xrightarrow[\rho]{*} (\square)\}$$

heißt die Menge der von $\mathcal{B}$ repräsentierten Widerlegungen. Offensichtlich ist $\Omega(\mathcal{B})$ nichtleer und endlich. Jede Ableitung aus $\Omega(\mathcal{B})$ ist eine tautologiefreie Widerlegung für S ohne Faktorisierung.

4.2.10 Satz. Sei $\mathcal{B}$ ein R-Baum für eine Klauselmenge S. Dann enthält $\Omega(\mathcal{B})$ stets eine tautologiefreie strikt unäre und eine tautologiefreie strikt initiale Widerlegung für S.

Beweis. Sei $\mathcal{B}$ ein R-Baum für eine Klauselmenge S. Wir müssen lediglich zeigen, daß $\Omega(\mathcal{B})$ eine unäre und eine initiale Ableitung enthält, da jede Ableitung in $\Omega(\mathcal{B})$ eine tautologiefreie Widerlegung für S ohne Faktorisierung ist.

Teil 1: $\Omega(\mathcal{B})$ enthält eine unäre Ableitung. Wegen Korollar 4.1.14(b) existiert eine Widerlegung $\Gamma: \mathcal{B} \xrightarrow[\rho]{*} (\square)$, bei der nur auf unären Links resolviert wird. $\Gamma_R \in \Omega(\mathcal{B})$ sei eine von Γ induzierte Resolutionsableitung. Offensichtlich ist Γ_R eine unäre Ableitung.

Teil 2: $\Omega(\mathcal{B})$ enthält eine initiale Widerlegung für S. Satz 4.1.13 garantiert, daß bei einem Resolutionsschritt auf einem R-Baum stets die beiden Elternknoten gelöscht werden, und die Resolvente immer im Folgebaum enthalten ist. Sei $\Gamma: \mathcal{B} \xrightarrow[\rho]{*} (\square)$ eine lineare Ableitung, das heißt, ausgehend von einem beliebigen Link in $\mathcal{B}$ wird nur noch auf Links resolviert, die mit der jeweils zuletzt gebildeten Resolvente inzidieren. Offensichtlich hat jeder Link, auf dem in Γ resolviert wird, mindestens einen Inzidenzknoten, der schon in $\mathcal{B}$ enthalten ist. Also ist jede von Γ induzierte Ableitung $\Gamma_R \in \Omega(\mathcal{B})$ eine initiale Widerlegung für S. [ ]

R-Bäume stellen eine graphentheoretische Charakterisierung für die Unerfüll-

barkeit von unären Klauselmengen dar. Gegenüber einer Resolutionswiderlegung für S hat ein R-Baum $\mathcal{B}$ für S die Eigenschaft, daß in $\mathcal{B}$ bis auf Variantenbildung nur Klauseln aus S vorkommen.

R-Bäume können zu zwei verschiedenen Zwecken benutzt werden. Wir verwenden sie in dieser Arbeit als *Beweiswerkzeug*. Andererseits etablieren sie einen neuen *Beweisbegriff* für die Unerfüllbarkeit unärer Klauselmengen. Eine Beweisprozedur könnte also - statt beispielsweise aus $\mathcal{G}_S$ im KOWALSKI-Kalkül solange neue Graphen zu erzeugen, bis die leere Klausel auftaucht - direkt in $\mathcal{G}_S$ nach einem R-Baum suchen, der dort, wenn S unär widerlegbar ist, in "überlagerter Form" (Präzisierung im nächsten Abschnitt) als Subgraph enthalten sein muß. Der Autor hat für das Karlsruher Markgraf Carl System einen sogenannten Terminator Modul implementiert, der genau diese Suche durchführt. Näheres findet sich in /BEHSSW81/. Da ein einziger R-Baum viele verschiedene Resolutionswiderlegungen darstellt, liegt der Gedanke nahe, daß die Suche nach einem R-Baum effizienter als die Suche nach einer unären Resolutionswiderlegung ist. Die praktischen Ergebnisse mit dem Terminator Modul bestätigen dies.

Die Idee, die Unerfüllbarkeit einer Klauselmenge S durch *Widerlegungsgraphen*, deren Knoten ausschließlich Varianten in S sind, zu charakterisieren, wird im Automatischen Beweisen seit einiger Zeit intensiv erforscht. Frühe Ansätze hierzu sind die *Resolution Graphs* von YATES, RAPHAEL und HART /YRH70/ und die *Refutation Graphs* von SHOSTAK /Sh76/. Während die Widerlegungsgraphen dieser Arbeiten noch primär dazu intendiert sind, als Beweishilfsmittel Ergebnisse zur Vollständigkeit von Strategien und Restriktionen für den Resolutionskalkül zu liefern, erkannten ANDREWS /An76/ und SICKEL /Si76/, daß die Suche nach Widerlegungsgraphen zu Beweisverfahren führt, die eine wahrscheinlich überlegene Alternative zu den auf dem Resolutionskalkül basierenden Beweisprozeduren darstellen. Die Verfolgung dieses Gedankens führte zu den sogenannten *Matrixkalkülen* von ANDREWS /An81/ und BIBEL /Bi82b/. Die Matrixkalküle vermeiden einen der Hauptnachteile des Resolutionskalküls: Die zu beweisende Formel muß nicht mehr in eine Klauselmenge und die damit verbundene SKOLEM-Normalform (siehe 1.3.6) überführt werden. Dadurch wird die durch das Ausmultiplizieren bewirkte Aufblähung der Ausgangsformel vermieden, und die in der Formulierung der Axiome und des Theorems enthaltenen problembezogenen Strukturen bleiben erhalten.

Widerlegungsgraphen, die die Unerfüllbarkeit von allgemeinen Klauselmengen

charakterisieren, kommen nicht mehr ohne Zyklen aus. Unäre Klauselmengen zeichnen sich also gerade dadurch aus, daß sich ihre Unerfüllbarkeit mit baumartigen Graphen charakterisieren läßt. Die Widerlegungsgraphen aus /Sh76/ charakterisieren die Unerfüllbarkeit von aussagenlogischen Klauselmengen. Der Clou dabei ist, daß sie Zyklen nur in sehr eingeschränkter Form zulassen und deren Funktion transparent machen.

Von CHANG /Ch70/ stammt die Idee, einen Beweiser in zwei Teile zu zerlegen. Der eine Teil ist ein sogenannter *Lemma-Generator*, der zu einer Klauselmenge S neue Klauselmengen $S_1, \ldots S_n$ erzeugt, so daß S genau dann unerfüllbar ist, wenn alle S_i unär widerlegbar sind. Der zweite Teil ist dann ein spezieller Beweiser für unäre Klauselmengen. Es ist klar, daß ein Beweiser, der nur für unäre Klauselmengen vollständig sein muß, viel effizienter sein kann als ein Beweiser, der für allgemeine Klauselmengen vollständig ist. Um aber den Lemma-Generator bauen zu können, ist ein Kriterium notwendig, mit dem sich entscheiden läßt, ob eine Klauselmenge unär ist. Die sogenannte *HORN-Eigenschaft* ist ein dafür hinreichendes Kriterium.

4.2.11 HORN-Mengen. Eine Klausel, die höchstens ein positives Literal enthält, heißt HORN-Klausel (siehe /Ho51/). Eine Klauselmenge, die nur aus HORN-Klauseln besteht heißt HORN-Menge.

4.2.12 Satz (HENSCHEN, WOS). Jede unerfüllbare HORN-Menge ist strikt unär widerlegbar.

Beweis. Siehe /HW74, Theorem 1/. []

Eine Klauselmenge heißt HORN-umbenennbar, wenn sie sich zu einer HORN-Menge umbenennen läßt. Für eine präzise Definition von "umbenennen" verweisen wir auf /Le78/. Man kann allgemeiner als Satz 4.2.12 zeigen, daß jede unerfüllbare Klauselmenge, die HORN-umbenennbar ist, auch strikt unär widerlegbar ist. Dagegen ist nicht jede strikt unär widerlegbare Klauselmenge HORN-umbenennbar. Ein Beispiel dafür ist $\{\bar{Q}b,\ Qa,\ \bar{Q}x\bar{Q}yPxPy,\ Qb\bar{P}a\}$. Die HORN-umbenennbaren Klauselmengen sind also eine echte Unterklasse der unären Klauselmengen. LEWIS gibt in /Le78/ einen effizienten Algorithmus an, der entscheidet, ob eine Klauselmenge HORN-umbenennbar ist und im positiven Fall eine entsprechende Umbenennung liefert.

4.2.13 Satz. Die Unerfüllbarkeit von unären Klauselmengen ist unentscheidbar.

Beweis. HERMES /He65/ und REYNOLDS /Re70/ zeigen, daß die Erfüllbarkeit von HORN-Mengen unentscheidbar ist. Da jede HORN-Menge eine unäre Klauselmenge ist, muß auch für unäre Klauselmengen die Erfüllbarkeit unentscheidbar sein.[]

Viele mathematische Theorien lassen sich durch HORN-Mengen axiomatisieren. Dazu gehören die Axiome für Gruppen, Ringe und BOOLsche Algebren. Weitere Beispiele findet man in /Co65/. In PROLOG /VK76, Ko79/ nutzt man aus, daß HORN-Mengen als indeterministische Algorithmen aufgefaßt werden können. Die Maschine, auf der solche Programme ablaufen können, ist ein spezieller Beweiser für HORN-Mengen.

Wir holen nun den Beweis von Satz 4.2.7 nach. Dafür benötigen wir zunächst das folgende Lemma, welches zeigt, daß man bei unären Widerlegungen mit den initialen Faktoren auskommt.

4.2.14 Lemma. Seien C, D und E Klauseln und L und K Literale mit $E = Res(C,L;D,K)$ und $D = \{K\}$. E' sei ein Faktor von E. Dann gibt es einen Faktor C' von C, ein Literal $L' \in C'$ und eine Substitution ξ mit:

(a) ξ ist ein normaler Unifikator für L' und $\bar{K}$.

(b) $Var(\xi) \subset Var(C \cup D)$.

(c) $|C'| = |\xi C'|$.

(d) $\xi(C' - L') \subset E'$.

Beweis. Sei $\sigma := nau(L,\bar{K})$ und sei τ eine F-Substitution für E mit $E' = \tau E$. Da $Ber(\sigma) \cap Var(\tau) = \emptyset$ ist, liefert Lemma 1.2.16(c), daß gilt:

(1) $\tau\sigma = \tau * \sigma$.

Also ist $\tau\sigma$ normal. Wir definieren $\xi := \tau\sigma$. Lemma 1.4.3 sichert die Existenz einer F-Substitution θ für C mit:

(2) $|\theta C| = |\xi C|$

(3) $\xi = \xi * \theta$.

Zusammen ergeben (1) und (3):

(4) $\xi = \tau\sigma = \tau * \sigma = \tau * \sigma * \theta$.

Wir definieren $L' := \theta L$ und $C' := \theta C$.

Teil (a). $\xi L' = \xi\theta L = \xi L$ wegen (4) und Lemma 1.2.16(b)

$= \xi\sigma L$ wegen (4) und Lemma 1.2.16(b)

$= \xi\sigma\bar{K}$ da $\sigma = nau(L,\bar{K})$

$= \xi\bar{K}$ wegen (4) und Lemma 1.2.16(b)

Teil (b). $Var(\xi) \subset Var(\tau) \cup Var(\sigma) \subset Var(E) \cup Var(L) \cup Var(K) \subset Var(C \cup D)$.

Teil (c). $|C'| = |\theta C| \overset{(2)}{=} |\xi C| \overset{(4)}{=} |\xi\theta C| = |\xi C'|$.

Teil (d). $\xi(C' - L') \overset{(c)}{=} \xi C' - \xi L' = \xi\theta C - \xi\theta L \overset{(4)}{=} \xi C - \xi L = \tau\sigma C - \tau\sigma L$

$\subset \tau(\sigma C - \sigma L) = \tau E = E'$. []

4.2.15 <u>Lemma</u>. Sei S eine Klauselmenge, zu der eine unäre Widerlegung existiert. Dann existiert für S zusammen mit seinen Faktoren ein R-Baum.

Beweis. Sei S eine Klauselmenge und sei Ω eine unäre Widerlegung für S. S_F bezeichne S zusammen mit seinen Faktoren. Wegen Satz 4.1.11 genügt es, zu zeigen, daß für S_F ein unifizierbarer Klauselbaum existiert.

Sei m die Anzahl der Resolventen in Ω, die keine Klauseln in S_F sind. Wir zeigen durch vollständige Induktion über m, daß ein unifizierbarer Klauselbaum β für S_F existiert.

"m = 0". Dann ist $\square \in S$. Folglich ist $(\square)$ ein unifizierbarer Klauselbaum für S_F.

"m > 0". Sei E die erste Resolvente in Ω, die keine Klausel in S_F ist. Es gibt dann Klauseln C und D und Literale L und K mit: C und D sind Varianten von Klauseln in S_F, $D = \{K\}$ und $E = Res(C,L;D,K)$. Auf Ω und $S \cup \{E\}$ ist die Induktionsvoraussetzung anwendbar. Es existiert also ein unifizierbarer Klauselbaum β, so daß für jeden Knoten a in β C_a eine Variante eines Faktors einer Klausel in $S \cup \{E\}$ ist. Wir transformieren nun β in einen unifizierbaren Klauselbaum für S_F, indem wir jeden Knoten a, für den C_a keine Variante einer Klausel in S_F ist, durch zwei neue Knoten ersetzen, die mit D und einem Faktor von C markiert sind.

Sei also a ein Knoten in β, für den C_a keine Variante einer Klausel in S_F ist. C_a muß eine Variante eines Faktors von E sein. O.B.d.A. sei C_a ein Faktor von E mit $Var(\beta - a) \cap Var(C \cup D) = \emptyset$. Sei ξ ein Unifikator von β mit $Ber(\xi) \subset Var(\beta)$. Lemma 4.2.14 garantiert nun die Existenz von C', L' und θ mit:

(1) C' ist ein Faktor von C und $L' \in C'$.

(2) θ ist ein normaler Unifikator von L' und $\bar{K}$ mit $Var(\theta) \subset Var(C \cup D)$.

(3) $|C'| = |\theta C'|$ und $\theta(C' - L') \subset C_a$.

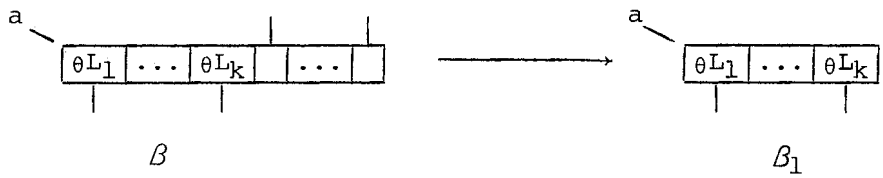
Also existieren Literale $L_1,\ldots,L_k$, $k \geq 0$ mit:

(4) $C' = \{L_1,\ldots,L_k,L'\}$ ist eine Variante einer Klausel in S_F und $|C'| = k+1$.

(5) θ ist ein normaler Unifikator für L' und $\bar{K}$ mit $Var(\theta) \cap Var(\beta - a) = \emptyset$.

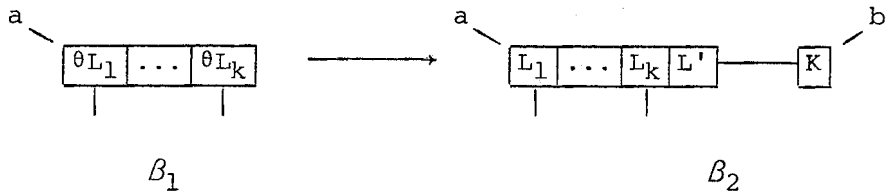
(6) $\{\theta L_1,\ldots,\theta L_k\} \subset C_a$ und $|\{\theta L_1,\ldots,\theta L_k\}| = k$.

Wir überführen β zunächst in den unifizierbaren Klauselbaum β_1, indem wir C_a auf $\{\theta L_1,\ldots,\theta L_k\}$ schrumpfen:



Sei $\xi_1 := \xi\theta$. Da θ idempotent ist, $Var(\beta_1 - a) \cap Var(\theta) = \emptyset$ gilt und ξ ein Unifikator von β_1 ist, ist auch ξ_1 ein Unifikator von β_1.

Wir überführen nun β_1 in einen Klauselbaum β_2, indem wir einen neuen Knoten b wie folgt einfügen:



Offensichtlich ist ξ_1 ein Unifikator für β_2, da $\xi_1 = \xi_1\theta$ gilt und wegen (5) $\xi_1 L' = \xi_1 \bar{K}$ ist.

Durch Wiederholung der obigen Transformation können alle Knoten a in β beseitigt werden, für die C_a keine Variante einer Klausel in S_F ist. Damit erhalten wir, wie zu zeigen war, einen unifizierbaren Klauselbaum für S_F. []

4.2.16 <u>Beweis für Satz 4.2.7.</u>

"(a) ==> (e)". Lemma 4.2.15.

"(e) ==> (b)". Satz 4.2.10.

"(b) ==> (a)". Trivial.

"(c) ==> (a)". Siehe /CL73, S.135, Theorem 7.1/.

"(e) ==> (d)". Satz 4.2.10. "(d) ==> (c)". Trivial. []

4.3 Projektionen

Wenn $\mathcal{G}_S$ der initiale Graph zu einer unären Klauselmenge S ist, die ihre Faktoren enthält, dann muß $\mathcal{G}_S$ einen R-Baum für S in überlagerter Form als Subgraphen enthalten. Dieser Kern geht, wenn man von $\mathcal{G}_S$ aus Ableitungsschritte ausführt, in jeweils neue Kerne über. In diesem Abschnitt werden wir zeigen, wie der Kern sich bei den verschiedenen Ableitungsschritten transformiert, und unter welchen Bedingungen er reduziert wird. Wenn man die Kerntransformationen im Griff hat, dann kann man auch die Bedingungen angeben, unter denen Filter konfluent und noethersch sind.

Die Kerntransformationen sind jedoch, wenn man sie direkt betrachtet, sehr verwickelt. Wir falten daher den Kern in den korrespondierenden R-Baum auf und verbinden die aufgefaltete und die überlagerte Version des Kerns durch eine Projektion. Betrachtet man nun, statt der Kerntransformationen, die Transformationen des korrespondierenden R-Baumes und der zugehörigen Projektion, dann lösen sich alle Schwierigkeiten in Wohlgefallen auf.

4.3.1 Projektionen. Seien $\mathcal{G} = \langle \text{Knoten}, C, \text{Links} \rangle$ und $\mathcal{G}' = \langle \text{Knoten}', C', \text{Links}' \rangle$ Klauselgraphen. Ein Paar $\langle \hat{\ }, \pi \rangle$ heißt Projektion von $\mathcal{G}$ in $\mathcal{G}'$, wenn gilt:

(a) $\hat{\ }$ ist eine Abbildung Knoten $\longrightarrow$ Knoten'. Für $\hat{\ }(a)$ schreiben wir kurz $\hat{a}$.

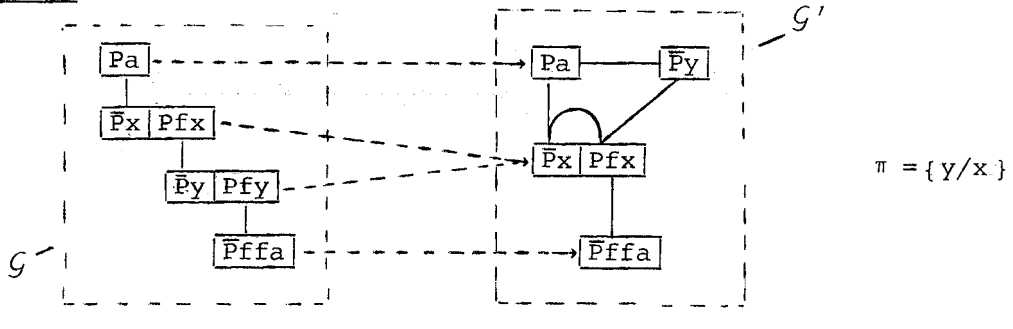
(b) Für jeden Knoten a in $\mathcal{G}$ ist $C'_{\hat{a}}$ eine Variante von C_a.

(c) π ist eine Substitution mit $\text{Ber}(\pi) \subset \text{Var}(\mathcal{G})$, so daß für jeden Knoten a in $\mathcal{G}$ gilt: $\pi C_a = C'_{\hat{a}}$.

(d) Für jeden Link ℓ in $\mathcal{G}$ ist $\hat{\ell}$ ein Link in $\mathcal{G}'$. Für einen Link $\ell = L^a K^b$ in $\mathcal{G}$ ist $\hat{\ell} := \{(\pi L)^{\hat{a}}, (\pi K)^{\hat{b}}\} := \pi L^{\hat{a}} \pi K^{\hat{b}}$.

Wir schreiben kurz $\langle \hat{\ }, \pi \rangle : \mathcal{G} \longrightarrow \mathcal{G}'$, wenn $\mathcal{G}$ und $\mathcal{G}'$ Klauselgraphen sind und $\langle \hat{\ }, \pi \rangle$ eine Projektion von $\mathcal{G}$ in $\mathcal{G}'$ ist.

4.3.2 Beispiel.



4.3.3 <u>Bild($\mathcal{G}$,$\mathcal{G}'$,<^,π>)</u>. Sei <^,π>: $\mathcal{G} \longrightarrow \mathcal{G}'$. Das Bild Bild($\mathcal{G}$,$\mathcal{G}'$,<^,π>) von $\mathcal{G}$ in $\mathcal{G}'$ bzgl. der Projektion <^,π> ist der Subgraph von $\mathcal{G}'$ mit den Knoten $\{\hat{a} \mid a$ Knoten in $\mathcal{G}\}$ und den Links $\{\hat{\ell} \mid \ell$ Link in $\mathcal{G}\}$. Wenn $\mathcal{G}'$ und <^,π> aus dem Kontext klar sind, schreiben wir für Bild($\mathcal{G}$,$\mathcal{G}'$,<^,π>) kurz Bild($\mathcal{G}$).
In Beispiel 4.3.2 ist Bild($\mathcal{G}$) der folgende Klauselgraph:

4.3.4 <u>Lemma</u>. Sei <^,π>: $\mathcal{G} \longrightarrow \mathcal{G}'$. Dann gilt:
(a) Wenn $\mathcal{G}$ unerfüllbar ist, dann ist auch $\mathcal{G}'$ unerfüllbar.
(b) Wenn L^a und K^b potentiell komplementäre Literalauftreten in $\mathcal{G}$ sind, dann sind πL und πK potentiell komplementär.
(c) Wenn $\square \in S(\mathcal{G})$, dann ist auch $\square \in S(\mathcal{G}')$.

Beweis. Sei <^,π>: $\mathcal{G} \longrightarrow \mathcal{G}'$.
Teil (a). Sei $\mathcal{G}$ unerfüllbar. Dann ist $S(\mathcal{G})$ unerfüllbar. Da jede Klausel in $S(\mathcal{G})$ eine Variante einer Klausel in $S(\mathcal{G}')$ ist, kann $S(\mathcal{G}')$ nicht erfüllbar sein.
Teil (b). Seien L^a und K^b potentiell komplementäre Literalauftreten in $\mathcal{G}$. Dann sind L und K potentiell komplementär. Da πL und πK durch Variablenumbenennung aus L und K entstehen, müssen auch πL und πK potentiell komplementär sein.
Teil (c). Trivial. []

4.3.5 <u>Satz</u>. Sei <^,π>: $\mathcal{B} \longrightarrow \mathcal{G}$ und sei $\mathcal{B}$ ein R-Baum. Dann gilt:
(a) Bild($\mathcal{B}$) ist ein unerfüllbarer, π-reduzierter Subgraph von $\mathcal{G}$.
(b) Bild($\mathcal{B}$) enthält keine Tautologien und keine τ-Links.
(c) Entweder ist Bild($\mathcal{B}$) = ($\square$) oder Bild($\mathcal{B}$) enthält einen unären Non-τ-Link.
(d) Bild($\mathcal{B}$) $\subset$ π($\mathcal{G}$).

Beweis. Sei <^,π>: $\mathcal{B} \longrightarrow \mathcal{G}$ und $\mathcal{B}$ ein R-Baum.
Teil (a1): Bild($\mathcal{B}$) ist unerfüllbar. <^,π> ist insbesondere eine Projektion von $\mathcal{B}$ in Bild($\mathcal{B}$). Da $\mathcal{B}$ als R-Baum unerfüllbar ist, ist nach Lemma 4.3.4(a) Bild($\mathcal{B}$) ebenfalls unerfüllbar.
Teil (a2): Bild($\mathcal{B}$) ist π-reduziert. Wir müssen zeigen, daß jedes Literalauftreten in Bild($\mathcal{B}$) mit einem Link in Bild($\mathcal{B}$) inzidiert, der kein Autolink ist.
Sei $\pi L^{\hat{a}}$ ein Literalauftreten in Bild($\mathcal{B}$). L^a ist dann ein Literalauftreten in $\mathcal{B}$. Sei $\ell = L^a K^b$ der eindeutig bestimmte Link an L^a in $\mathcal{B}$. Dann ist $\hat{\ell}$ ein Link in

Bild($\mathcal{B}$) an $\pi L^{\hat{a}}$.

Fall a21: $\hat{\ell}$ ist kein Autolink. Dann ist $\pi L^{\hat{a}}$, wie zu zeigen war, in Bild($\mathcal{B}$) nicht isoliert.

Fall a22: $\hat{\ell}$ ist ein Autolink. Dann ist $\hat{a} = \hat{b}$. Es existiert also ein Literal $L_1 \in C_b$ mit $\pi L_1^{\hat{b}} = \pi L^{\hat{a}}$ und $L_1 \neq K$. Wie vorher für L^a gibt es auch an L_1^b einen eindeutig bestimmten Link ℓ_1 in $\mathcal{B}$. Falls $\hat{\ell}_1$ wieder ein Autolink ist, weiche man entsprechend auf ℓ_2 aus, und so fort. Da die Links $\ell_1, \ell_2, \ell_3, \ldots$ einen Pfad in $\mathcal{B}$ bilden und $\mathcal{B}$ als Baum nur endlich lange Pfade erlaubt, erreicht man schließlich einen Link ℓ_n, für den $\hat{\ell}_n$ kein Autolink ist und mit $\pi L^{\hat{a}}$ inzidiert.

Teil (b1): Bild($\mathcal{B}$) enthält keine Tautologien. Wegen Proposition 4.1.12 enthält $\mathcal{B}$ als R-Baum keine Tautologie. Da jede Klausel in Bild($\mathcal{B}$) eine Variante einer Klausel in $\mathcal{B}$ ist, kann Bild($\mathcal{B}$) ebenfalls keine Tautologie enthalten.

Teil (b2): Bild($\mathcal{B}$) enthält keine τ-Links. Wegen Korrolar 4.1.14(a) enthält $\mathcal{B}$ als R-Baum keine τ-Links. Sei $\ell = L^a K^b$ ein Link in $\mathcal{B}$. Dann ist Res(C_a,L;C_b,K) also keine Tautologie. Res(πC_a,πL;πC_b,πK) ist eine Variante von Res(C_a,L;C_b,K), da $\pi|_{C_a}$ bzw. $\pi|_{C_b}$ eine U-Substitution für C_a bzw. C_b ist, und da πC_a und πC_b variablendisjunkt sind. Also ist auch $\hat{\ell}$ kein τ-Link.

Teil (c). $\mathcal{B}$ enthält einen unären Link oder $\mathcal{B} = (\square)$. Wenn $\mathcal{B} = (\square)$ ist, dann ist auch Bild($\mathcal{B}$) = ($\square$). Wenn ℓ ein unärer Link in $\mathcal{B}$ ist, dann ist $\hat{\ell}$ ein unärer Link in Bild($\mathcal{B}$).

Teil (d). Da $\pi(\mathcal{G})$ der größte π-reduzierte Subgraph von $\mathcal{G}$ ist, und Bild($\mathcal{B}$) ein π-reduzierter Subgraph von $\mathcal{G}$ ist, gilt Bild($\mathcal{B}$) $\subset \pi(\mathcal{G})$. []

4.3.6 <u>Satz</u>. Sei S eine UWF-Klauselmenge. Dann gibt es einen R-Baum $\mathcal{B}$ und eine Projektion von $\mathcal{B}$ in $\mathcal{G}_S$.

Beweis. Sei S eine UWF-Klauselmenge und sei $\mathcal{B}$ ein R-Baum zu S (existiert wegen Satz 4.2.7). Wenn man den Algorithmus 2.1.13 für die Bildung des initialen Klauselgraphen auf S anwendet, entsteht, direkt vor Ausführung von Schritt 2, ein totaler Klauselgraph $\mathcal{G}$ mit:

(1) $\mathcal{G}_S = \pi(\mathcal{G})$.

(2) Zu jeder Klausel $C \in S$, die keine Tautologie ist, existiert ein Knoten a in $\mathcal{G}$, so daß C_a eine Variante von C ist.

Wegen Satz 4.3.5(d) genügt es zu zeigen, daß eine Projektion $\langle\hat{}, \pi\rangle$ von $\mathcal{B}$ in $\mathcal{G}$ existiert. Für $\hat{}$ wählen wir eine Abbildung von den Knoten in $\mathcal{B}$ in die Knoten von $\mathcal{G}$, so daß für jeden Knoten a in $\mathcal{B}$ die Klausel $C_{\hat{a}}$ in $\mathcal{G}$ eine Variante von

C_a ist. Die Existenz einer solchen Abbildung ist durch (2) und die Proposition 4.1.12 gesichert. Für jeden Knoten a in $\mathcal{B}$ wählen wir nun eine Substitution π_a mit $Ber(\pi_a) \subset Var(C_a)$ und $\pi_a C_a = C_{\hat{a}}$. Wir definieren $\pi := \pi_{a_1} \cup \ldots \cup \pi_{a_n}$, wobei $a_1, \ldots, a_n$ die Knoten von $\mathcal{B}$ sind. Da $Var(C_{a_i}) \cap Var(C_{a_j}) = \emptyset$ für $i \neq j$ gilt, ist π eine Substitution. Da $\mathcal{G}$ total ist, folgt mit Lemma 4.3.4(b), daß $<\hat{}, \pi>$ eine Projektion von $\mathcal{B}$ in $\mathcal{G}$ ist. []

Wir wissen also jetzt, wie ein R-Baum $\mathcal{B}$ zu einer UWF-Klauselmenge S mittels einer Projektion <> überlagert werden muß, damit er als Subgraph Bild($\mathcal{B}$) in $\mathcal{G}_S$ enthalten ist. Als nächstes ist zu klären, wie sich $\mathcal{B}$ und <> transformieren, wenn man eine der vier Ableitungsregeln des KOWALSKI-Kalküls anwendet. Wir betrachten zunächst die Resolutionsregel.

4.3.7 <u>Lemma</u>. Sei $<\hat{}, \pi>: \mathcal{B} \longrightarrow \mathcal{G}'$ und sei $\mathcal{B}$ ein unifizierbarer Klauselgraph. Sei ℓ ein Link in $\mathcal{B}$ und sei $\mathcal{G}$ ein Klauselgraph mit $\mathcal{G} \xrightarrow[\rho]{\hat{\ell}} \mathcal{G}'$. Dann gibt es $\mathcal{B}'$ und $<\tilde{}, \psi>$ mit:

$$\begin{tikzcd}
\mathcal{G} \arrow[r, "\hat{\ell}", "\rho_o"'] & \mathcal{G}' \\
\mathcal{B} \arrow[u, "<\hat{}, \pi>"] \arrow[r, "\ell", "\rho"'] & \mathcal{B}' \arrow[ul, "<\tilde{}, \psi>"']
\end{tikzcd}$$

Beweis. Seien $\mathcal{G}$, $\mathcal{G}'$, $\mathcal{B}$, ℓ und $<\hat{}, \pi>$ wie oben gewählt. Es genügt offensichtlich zu zeigen, daß $\mathcal{B}'$ und $<\tilde{}, \psi>$ existieren mit:

$$\begin{tikzcd}
\mathcal{G} \arrow[r, "\hat{\ell}", "\rho_o"'] & \mathcal{G}' \\
\mathcal{B} \arrow[u, "<\hat{}, \pi>"] \arrow[r, "\ell", "\rho_o"'] & \mathcal{B}' \arrow[ul, "<\tilde{}, \psi>"']
\end{tikzcd}$$

Sei $\ell = L_o^a K_o^b$. $\mathcal{B}'$ sei ein Graph, der durch einen Übergang $\mathcal{B} \xrightarrow[\rho_o]{\ell} \mathcal{B}'$ entsteht, bei dem nur solche K^d, $d \in \{a,b\}$, $K \in C_d$ als signifikante Vorgänger gewählt werden, für die $K^{\hat{d}}$ ein signifikanter Vorgänger bei dem Übergang $\mathcal{G} \xrightarrow[\rho_o]{\hat{\ell}} \mathcal{G}'$ war. $\mathcal{B}'$ ist bis auf die Knoten- und Variablenbenennungen eindeutig bestimmt. Sei c der Resolventenknoten zu ℓ in $\mathcal{B}'$. Wir konstruieren eine Projektion $<\tilde{}, \psi>$: $\mathcal{B}' \longrightarrow \mathcal{G}'$ wie folgt:

$\tilde{}$ sei die Erweiterung von $\hat{}$, bei der $\tilde{c}$ der Resolventenknoten zu $\hat{\ell}$ in $\mathcal{G}'$ ist.

$\psi := \pi \cup \sigma$, wobei σ eine U-Substitution für C_c ist mit:

(1) $\sigma C_c = C_{\tilde{c}}$.

(2) Für alle $L \in C_c$ gilt: Wenn K^d der signifikante Vorgänger zu L^c ist, dann ist $\pi K^{\hat{d}}$ der signifikante Vorgänger zu σL^c.

Da Ber(σ) und Ber(π) disjunkt sind, ist ψ jedenfalls eine Substitution. Um zu zeigen, daß $<\tilde{},\psi>$ eine Projektion von $\mathcal{B}'$ in $\mathcal{G}'$ ist, genügt es, da die anderen Bedingungen offensichtlich erfüllt sind, zu zeigen:

(3) Für jeden Link ℓ_1 in $\mathcal{B}'$, der mit dem Resolventenknoten c inzidiert, ist $\tilde{\ell}_1$ ein Link in $\mathcal{G}'$.

Sei also $\ell_1 = M^e L^c$ ein Link in $\mathcal{B}'$. Sei K^d der signifikante Vorgänger zu L^c. Lemma 2.3.6 liefert, daß $M^e K^d$ ein Link in $\mathcal{B}'$ ist. Da $e \neq c$ ist (siehe Beweis von Satz 4.1.13), ist $M^e K^d$ auch ein Link in $\mathcal{B}$. Also ist $\pi M^{\hat{e}} \pi K^{\hat{d}}$ ein Link in $\mathcal{G}'$. Weil M und L potentiell komplementär sind, sind auch πM und πL potentiell komplementär. Da wegen (2) $\pi K^{\hat{d}}$ der signifikante Vorgänger zu $\psi L^{\tilde{c}}$ ist, liefert nun die andere Richtung von Lemma 2.3.6, daß $\pi M^{\hat{e}} \psi L^{\tilde{c}}$ ein Link in $\mathcal{G}'$ ist. Also ist, wie zu zeigen war, $\tilde{\ell}_1 = \psi M^{\tilde{e}} \psi L^{\tilde{c}} = \pi M^{\hat{e}} \psi L^{\tilde{c}}$ Link in $\mathcal{G}'$. []

4.3.8 <u>Satz</u>. Sei $\mathcal{G} \xrightarrow[\rho]{\ell} \mathcal{G}'$ und $<\hat{},\pi>: \mathcal{B} \longrightarrow \mathcal{G}$, wobei $\mathcal{B}$ ein unifizierbarer Klauselbaum ist. Dann existieren $\mathcal{B}'$ und $<\tilde{},\psi>$ mit:

$$\begin{tikzcd}
\mathcal{G} \arrow[r, "\ell", "\rho"'] & \mathcal{G}' \\
\mathcal{B} \arrow[u, "<\hat{},\pi>"] \arrow[r, "*", "\rho"'] & \mathcal{B}' \arrow[u, "<\tilde{},\psi>"']
\end{tikzcd}$$

Wenn ℓ ein Link in Bild($\mathcal{B}$) ist, dann gilt $\mathcal{B} \xrightarrow[\rho]{+} \mathcal{B}'$. Wenn ℓ kein Link in Bild($\mathcal{B}$) ist, dann gilt $\mathcal{B}' = \mathcal{B}$ und $<\tilde{},\psi> = <\hat{},\pi>$.

Beweis. Sei also $\mathcal{G} \xrightarrow[\rho]{\ell} \mathcal{G}'$ und $<\hat{},\pi>: \mathcal{B} \longrightarrow \mathcal{G}$, so daß $\mathcal{B}$ ein unifizierbarer Klauselbaum ist. Dann existiert ein Klauselgraph $\mathcal{G}''$ mit:

(1) $\mathcal{G} \xrightarrow[\rho_o]{\ell} \mathcal{G}''$ und $\mathcal{G}' = \pi(\mathcal{G}'' - \ell)$.

(2) $<\hat{},\pi>: \mathcal{B} \longrightarrow \mathcal{G}''$ (da $\mathcal{G} \subset \mathcal{G}''$).

Fall 1: ℓ ist kein Link in Bild($\mathcal{B}$). Dann gilt wegen Bild($\mathcal{B}$) $\subset \mathcal{G}'' - \ell$, Lemma 4.3.5(d) und (1), daß $<\hat{},\pi>$ eine Projektion von $\mathcal{B}$ in $\mathcal{G}'$ ist. Das war zu zeigen.

Fall 2: ℓ ist ein Link in Bild($\mathcal{B}$). Dann gibt es einen Link ℓ_1 in $\mathcal{B}$, so daß $\hat{\ell}_1 = \ell$ ist. Durch eventuell mehrfaches Anwenden von Lemma 4.3.7 erhält man $\mathcal{B}'$ und $<\tilde{\ }, \psi>$ mit:

$$\begin{tikzcd}
\mathcal{G} \arrow[r, "\ell", "\rho_0"'] & \mathcal{G}'' \\
\mathcal{B} \arrow[u, "<\hat{\ }, \pi>"] \arrow[r, "+", "\rho"'] & \mathcal{B}' \arrow[ul, "<\tilde{\ }, \psi>"']
\end{tikzcd}$$

so daß ℓ kein Link in Bild($\mathcal{B}'$) ist. Ein solcher Baum $\mathcal{B}'$ muß nach endlich vielen ρ-Schritten erreicht sein, weil jeder ρ-Schritt die Knotenanzahl echt reduziert. Also ergibt sich mit derselben Argumentation wie bei Fall 1, daß $<\tilde{\ }, \psi>$ eine Projektion von $\mathcal{B}'$ in $\mathcal{G}'$ ist. []

Bei der Faktorisierungs- und bei der Tautologieeliminations-Regel sind die Verhältnisse sehr einfach, da sie Bild($\mathcal{B}$) nicht berühren.

4.3.9 <u>Satz</u>. Sei $\mathcal{G} \xrightarrow[\phi]{} \mathcal{G}'$ und $<\hat{\ }, \pi>: \mathcal{B} \longrightarrow \mathcal{G}$. Dann gilt:

$$\begin{tikzcd}
\mathcal{G} \arrow[r, "\phi"'] & \mathcal{G}' \\
\mathcal{B} \arrow[u, "<\hat{\ }, \pi>"] \arrow[r, equal] & \mathcal{B} \arrow[u, "<\hat{\ }, \pi>"']
\end{tikzcd}$$

Beweis. Trivial, da $\mathcal{G} \subset \mathcal{G}'$. []

4.3.10 <u>Satz</u>. Sei $\mathcal{G} \xrightarrow[\tau]{} \mathcal{G}'$, $<\hat{\ }, \pi>: \mathcal{B} \longrightarrow \mathcal{G}$ und sei $\mathcal{B}$ ein R-Baum. Dann gilt:

$$\begin{tikzcd}
\mathcal{G} \arrow[r, "\tau"'] & \mathcal{G} \\
\mathcal{B} \arrow[u, "<\hat{\ }, \pi>"] \arrow[r, equal] & \mathcal{B} \arrow[u, "<\hat{\ }, \pi>"']
\end{tikzcd}$$

Beweis. Sei $\mathcal{G} \xrightarrow[\tau]{} \mathcal{G}'$ und $<\hat{\ }, \pi>: \mathcal{B} \longrightarrow \mathcal{G}$. Dann gibt es einen Knoten a in $\mathcal{G}$, so

daß C_a eine Tautologie und $\mathcal{G}' = \pi(\mathcal{G} - a)$ ist. Da Bild$(\mathcal{B},\mathcal{G},\langle\hat{}\,,\pi\rangle)$ keine Tautologie enthält (Satz 4.3.5(b)), gilt Bild$(\mathcal{B},\mathcal{G},\langle\hat{}\,,\pi\rangle) \subset \mathcal{G} - a$. Mit Satz 4.3.5(d) folgt nun Bild$(\mathcal{B},\mathcal{G},\langle\hat{}\,,\pi\rangle) \subset \pi(\mathcal{G} - a) = \mathcal{G}'$, wie zu zeigen war. []

Abschließend betrachten wir die Transformation des korrespondierenden R-Baums und seiner Projektion bei der Anwendung der Subsumtions-Eliminationsregel.

4.3.11 <u>Satz</u>. Sei $\mathcal{G} \xrightarrow[\sigma]{a} \mathcal{G}'$ und sei $\langle\hat{}\,,\pi\rangle: \mathcal{B} \longrightarrow \mathcal{G}$, wobei a die BSEB in $\mathcal{G}$ erfüllt und $\mathcal{B}$ ein unifizierbarer Klauselbaum ist. Dann existieren $\mathcal{B}'$ und $\langle\tilde{}\,,\psi\rangle$ mit:

$$\begin{tikzcd}
\mathcal{G} \arrow[r, "a", "\sigma"'] & \mathcal{G}' \\
\mathcal{B} \arrow[u, "\langle\hat{}\,,\pi\rangle"] \arrow[r, "*", "\sigma_B"'] & \mathcal{B}' \arrow[u, "\langle\tilde{}\,,\psi\rangle"']
\end{tikzcd}$$

Wenn a ein Knoten in Bild$(\mathcal{B})$ ist, dann gilt $\mathcal{B} \xrightarrow[\sigma_B]{+} \mathcal{B}'$, wobei, wenn $\mathcal{G} \xrightarrow[\sigma]{a} \mathcal{G}'$ reduzierend (siehe 2.8.9(c)) ist, auch die Schritte $\mathcal{B} \xrightarrow[\sigma_B]{+} \mathcal{B}'$ reduzierend (siehe 4.1.17) sind. Wenn a kein Knoten in Bild$(\mathcal{B})$ ist, dann gilt $\mathcal{B}' = \mathcal{B}$ und $\langle\tilde{}\,,\psi\rangle = \langle\hat{}\,,\pi\rangle$.

Beweis. Da $\mathcal{G}' = \pi(\mathcal{G} - a)$ ist, folgt die Aussage des Satzes, wenn a nicht in Bild$(\mathcal{G})$ liegt, aus Satz 4.3.5(d). Wir betrachten nun den interessanten Fall. Seien also $\mathcal{B}$, a, $\mathcal{G}$, $\langle\hat{}\,,\pi\rangle$, b, θ und $\mathcal{G}'$ gegeben mit

$$\begin{tikzcd}
\mathcal{G} \arrow[r, "\hat{a}", "\sigma"'] & \mathcal{G}' \\
\mathcal{B} \arrow[u, "\langle\hat{}\,,\pi\rangle"] &
\end{tikzcd}$$

wobei a ein Knoten in $\mathcal{B}$ ist und $\hat{a}$ die BSEB bzgl. b und θ in $\mathcal{G}$ erfüllt (Notabene: b ist Knoten in $\mathcal{G}$). Es genügt zu zeigen, daß es $\mathcal{B}'$ und $\langle\tilde{}\,,\psi\rangle$ gibt mit:

(1)

$$\begin{tikzcd} \mathcal{G} \arrow[r, "\hat{a}", "\sigma"'] & \mathcal{G}' \\ \mathcal{B} \arrow[u, "\langle\hat{\ },\pi\rangle"] \arrow[r, "\sigma_B"'] & \mathcal{B}' \arrow[ul, "\langle\tilde{\ },\psi\rangle"'] \end{tikzcd}$$

(2) Die Anzahl der Knoten in $\mathcal{B}'$, die auf $\hat{a}$ projiziert werden, ist echt kleiner als die Anzahl der entsprechenden Knoten in $\mathcal{B}$.

(3) Wenn der σ-Schritt reduzierend ist, dann ist auch der σ_B-Schritt reduzierend.

Wir konstruieren nun $\mathcal{B}'$ und $\langle\tilde{\ },\psi\rangle$ mit diesen Eigenschaften wie folgt:

- D sei eine Variante von C_b, die variablendisjunkt zu $\mathcal{B}$ ist.
- π_D sei eine U-Substitution für D mit $\pi_D D = C_b$.
- $\pi_a := \pi|_{Var(C_a)}$; π_a ist offensichtlich eine U-Substitution für C_a mit $\pi_a C_a = C_{\hat{a}}$.
- $\tau := \pi_a^{-1}\theta\pi_D$; τ ist offensichtlich eine S-Substitution von D nach C_a.
- $\mathcal{B}'$ sei ein Baum mit $\mathcal{B} \xrightarrow[\sigma_B]{a,D,\tau} \mathcal{B}'$; $\mathcal{B}'$ enthält den Knoten a nicht mehr.
- d sei der gegenüber $\mathcal{B}$ neue Knoten in $\mathcal{B}'$.
- $\tilde{\ }$ sei die Abbildung von den Knoten von $\mathcal{B}'$ in die Knoten von $\mathcal{G}$ mit

$$\tilde{c} := \begin{cases} \hat{c} & \text{wenn c ein Knoten in } \mathcal{B} \text{ ist} \\ b & \text{wenn } c = d \text{ ist.} \end{cases}$$

- $\psi := \pi|_{Var(\mathcal{B}')} \cup \pi_D$. Offensichtlich ist ψ eine Substitution mit: Für jeden Knoten c in $\mathcal{B}'$ gilt: $\psi C_c = C_{\tilde{c}}$.

Wir zeigen nun, daß $\langle\tilde{\ },\psi\rangle$ eine Projektion von $\mathcal{B}'$ in $\mathcal{G}$ ist. Da die anderen Bedingungen evident sind, bleibt nur zu zeigen, daß für jeden Link ℓ in $\mathcal{B}'$, der mit d inzidiert, $\tilde{\ell}$ ein Link in $\mathcal{G}$ ist.

Sei also $\ell = M^e L^d$ ein Link in $\mathcal{B}'$. Dann ist:

(4) $\tilde{\ell} = \psi M^{\tilde{e}} \psi L^{\tilde{d}} = \pi M^{\hat{e}} \pi_D L^b$.

Andererseits muß $M^e \tau L^a$ ein Link in $\mathcal{B}$ sein, für den gilt:

(5) $\widehat{M^e \tau L^a} = \pi M^{\hat{e}} \pi\tau L^{\hat{a}} = \pi M^{\hat{e}} \pi_a(\pi_a^{-1}\theta\pi_D) L^{\hat{a}} = \pi M^{\hat{e}} \theta\pi_D L^{\hat{a}}$.

Da $\langle\hat{\ },\pi\rangle$ eine Projektion von $\mathcal{B}$ in $\mathcal{G}$ ist, ist $\pi M^{\hat{e}} \theta\pi_D L^{\hat{a}}$ ein Link in $\mathcal{G}$. Da $\pi_D L$ ein Literal von C_b ist, garantiert die BSEB, daß auch $\pi M^{\hat{e}} \pi_D L^b$ ein Link in $\mathcal{G}$ ist. Also ist wegen (4), wie zu zeigen war, $\tilde{\ell}$ ein Link in $\mathcal{G}$. Man überzeugt sich leicht, daß auch die Bedingungen (2) und (3) erfüllt sind. [ ]

4.3.12 <u>Bemerkung</u>. Der Beweis zu Satz 4.3.11 macht klar, warum in der

BSEB (2.8.7) gefordert wird, daß a von b nicht nur *klauselmäßig* (d.h. $\theta C_b \subset C_a$), sondern auch *linkmäßig* (d.h., wenn $K^c \theta L^a$ ein Link in $\mathcal{G}$ ist, dann muß auch $K^c L^b$ ein Link in $\mathcal{G}$ sein) subsumiert werden muß. Außerdem wird ersichtlich, warum wir für eine S-Substitution (siehe 2.8.1) θ von D nach C verlangen, daß $|\theta D| = |D|$ ist. Würden wir darauf verzichten und statt dessen, wie LOVELAND in /Lo78/, nur verlangen, daß $|D| \leq |C|$ ist, dann subsumiert z.B. die Klausel PxPyPz die Klausel PaQR. Das würde beim Übergang $\beta \xrightarrow[\sigma_B]{} \beta'$ erforderlich machen, daß man den Subbaum für Pa zweimal kopiert. Damit würde β' echt mehr Knoten als β haben. Zudem könnten durch das Kopieren neue Auftreten von PaQR entstehen. Das bedeutet, daß man selbst die Terminierung des Ersetzungsprozesses $\beta \xrightarrow[\sigma_B]{+} \beta'$ nicht mehr garantieren kann.

4.4 Ein Kriterium für vollständige Filter

Der Leser erreicht nun den Höhepunkt der Arbeit. Wir geben in diesem Abschnitt allgemeine, leicht zu verifizierende Bedingungen an, die für die Konfluenz und die NOETHER-Eigenschaft von Filtern bei unären Klauselmengen hinreichend sind. Wie sich zeigen wird, lassen sich diese Kriterien mit den in den vorangegangenen Abschnitten bereitgestellten Werkzeugen kurz und elegant beweisen.

4.4.1 Satz (Konfluenz). Sei S eine UWF-Klauselmenge und sei $\mathcal{G}$ ein Klauselgraph mit $\mathcal{G}_S \vdash_{BSE} \mathcal{G}$. Dann gibt es eine Widerlegung $\mathcal{G} \xrightarrow[\rho]{*} \square$, bei der nur auf unären Non-τ-Links resolviert wird.

Beweis. Sei S eine UWF-Klauselmenge und sei $\mathcal{G}$ ein Graph mit $\mathcal{G}_S \vdash_{BSE} \mathcal{G}$. Sei $\mathcal{B}$ ein R-Baum und sei $<>$ eine Projektion von $\mathcal{B}$ in $\mathcal{G}$. Die Existenz von $\mathcal{B}$ und $<>$ ist durch die Sätze 4.3.6, 4.3.8, 4.3.9, 4.3.10 und 4.3.11 gesichert. Es genügt nun zu zeigen, daß eine der folgenden zwei Bedingungen erfüllt ist:

(1) $\square \in S(\mathcal{G})$.

(2) $\mathcal{G}$ enthält einen unären Non-τ-Link ℓ und es existieren $\mathcal{G}'$, $\mathcal{B}'$ und $<>'$ mit:

$$\begin{tikzcd}
\mathcal{G} \arrow[r, "\ell", "\rho"'] & \mathcal{G}' \\
\mathcal{B} \arrow[u, "<>"] \arrow[r, "+", "\rho"'] & \mathcal{B}' \arrow[u, "<>'"']
\end{tikzcd}$$

und $|\mathcal{B}| > |\mathcal{B}'| > 0$.

Sei also Bedingung (1) nicht erfüllt. Dann folgt aus Satz 4.3.5(c), daß $\mathrm{Bild}(\mathcal{B}) \subset \mathcal{G}$ einen unären Non-τ-Link ℓ enthält. Satz 4.3.8 liefert nun, daß auch $\mathcal{B}'$ und $<>'$ wie in (2) gefordert existieren. Also ist Bedingung (2) erfüllt. []

Im Gegensatz zu den aussagenlogischen Klauselmengen (Beispiel 2.7.2) kann also bei den UWF-Klauselmengen die Widerlegbarkeit, auch bei uneingeschränkter Anwendung der τ-Regel, nicht verloren gehen. Beispiel 2.7.8 zeigt jedoch, daß die Anwesenheit der initialen Faktoren dafür aber unbedingt erforderlich ist. Der dort dargestellte Graph ist der initiale Graph zu einer UW-Klauselmenge. Er kann durch je zwei ρ- und τ-Schritte auf den leeren (erfüllbar!) Graphen

abgeleitet werden.

4.4.2 <u>Unär erschöpfende nichtabbrechende Ableitungen</u>. Eine nichtabbrechende Ableitung heißt unär erschöpfend, wenn jeder unäre Non-τ-Link nach endlich vielen Schritten wieder gelöscht wird.

4.4.3 <u>Satz</u> (NOETHER-Eigenschaft). Sei S eine UWF-Klauselmenge. Dann gibt es keine unär erschöpfende, nichtabbrechende BSERF-Ableitung von $\mathcal{G}_S$ aus.

Beweis. Wir schreiben hier $\mathcal{B} \xrightarrow[\beta]{} \mathcal{B}'$, wenn $\mathcal{B} \xrightarrow[\rho]{} \mathcal{B}'$ oder $\mathcal{B} \xrightarrow[\sigma_B]{} \mathcal{B}'$ gilt. Sei also S eine UWF-Klauselmenge.

Angenommen, $\Gamma: \mathcal{G}_S \longrightarrow \mathcal{G}_1 \longrightarrow \mathcal{G}_2 \longrightarrow \ldots$ ist eine unär erschöpfende, nichtabbrechende BSERF-Ableitung. Seien $\mathcal{B}_0, \mathcal{B}_1, \ldots$ R-Bäume und $<>_0, <>_1, \ldots$ Projektion wie folgt:

$$\begin{tikzcd}
\mathcal{G}_S \arrow[r] & \mathcal{G}_1 \arrow[r] & \mathcal{G}_2 \arrow[r] & \cdots\cdots \arrow[r] & \mathcal{G}_i \arrow[r] & \cdots\cdots \\
\mathcal{B}_0 \arrow[u, "<>_0"] \arrow[r, "\beta"', "*"] & \mathcal{B}_1 \arrow[u, "<>_1"] \arrow[r, "\beta"', "*"] & \mathcal{B}_2 \arrow[u, "<>_2"] \arrow[r, "\beta"', "*"] & \cdots\cdots \arrow[r, "\beta"', "*"] & \mathcal{B}_i \arrow[u, "<>_i"] \arrow[r, "\beta"', "*"] & \cdots\cdots
\end{tikzcd}$$

Die Existenz der $\mathcal{B}_i$ und der $<>_i$ ist durch die Sätze 4.3.6, 4.3.8, 4.3.9, 4.3.10 und 4.3.11 gesichert. Offensichtlich gilt:

(1) $|\mathcal{B}_0| \geq |\mathcal{B}_1| \geq \ldots \geq |\mathcal{B}_i| \geq 2$ für alle $i \geq 0$.

Es gibt folglich ein $m \geq 1$ mit:

(2) $2 \leq |\mathcal{B}_m| = |\mathcal{B}_{m+1}| = |\mathcal{B}_{m+2}| = \ldots$.

Das wiederum bedeutet, daß für alle $i \geq m+1$ mit $\mathrm{Bild}(\mathcal{B}_i) \not\subset \mathcal{G}_{i+1}$ gilt, daß ein Knoten a in $\mathrm{Bild}(\mathcal{B}_i)$ existiert, so daß $\mathcal{G}_i \xrightarrow[\sigma]{a} \mathcal{G}_{i+1}$ ein reduzierender σ-Schritt ist. Da nach Satz 4.3.11 ein solcher reduzierender σ-Schritt auf der $\mathcal{G}$-Ebene mindestens einen reduzierenden σ_B-Schritt auf der $\mathcal{B}$-Ebene erzeugt, und nach Lemma 4.1.17 nur endlich viele aufeinanderfolgende reduzierende σ_B-Schritte auf der $\mathcal{B}$-Ebene möglich sind, existiert folglich ein n, für das gilt:

(3) $\mathrm{Bild}(\mathcal{B}_n) \subset \mathcal{G}_i$ für alle $i \geq n$.

Da $\square \notin S(\mathcal{G}_n)$ ist, enthält $\mathrm{Bild}(\mathcal{B}_n)$ nach Satz 4.3.5(c) mindestens einen unären Non-τ-Link. Dieser wird wegen (3) nicht mehr gelöscht. Also ist Γ nicht unär erschöpfend. *W!* [ ]

4.4.4 <u>Unär erschöpfende Filter</u>. Ein Filter $\mathcal{F}$ heißt unär erschöpfend, wenn jede

nichtabbrechende $\mathcal{F}$-Ableitung, die vom initialen Graphen $\mathcal{G}_S$ eine UWF-Klauselmenge S ausgeht, unär erschöpfend ist.

4.4.5 <u>Satz</u> (Kriterium). Jeder Filter $\mathcal{F}$, der die folgenden drei Bedingungen erfüllt, ist für unäre Klauselmengen vollständig:

(a) $\mathcal{F}$ ==> BSERF.

(b) $\mathcal{F}$ ist unär erschöpfend.

(c) Jede $\mathcal{F}$-Ableitung Γ, die keine Widerlegung ist und deren finaler Graph einen unären Non-τ-Link enthält, ist $\mathcal{F}$-fortsetzbar.

Beweis. Sei $\mathcal{F}$ ein Filter, der die Bedingungen (a), (b) und (c) erfüllt. Wir müssen zeigen, daß $\mathcal{F}$ für UF-Klauselmengen noethersch und konfluent ist.

Teil 1: $\mathcal{F}$ ist für UF-Klauselmengen noethersch. Wir müssen zeigen, daß keine nichtabbrechende $\mathcal{F}$-Ableitung existiert, die vom initialen Graphen $\mathcal{G}_S$ einer UWF-Klauselmenge S ausgeht.

Angenommen, Γ ist eine solche nichtabbrechende $\mathcal{F}$-Ableitung. Da $\mathcal{F}$ unär erschöpfend ist, ist auch Γ unär erschöpfend. Außerdem ist Γ eine nichtabbrechende BSERF-Ableitung, da $\mathcal{F}$ ==> BSERF. Nach Satz 4.4.3 kann aber ein solches Γ nicht existieren. ///

Teil 2: $\mathcal{F}$ ist für UF-Klauselmengen konfluent. Wegen Satz 2.6.15 müssen wir lediglich zeigen, daß $\mathcal{F}$ für UF-Klauselmengen fortsetzbar ist. Sei S also eine UWF-Klauselmenge und sei $\Gamma: \mathcal{G}_S \vdash\!\!- \mathcal{G}$ eine $\mathcal{F}$-Ableitung, die keine Widerlegung ist. Da $\mathcal{F}$ die Bedingung (a) erfüllt, gilt $\mathcal{G}_S \vdash_{BSE} \mathcal{G}$. Satz 4.4.1 liefert, daß eine Widerlegung $\mathcal{G} \xrightarrow[\rho]{*} \square$ existiert, bei der nur auf unären Non-τ-Links resolviert wird. Da $\mathcal{G}$ die leere Klausel nicht enthält, muß $\mathcal{G}$ einen unären Non-τ-Link enthalten. Die Bedingung (c) liefert nun, daß Γ $\mathcal{F}$-fortsetzbar ist, was zu zeigen war. [ ]

4.4.6 <u>Korollar</u>. Die BF-Strategie ist für UF-Klauselmengen vollständig.

Beweis. Wir zeigen, daß die BF-Strategie das Kriterium 4.4.5 erfüllt.

Bedingung (a): Offensichtlich gilt BF ==> NR ==> BSERF.

Bedingung (b): Offensichtlich ist M$\wedge$FF erschöpfend und damit insbesondere unär erschöpfend. Wegen BF ==> M$\wedge$FF ist auch BF unär erschöpfend.

Bedingung (c): Trivial. [ ]

4.4.7 <u>Die MUP-Strategie</u>. Eine BSERF-Ableitung Γ heißt MUP-Ableitung, wenn für jeden ρ-Schritt $\mathcal{G} \xrightarrow[\rho]{\ell} \mathcal{G}'$ in Γ eine der folgenden zwei Bedingungen erfüllt ist:

(a) $\mathcal{G}$ enthält keinen unären Link.

(b) ℓ ist ein unärer Link mit $\mathrm{Gen}(\ell) = \min\{\mathrm{Gen}(\ell') \mid \ell'$ ist unärer Link in $\mathcal{G}\}$.

MUP steht für "monotone unit preference". Offensichtlich ist die MUP-Strategie unär erschöpfend. Bei der MUP-Strategie können die ϕ- und die τ-Regeln ohne Einschränkung angewendet oder *nicht* angewendet werden.

4.4.8 <u>Korollar</u>. Die MUP-Strategie ist für UF-Klauselmengen vollständig.

Beweis. Offensichtlich sind alle Bedingungen des Kriteriums 4.4.5 erfüllt. [ ]

L I T E R A T U R V E R W E I S E

Am75 T. AMBLE, A connection graph theorem prover. Intitut for Databehandling, Universitetet i Trondheim, 1975.

An76 P.B. ANDREWS, Refutations by matings. IEEE Trans. Comp. C-25 (1976), 801-807.

An81 P.B. ANDREWS, Theorem proving via general matings. J. ACM 28,2 (Apr. 1981), 193-214.

BEHSSW81 K. BLÄSIUS, N. EISINGER, A. HEROLD, J.H. SIEKMANN, G. SMOLKA und C. WALTHER, The Markgraf Carl refutation procedure. 7th Int. Joint Conf. on Artificial Intelligence, Vancouver, B.C., Canada, 1981, 511-518.

Bi81a W. BIBEL, A strong completeness result for the connection graph proof procedure. Technische Universität München, 1981.

Bi81b W. BIBEL, On the completeness of connection graph resolution. Proc. of GWAI-81, Springer Informatik Fachberichte 47, 1981, 246-247.

Bi81c W. BIBEL, On matrices with connections. J. ACM 28,4 (Oct. 1981), 633-645.

Bi82a W. BIBEL, A comparative Study of several proof procedures. Artif. Int. 18 (1982), 269-293.

Bi82b W. BIBEL, Automated theorem proving. Vieweg, Wiesbaden, 1982.

Br75 M. BRUYNOOGHE, The inheritance of links in a connection graph. Report CW2, Katholieke Universiteit Leuven, 1975.

Br76 F. BROWN, Notes on chains and connection graphs. Personal Notes, Dep. of Computation and Logic, Edinburgh University, 1976.

Ch70 C.L. CHANG, The unit proof and the input proof in theorem proving. J. ACM 17,4 (Oct. 1970), 698-707.

Ch78 D. de CHAMPEAUX, A theorem prover dating a semantic network. Proc. of the AISB/GI Conference on Artif. Int., Hamburg, W. Germany, 1978.

Ch82 D. de CHAMPEAUX, A note on resolution, connection graphs, and subsumption. University of Amsterdam, 1982.

CL73 C.L. CHANG und R.C. LEE, Symbolic logic and mechanical theorem proving. Academic Press, 1973.

Co65 P.M. COHN, Universal Algebra. Harper and Row, New York & London, 1965.

CS79 C.L. CHANG und J.R. SLAGLE, Using rewriting rules for connection graphs to prove theorems. Artif. Int. 12 (1979), 159-180.

DP60 M. DAVIS und H. PUTNAM, A computing procedure for quantification theory. J. ACM 7,3 (July 1960), 201-215.

Ei80 N. EISINGER, Connectiongraphen und Subsumptionsregeln. Diplomarbeit, Universität Karlsruhe, 1980.

Ei81 N. EISINGER, Subsumption and connection graphs. 7th Int. Joint Conf. on Artificial Intelligence, Vancouver, B.C., Canada, 1981, 480-486.

Ga77 Z. GALIL, On the complexity of regular resolution and the Davis-Putnam procedure. Theoret. Comput. Sci. 4 (1977), 23-46.

He29 J. HERBRAND, Recherches sur la théorie de la démonstration. 1929. Englische Übersetzung in /vH67/, 529-567.

He68 H. HERMES, Enumerability, decidability, Countability. Springer Verlag, New York, 1965.

He82 A. HEROLD, Basic notions in first order unification theorie. Universität Karlsruhe, erscheint 1982.

Ho51 A. HORN, On sentences which are true of direct unions of algebras. J. Symb. Logic 16 (1951), 14-21.

HR78 M.C. HARRISON und N. RUBIN, Another generalization of resolution. J. ACM 25,3 (July 1978), 341-351.

Hu80 G. HUET, Confluent reductions: abstract properties and applications to term rewriting systems. J. ACM 27,4 (Oct. 1980), 797-821.

HW74 L. HENSCHEN und L. WOS, Unit refutations and Horn sets. J. ACM 22,4 (Oct. 1974), 590-605.

Ko75 R. KOWALSKI, A proof procedure using connection graphs. J. ACM 22,4 (Oct. 1975), 572-595.

Ko79 R. KOWALSKI, Logic for problem solving. North Holland, 1979.

Ku72 D. KUEHNER, Some special purpose resolution systems. 1972. In /MM72/, 163-190.

Le78 H.R. LEWIS, Renaming a set of clauses as a Horn set. J. ACM 25,1 (Jan. 1978), 134-135.

Lo78 D.W. LOVELAND, Automated theorem proving: a logical basis. North Holland, 1978.

MM70 B. MELTZER und D. MICHIE (Eds), Machine intelligence 5. American Elsevier, New York, 1970.

MM72 B. MELTZER und D. MICHIE (Eds), Machine intelligence 7. Edinburgh University Press, 1972.

Ni71 N. NILSSON, Problem solving methods in artificial intelligence. McGraw-Hill, New York, 1971.

PW78 M.S. PATERSON und M.N. WEGMAN, Linear unification. J. of Computer and System Sciences 16 (1978), 158-167.

Re70 J.C. REYNOLDS, Transformational systems and the algebraic structure of atomic formulas. 1970. In /MM70/, 135-152.

Ro65 J.A. ROBINSON, A machine-oriented logic based on the resolution principle. J. ACM 12,1 (Jan. 1965), 23-41.

Sh76 R.E. SHOSTAK, Refutation graphs. Artif. Int. 7 (1976), 51-64.

Si76 S. SICKEL, A search technique for clause interconnectivity graphs. IEEE Trans. Comp. C-25 (1976), 823-835.

Sk28 T. SKOLEM, Über die mathematische Logik. Englische Übersetzung in /vH67/, 512-524.

SS76 J.H. SIEKMANN und W. STEPHAN, Completeness and soundness of the connection graph proof procedure. Universität Karlsruhe, 1976.

SS80 J.H. SIEKMANN und W. STEPHAN, Completeness and consistency of the connection graph proof procedure. Universität Karlsruhe, 1980.

SW80 J.H. SIEKMANN und G. WRIGHTSON, Paramodulated connection graphs. Acta Informatica 13 (1980), 67-86.

Ts68 G.S. TSEITIN, On the complexity of derivations in the propositional calculus. 1968. In A.O. SLISENKO (Ed), Structures in constructive mathematics and mathematical logic, Part II, 1968, 115-125.

vH67 J. VAN HEIJENOORT, From Frege to Gödel: a source book in mathematical logic, 1879-1931. Harvard University Press, 1967.

VK76 M.H. VAN EMDEN und R. KOWALSKI, The semantics of predicate logic as a programming language. J. ACM 23,4 (Oct. 1976), 733-742.

Wa81a C. WALTHER, Elimination of redundant links in extended connection graphs. Proc. of GWAI-81, Springer Informatik Fachberichte 47, 1981, 201-213.

Wa81b C. WALTHER, Elimination of redundant links in extended connection graphs. Interner Bericht 10/81, Institut für Informatik I, Universität Karlsruhe, 1981.

YRH70 R.A YATES, B. RAPHAEL und T.P. HART, Resolution graphs. Artif. Int. 1 (1970), 257-289.

S A C H V E R Z E I C H N I S

S Y M B O L V E R Z E I C H N I S