2.4 Hornformeln

wichtiger, effizient zu behandelnder Spezialfall (benannt nach Alfred Horn)

Def.: (Hornformel)
Eine Formel F ist eine Hornformel, falls F in KNF ist und jedes Konjunktionsglied (also jede Disjunktion) in F höchstens ein positives Literal enthält.

Bsp.: $(A \vee \neg B) \wedge (\neg C \vee \neg A \vee D) \wedge (\neg A \vee \neg B) \wedge D \wedge \neg E$

äquivalent lässt sich das durch folgende Implikationen darstellen:
$(B \to A) \wedge (C \wedge A \to D) \wedge (A \wedge B \to 0) \wedge D \wedge (E \to 0)$

Die negativen Literale werden (nichtnegiert) Vorbedingungen der Implikation, das positive (wenn existent) Nachbedingung (Kopf). Falls es kein positives Literal gibt, wird 0 als Kopf verwendet. 0 steht hier für eine unerfüllbare Formel, bedeutet also Widerspruch. Falls es keine negativen Literale gibt, so entsteht wie im Fall von D ein Fakt (das entspricht einer Implikation mit einer Tautologie als Vorbedingung, Schöning schreibt deshalb $(1 \to D)$, D ist aber üblicher).

Beobachtung: die Implikationen enthalten keine Negation!

Effizienter Erfüllbarkeitstest für Hornformeln:

Eingabe: Hornformel F

1. Markiere jedes Vorkommen von A in F, falls A Fakt ist;
2. **while** es gibt in F Teilformel $G = (A_1 \wedge ... \wedge A_n \to B)$ oder $(A_1 \wedge ... \wedge A_n \to 0)$ mit
   A_1 und ... und A_n markiert, B unmarkiert **do**
   if G hat erste Form **then** markiere jedes Vorkommen von B
   else gib aus "unerfüllbar" und stoppe;
3. gib aus "erfüllbar" und stoppe.

Die erfüllende Belegung ergibt sich aus den markierten Atomen: $I(A_i) = 1$ gdw. A_i markiert.

Satz: Obiger Markierungsalgorithmus ist korrekt und stoppt nach maximal n Markierungsschritten (n = Anzahl der Atome in F).

Beweisskizze:
Komplexitätsabschätzung offensichtlich, da jedes Atom nut einmal markiert werden kann.

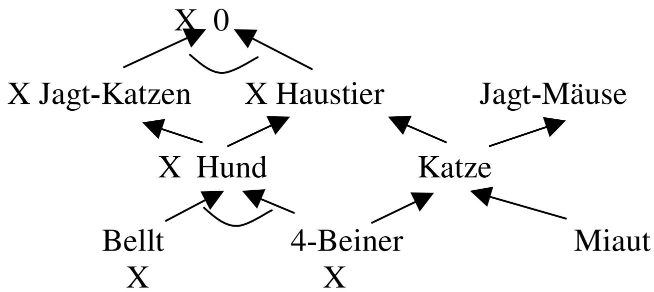
Korrektheit: Schritte 1 und 2 markieren nur solche Atome, die in allen Modellen von F den Wert 1 haben müssen. Falls es dazu kommt, dass 0 markiert werden müsste (else-Fall), kann es kein Modell geben und die Ausgabe in 2. ist korrekt.
Ansonsten ist nach Verlassen von 2. die Belegung I für die gilt: $I(A_i) = 1$ gdw. A_i markiert, ein Modell von F, denn für jede Implikation G in F gilt: entweder die Vorbedingung und die Nachbedingung von G sind wahr in I oder die Vorbedingung von I ist falsch. In beiden Fällen ist die Implikation wahr.

Beispiel: Betrachte folgende „Wissensbasis" WB (wir lassen die Konjunktion zwischen den Implikationen weg):

4-Beiner ∧ Miaut → Katze	4-Beiner ∧ Bellt → Hund
Katze → Haustier	Hund → Haustier
Katze → Jagt-Mäuse	Hund → Jagt-Katzen
4-Beiner	Bellt

Haustier ∧ Jagt-Katzen ableitbar?
gdw WB ∧ ¬(Haustier ∧ Jagt-Katzen) unerfüllbar
gdw WB ∧ (Haustier ∧ Jagt-Katzen → 0) unerfüllbar.

Repräsentation als Graph, Markierung X:



Bemerkung 1: Falls die zu testende Formel erfüllbar ist, berechnet der Algorithmus ein Modell von F (darin sind alle markierten Atome wahr, alle anderen falsch). Dieses Modell ist das kleinste Modell von F. Hierbei gilt:

M ≤ M' gdw M(A) ≤ M'(A) für alle atomaren Formeln A (wie üblich ist 0 < 1).

Also: M wird als kleiner betrachtet, wenn die Menge der wahren Atome eine Teilmenge der wahren Atome in M' ist.
Hornformeln besitzen immer ein kleinstes Modell. Alle Atome, die in diesem Modell wahr sind, sind in allen Modellen wahr und damit auch folgerbar.
Für allgemeine Formeln kann es mehrere minimale Modelle geben (etwa A ∨ B). (Warum?)

Frage: Hätten wir nicht ohne Widerspruchsbeweis auskommen und einfach prüfen können, ob Jagt-Katzen und Haustier markiert ist? In diesem Fall ja, aber das klappt nicht immer:

(¬A ∨ ¬B) ∧ A ergibt: (A ∧ B → 0) ∧ A

¬B ist hieraus offensichtlich folgerbar. Aber: im Graphen ist kein Knoten für ¬B, deshalb müssen wir hier einen Widerspruchsbeweis machen, also testen, ob (A ∧ B → 0) ∧ A ∧ B erfüllbar ist (was offensichtlich nicht der Fall ist).

Bemerkung 2: Aus dem Algorithmus ergibt sich unmittelbar: Falls eine Hornformel keine Implikation der Form (A_1 ∧ ... ∧ A_n → 0) enthält, so ist sie immer erfüllbar. Ebenso wenn kein Fakt der Form A vorkommt.

2.5 Endlichkeitssatz (Kompaktheitstheorem)

Der folgende Satz ist von Bedeutung insbesondere für Beweisverfahren, die wir später kennenlernen werden.

Satz: Eine Menge M von Formeln ist erfüllbar gdw. jede der endlichen Teilmengen von M erfüllbar ist.

Beweisskizze (für mehr Details siehe Schöning):
"=>" offensichtlich, da Modell für M auch Modell für jede Teilmenge von M.
"<=". Für jedes n $\geq$ 1 sei Mn die Menge der Formeln in M, die nur Atome A1, ..., An enthalten. Es gibt k $\leq$ 2 hoch 2 hoch n verschiedene, nicht äquivalente Formen in Mn. Also: für jedes F in Mn gibt es i $\leq$ k mit F $\equiv$ Fi. Jedes Modell für {F1, ..., Fk} ist Modell für Mn. So ein Modell existiert nach Voraussetzung. Wir nennen es I_n. Es ist gleichzeitig Modell für M1, M2, ..., Mn-1 (denn diese Mengen sind Teilmengen von Mn).

Das gesuchte Modell I für M wird so konstruiert (wir schreiben (An, 1) in I, statt I(An) = 1). Ebenso für 0):

```
Stufe 0:      I := {};
              J := {1,2,3 ...}.

Stufe n > 0:  if es gibt unendlich viele Indizes i in J mit Ii(An) = 1 then
                    begin  I := I ∪ {(An,1)};
                           J := J - {i | Ii(An) ≠ 1}
                    end
              else  begin  I := I ∪ {(An,0)};
                           J := J - {i | Ii(An) ≠ 0}
                    end.
```

In jeder Stufe n wird I um (An,0) oder (An,1) erweitert, aber nie um beides. I ist daher wohldefinierte Funktion mit Definitionsbereich {A1, A2, ...} und Wertebereich {0,1}.

Wir zeigen: I ist Modell für M. Sei F beliebige Formel in M. In F kommen endlich viele Atome vor, der höchste Index eines solchen Atoms sei k, d.h. F in Mk, Mk+1, ... und jede der Belegungen Ik, Ik+1, ... ist Modell von F.

Bei der Konstruktion von I wird J zwar "ausgedünnt", aber nie endlich. Damit verbleiben auch unendlich viele Indizes i > k in J. Für all diese i gilt: $I_i(A_1) = I(A_1)$, ..., $I_i(A_k) = I(A_k)$, und deshalb ist I Modell von F.

Bemerkung: Beweis nicht-konstruktiv: if-Bedingung nicht algorithmisch überprüfbar. Reine Gedankenkonstruktion, I kann nicht tatsächlich von Algorithmus berechnet werden.

Satz später in folgender Form verwendet: wenn unendliche Menge unerfüllbar ist, dann gibt es eine endliche Teilmenge, die bereits unerfüllbar ist.

2.6 Resolution

Resolution ist eine syntaktische Umformungsregel, die aus zwei Formeln eine neue erzeugt (die Resolvente). Resolution wird verwendet, um Unerfüllbarkeit einer Formel(menge) zu testen.

Anmerkung:
Mengen von syntaktischen, mechanisch ausführbaren Umformungsregeln heißen auch Kalküle. Kalküle erzeugen aus vorgegebenen Mengen von Formeln neue.

Kalküle heißen	korrekt,	wenn sie nur folgerbare Formeln ableiten,
	vollständig,	wenn sie alle folgerbaren Formeln ableiten

Resolution setzt voraus, dass Formel(menge) in KNF vorliegt (sonst Umformen).

Bequem ist Darstellung der KNF in Mengennotation: statt
$F = (L_{1,1} \vee \ldots \vee L_{1,n1}) \wedge \ldots \wedge (L_{k,1} \vee \ldots \vee L_{k,nk})$
schreiben wir
$F = \{\{L_{1,1}, \ldots, L_{1,n1}\}, \ldots, \{L_{k,1}, \ldots, L_{k,nk}\}\}$

Jedes Element von F heißt Klausel.

Def.: (Resolvente)
Seien K1, K2 Klauseln. R heißt Resolvente von K1 und K2, falls es ein Literal L gibt mit L in K1 und –L in K2 und es gilt: $R = (K1 - L) \cup (K2 - –L)$.
Hierbei ist $–L = \neg L$ falls $L=Ai$, $–L = Ai$ falls $L = \neg Ai$.

Beispiel: $\{A,\neg C\}$ ist Resolvente aus $\{A, B\}$ und $\{\neg B, \neg C\}$.

[Frage: Kann Resolution von Hornformeln nicht-Hornformel produzieren?]

Wichtiger Spezialfall: die leere Klausel {} entsteht durch Resolvieren von widersprüchlichen Einerklauseln, etwa $\{A\},\{\neg A\}$. Sie entspricht falsch und wird auch als $\perp$ oder [] repräsentiert.

Die Vorgehensweise bei der Überprüfung, ob eine Formel G aus einer Formel F folgerbar ist, ist wie folgt:

- Wir testen mit Resolution, ob $F \wedge \neg G$ unerfüllbar ist. Dazu muss zunächst die KNF von $F \wedge \neg G$ erzeugt werden.
- Wir erzeugen systematisch Resolventen (aus der KNF von $F \wedge \neg G$ sowie aus bereits erzeugten Resolventen).
- Wird auf diese Weise irgendwann [] erzeugt, so ist $F \wedge \neg G$ unerfüllbar und damit G aus F folgerbar
- wird [] nicht erzeugt, so ist $F \wedge \neg G$ erfüllbar und G nicht folgerbar aus F.

Wir müssen natürlich noch beweisen, dass diese Vorgehensweise auch sinnvoll ist, d.h. dass wir auf diese Weise die richtigen Resultate erhalten. Wir zeigen zunächst, dass Resolution korrekt ist, dass also Resolventen folgerbar sind aus den sie erzeugenden Klauseln:

Resolutionslemma:
Sei F eine Formel in KNF, dargestellt als Klauselmenge, R Resolvente zweier Klauseln aus F. Dann sind F und $F \cup \{R\}$ äquivalent (und damit R aus F folgerbar).

Bew.: Sei I eine Belegung. Zu zeigen: $I \models F \cup \{R\}$ gdw. $I \models F$.
Falls $I \models F \cup \{R\}$, dann gilt offensichtlich auch $I \models F$.
Es gelte umgekehrt $I \models F$. Dann gilt $I \models K$ für jede Klausel K in F. Ferner sei $R = (K1 - L) \cup (K2 - -L)$.
Fall 1: $I \models L$. Dann gilt $I \models K2 - -L$ und deshalb $I \models R$.
Fall 2: $I \not\models L$. Dann gilt $I \models K1 - L$ und deshalb $I \models R$.

Bemerkung: Resolvente ist nicht äquivalent zu beiden resolvierten Klauseln. Deshalb werden resolvierte Klauseln nicht ersetzt.
Beispiel: $A \vee \neg B$, $B \vee C$ ==> $A \vee C$.
$\{\neg A, B, C\}$ ist Modell der Resolvente, aber nicht der resolvierten Klauseln.

Wir wissen also, jetzt, dass Resolution korrekt ist. Ist Resolution auch vollständig in dem Sinn, dass ich alle aus einer Formel folgerbaren Formeln herleiten kann? Die Antwort ist nein. Z.B. folgt $A \vee B$ aus A, aber $A \vee B$ ist nicht per Resolution aus A ableitbar.
Wir werden jedoch sehen, dass Resolution widerlegungsvollständig ist. Das bedeutet, dass aus unerfüllbaren Klauselmengen immer [] hergeleitet werden kann.

Um präziser über Resolution reden zu können, benötigen wir noch folgende Definitionen:

Def.: Sei F eine Klauselmenge. Wir definieren:

$$\begin{aligned} Res(F) &= F \cup \{R \mid R \text{ ist Resolvente zweier Klauseln in } F\} \\ Res^0(F) &= F \\ Res^{n+1}(F) &= Res(Res^n(F)) \\ Res^*(F) &= \bigcup_{n \geq 0} Res^n(F) \end{aligned}$$

Frage: Bestimme $Res^1(F)$ für $F = \{\{A,\neg B,C\}, \{B,C\},\{\neg A,C\}, \{B,\neg C\},\{\neg C\}\}$

Res^1: $F \cup \{\{A,C\}, \{\neg B,C\}, \{A,C,\neg C\}, \{A,B,\neg B\},\{A,\neg B\}, \{B\}, \{\neg A,B\},\{\neg A\}\}$

Bemerkung: Wieviele verschiedene Klauseln aus n Atomen kann es geben? 4^n (jedes Ai kann nicht , pos, neg, pos+neg vorkommen). Daraus folgt für endliche Klauselmengen:
es gibt ein k, so dass $Res^k(F) = Res^{k+1}(F) = ... = Res^*(F)$.

Def.: (Deduktion) Eine Deduktion (Herleitung, Resolutionsbeweis) einer Klausel C aus einer Klauselmenge F ist eine Folge $K_1, ..., K_m$ von Klauseln, für die gilt:

1) $K_m = C$,
2) für alle i ($1 \leq i \leq m$): entweder K_i aus F oder K_i Resolvente von K_j, K_k mit $j,k < i$.

Bemerkung 1: durch die lineare Anordnung wird erreicht, dass neue Klauseln nur aus bereits hergeleiteten Klauseln bzw. Klauseln in F erzeugt werden. Unerwünschte Zyklen werden so vermieden.
Bemerkung 2: Übersichtlicher als Herleitungen sind oft Repräsentationen mit so genannten Resolutionsgraphen. Knoten in diesen Graphen sind Klauseln, eine Kante führt jeweils von

den für Resolution verwendeten Klauseln zur Resolvente (Beispiele folgen). Diese Darstellungen lassen sich ineinander überführen.
Res* und Deduktion hängen natürlich eng zusammen. Es gilt:

Satz: Sei F eine Klauselmenge, C eine Klausel. Es gilt $C \in Res^*(F)$ gdw. es eine Herleitung von C aus F gibt.
Beweis: Übung

Wir wollen nun Widerlegungsvollständigkeit der Resolution beweisen. Dazu ist folgendes Lemma hilfreich:

Lemma: Sei F eine Klauselmenge, A ein Atom in F. $F_{A=0}$ entstehe aus F durch

1) Streichen der Klauseln, in denen $\neg A$ vorkommt, sowie
2) Streichen jedes Vorkommens von A in einer Klausel.

(Belegung von A mit 0 fixieren.) Analog entstehe $F_{A=1}$ aus F durch

3) Streichen der Klauseln, in denen A vorkommt, sowie
4) Streichen jedes Vorkommens von $\neg A$ in einer Klausel.

(Belegung von A mit 1 fixieren.) Es gilt: F ist unerfüllbar genau dann wenn $F_{A=0}$ und $F_{A=1}$ unerfüllbar sind.

Beweis: Sei F erfüllbar. Dann gibt es ein Modell M von F. Falls $M(A) = 1$, so ist M auch ein Modell von $F_{A=1}$ (jede in $F_{A=1}$ enthaltene Klausel K muss von M zu 1 ausgewertet werden, denn sonst würde die ursprüngliche Klausel in F, die entweder K oder $K \cup \{\neg A\}$ ist, nicht zu 1 ausgewertet). Falls $M(A) = 0$, so ist M auch ein Modell von $F_{A=0}$ (jede in $F_{A=0}$ enthaltene Klausel K muss von M zu 1 ausgewertet werden, denn sonst würde die ursprüngliche Klausel, die entweder K oder $K \cup \{A\}$ ist, nicht zu 1 ausgewertet).

Sei F unerfüllbar. Gäbe es ein Modell für $F_{A=0}$ oder für $F_{A=1}$, so könnte daraus ein Modell für F konstruiert werden, indem man entweder das Modell für $F_{A=0}$ so ergänzt, dass A zu 0 interpretiert wird, oder das Modell für $F_{A=1}$ so ergänzt, dass A zu 1 interpretiert wird.

Resolutionssatz (der Aussagenlogik)
Eine Klauselmenge F ist unerfüllbar gdw. $[] \in Res^*(F)$.

Beweis: "<=" folgt aus Resolutionslemma: F äquivalent $Res^n(F)$ für alle n. Da [] in $Res^k(F)$ für ein k und damit $Res^k(F)$ unerfüllbar ist, so muss auch F unerfüllbar sein.
"=>" Sei F unerfüllbar. Falls F unendlich, so muss es wegen Kompaktheit endliche Teilmenge von F geben, die unerfüllbar ist. Wir können uns deshalb auf endliche Formelmengen beschränken.
Wir zeigen $[] \in Res^*(F)$ durch Induktion über die Anzahl n der in F vorkommenden Atome:
Induktionsanfang: $n = 0$, dann muss $F = \{[]\}$ sein und damit $[] \in Res^*(F)$.
Induktionsschritt: Sei F eine Klauselmenge mit Atomen $A_1, \ldots, A_{n+1}$. Wir konstruieren aus F die beiden Klauselmengen $F_{An+1=0}$ und $F_{An+1=1}$. Da F unerfüllbar ist, müssen auch diese beiden Mengen unerfüllbar sein. Da sie nur noch n Atome enthalten, ist die Induktionsvoraussetzung auf sie anwendbar.

Es muss also einen Resolutionsbeweis $K_1, ..., K_m$ aus $F_{An+1=0}$ für [] geben. Benutzt man nun statt $K_1, ..., K_m$ die ursprünglichen Klauseln aus F, so erhält man einen Resolutionsbeweis entweder wieder für [], oder für A_{n+1}. Im ersten Fall ist bewiesen, dass die leere Klausel aus F ableitbar ist.

Ebenso gibt es einen Resolutionsbeweis für [] aus $F_{An+1=1}$, der durch Verwendung der ursprünglichen Klauseln zu einem Beweis für [] oder für $\neg A_{n+1}$ wird. Wieder ist im ersten Fall die Behauptung bewiesen. Ansonsten lässt sich aus A_{n+1} und $\neg A_{n+1}$ in einem weiteren Resolutionsschritt die leere Klause ableiten, d.h. $[] \in$ Res*(F). qed.

Zur Veranschaulichung des Beweises: betrachte F = {{¬A,¬B}, {¬A, B}, {A, ¬B}, {A, B}}. Wenn F unerfüllbar ist, dann kann es kein Modell M geben mit M(B) = 0. Also muss $F_{B=0}$ = {{¬A}, {A}}} unerfüllbar sein (entstanden durch Streichen der ersten und dritten Klausel in F und Streichen von B aus den anderen Klauseln), denn sonst gäbe es so ein Modell M (einfach das Modell für $F_{B=0}$ um die Belegung für B (= 0) erweitern).
Die Herleitung der leeren Klausel ist {¬A}, {A}, []. Verwendet man wieder die ursprünglichen Klauseln, so ist die Herleitung {¬A,¬B}, {A,¬B}, {¬B}.
Analog: Wenn F unerfüllbar ist, dann kann es kein Modell M geben mit M(B) = 1. Also muss $F_{B=1}$ = {{¬A}, {A}}} unerfüllbar sein (entstanden durch Streichen der zweiten und vierten Klausel in F und Streichen von ¬B aus den anderen Klauseln), denn sonst gäbe es so ein Modell M (einfach das Modell für $F_{B=1}$ um die Belegung für B (= 1) erweitern).
Die Herleitung der leeren Klausel ist {¬A}, {A}, []. Verwendet man wieder die ursprünglichen Klauseln, so ist die Herleitung {¬A,B}, {A,B}, {B}.
Insgesamt bekommen wir damit eine Herleitung für die leere Klausel aus F, nämlich {¬A,¬B}, {A,¬B}, {¬B}, {¬A,B}, {A,B}, {B}, [].

Aus Resolutionssatz lässt sich folgender Erfüllbarkeitstest ableiten:

```
Eingabe: Formel in KNF (repräsentiert als Klauselmenge)
repeat  G := F;
        F := Res(F)
until ([] in F) or (F = G);
if [] in F then "nein, unerfüllbar" else "ja, erfüllbar";
```

Dieser Algorithmus terminiert immer. Außerdem wissen wir aufgrund des Resolutionssatzes, dass er immer die richtige ja/nein-Antwort liefert. Es handelt sich damit um ein Entscheidungsverfahren.

Es muss natürlich nicht immer ganz Res* berechnet werden, es genügt, eine Herleitung der leeren Klausel zu finden:

früheres Beispiel: ist G = Haustier ∧ Jagt-Katzen ableitbar aus:

WB:

1)	4-Beiner ∧ Miaut → Katze	{¬4-Beiner, ¬Miaut, Katze}
2)	4-Beiner ∧ Bellt → Hund	{¬4-Beiner, ¬Bellt, Hund}
3)	Katze → Haustier	{¬Katze, Haustier}

4)	Hund → Haustier	{¬Hund, Haustier}	
5)	Katze → Jagt-Mäuse	{¬Katze, Jagt-Mäuse}	
6)	Hund → Jagt-Katzen	{¬Hund, Jagt-Katzen}	
7)	4-Beiner	{4-Beiner}	
8)	Bellt	{Bellt}	

Negation von G:

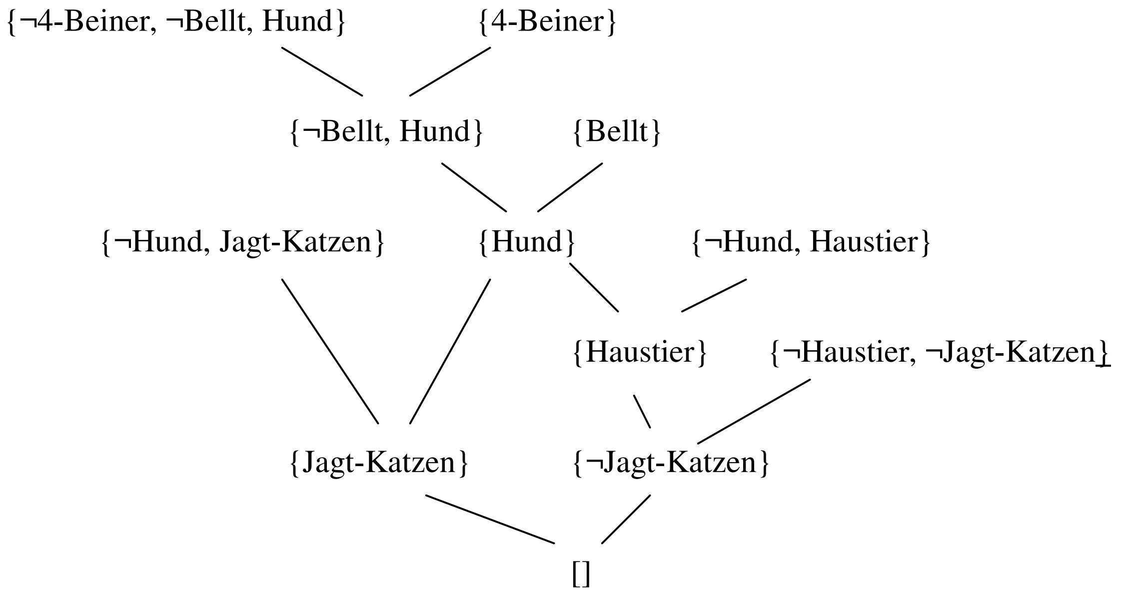
9)	¬Haustier ∨ ¬Jagt-Katzen	{¬Haustier, ¬Jagt-Katzen}	

Resolventen:

10)		{¬Bellt, Hund}	2,7
11)		{Hund}	10,8
12)		{Haustier}	11,4
13)		{¬Jagt-Katzen}	12,9
14)		{Jagt-Katzen}	11,6
15)		[]	14,13

Die Folge der Klauseln 1) … 15) ist eine Deduktion von []. 1), 3) und 5) sind nicht verwendet worden und können aus der Deduktion noch gestrichen werden.

Resolutionsgraph:



Weiteres Beispiel: zeige, dass (¬B ∧ ¬C ∧ D) ∨ (¬B ∧ ¬D) ∨ (C ∧ D) ∨ B Tautologie ist.
Ist {B, C,¬D}, {B, D}, {¬C, ¬D},{¬B} unerfüllbar? ja.

Def.: Einschränkungen der Resolutionsverfahrens:
Unit-Resolution entsteht durch Beschränkung der Resolution auf Fälle, in denen mindestens eine resolvierte Klausel einelementig ist (Einer-Klausel).
Input-Resolution entsteht durch Beschränkung der Resolution auf Fälle, in denen mindestens eine resolvierte Klausel Eingabe-Klausel (aus F) ist.

Satz: Sowohl Unit- wie Input-Resolution sind widerlegungsvollständig für Hornklauseln, d.h. falls F Hornklauselmenge, so ist die leere Klausel ableitbar aus F gdw. F unerfüllbar.

Katzen-Beispiel bereits Unit-Resolution.

Gegenbeispiel im allgemeinen Fall (Unit): {¬A,¬B}, {¬A, B}, {A, ¬B}, {A, B}.
unerfüllbar, aber natürlich keine Unit-Resolution möglich.

Bei der linearen Resolution wird in jedem (bis auf den ersten) Resolutionsschritt die vorher erzeugte Resolvente verwendet (zusammen mit einer Klausel, die entweder in F ist oder vorher erzeugt wurde).

Satz: Lineare Resolution ist widerlegungsvollständig.

obiges Beispiel mit linearer Resolution:

{¬4-Beiner, ¬Bellt, Hund}   {4-Beiner}

{¬Bellt, Hund}   {Bellt}

{Hund}   {¬Hund, Haustier}

{Haustier}   {¬Haustier, ¬Jagt-Katzen}

{¬Jagt-Katzen}   {¬Hund, Jagt-Katzen}

{¬Hund}   {Hund}

[]