2.7 Das Davis-Putnam-Verfahren

Erfüllbarkeitstest für Formel $F = \{C_1, ..., C_n\}$ in Klauselform

Def.: (reduzierte Klauselmenge)
Sei F eine Menge von Klauseln, L ein Literal. Die um L reduzierte Klauselmenge F_L entsteht aus F durch

a) Streichen aller Klauseln, die L enthalten
b) Streichen des Komplements von L aus den verbleibenden Klauseln.

Komplement von L: A falls $L = \neg A$, $\neg A$ falls L = A (A Atom). F_L entspricht offensichtlich $F_{A=1}$ (wie in Abschn. 2.6 definiert) falls L = A, $F_{A=0}$ falls $L = \neg A$.

Beispiel: $F = \{\{A, B\}, \{C, \neg A\}, \{\neg B\}\}$

Reduzieren um A ergibt:
$F_A = \{\{C\}, \{\neg B\}\}$

Reduzieren um ¬B ergibt:
$F_{\neg B} = \{\{A\}, \{C, \neg A\}\}$

Anmerkung (bereits im Abschnitt Resolution implizit gezeigt):
jedes Modell M von F, das L zu wahr auswertet, ist Modell von F_L.
Zu jedem Modell M' von F_L gibt es ein Modell M von F, das die Atome in F_L gleich auswertet wie M'. (M(L) = 1, M(A) = M'(A) für alle Atome, die in F_L vorkommen)
also: F_L erfüllbar genau dann wenn F Modell hat, das L zu wahr auswertet.

Beobachtungen:

1. Wenn [] in F, dann ist F unerfüllbar.
2. Wenn F leer, dann ist F erfüllbar.
3. Wenn Einerklausel der Form {L} in F, so ist F erfüllbar genau dann wenn F_L erfüllbar.
4. Sei L Literal in F. F ist erfüllbar gdw. F_L erfüllbar oder F_{-L} erfüllbar.

Daraus lässt sich direkt der folgende rekursive Erfüllbarkeitstest ableiten:

```
boolean Satisfiable(S)          ;; liefert true falls Klauselmenge S erfüllbar, false sonst
begin
    if S = {} then return true;
    if [] ∈ S then return false;
    if S enthält Einerklausel select Einerklausel {L} and return Satisfiable(S_L)
    else select Literal L and return (Satisfiable(S_L) or Satisfiable(S_-L));
end;
```

Beispiel 1:

erfüllbar $\{\{A, B\}, \{C, \neg A\}, \{\neg B\}\}$	gdw.
erfüllbar $\{\{A\}, \{C, \neg A\}\}$	gdw.
erfüllbar $\{\{C\}\}$	gdw.
erfüllbar $\{\}$	true

Beispiel 2:

erfüllbar {{A, B}, {¬A, B}, {A, ¬B}}?		gdw. (wähle A)
erfüllbar {{B}}	oder erfüllbar {{B}, {¬B}}	gdw.
erfüllbar {}	oder erfüllbar {{}}	true

Das Verfahren lässt sich veranschaulichen durch einen Graphen, der die jeweils zu überprüfenden Klauselmengen enthält. Wir haben den Begriff Graph bisher informell verwendet und wollen die Gelegenheit nutzen, ihn auch formal korrekt einzuführen.

Exkurs Graphentheorie

Def.: (gerichteter Graph, Pfad) Ein gerichteter Graph G = (V, E) besteht aus einer nichtleeren Menge V (der Knotenmenge des Graphen) sowie einer Menge $E \subseteq V \times V$, der Menge der Kanten von G. Falls $(v_1, v_2) \in E$, so heißt v_1 Vorgänger(knoten) von v_2, v_2 Nachfolger(knoten) von v_1. Eine Folge von Knoten $(v_1,...,v_n)$ heißt Pfad von v_1 nach v_n, wenn für $1 \leq i < n$ gilt: $(v_i,v_{i+1}) \in E$.

Anmerkung: man zeichnet gerichtete Graphen, indem man alle Knoten darstellt und für jede Kante (v_i,v_j) einen Pfeil zwischen v_i und v_j zeichnet. Wenn die Richtung implizit ist (etwa weil man wie in unseren Resolutionsgraphen davon ausgeht, dass Kanten immer zu weiter unten stehenden Knoten führen) kann man statt Pfeil auch einfach eine Linie zeichnen.

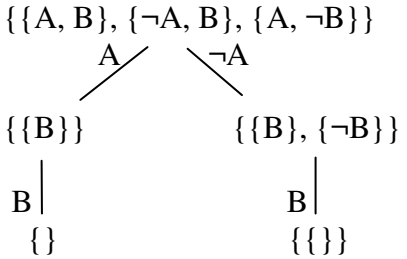
Bei einem markierten Graphen gibt es zusätzlich noch eine Abbildung m, die Knoten oder Kanten (oder beiden) eine Markierung zuordnet (auch Label genannt). Die Menge der Markierungen/Labels hängt von der jeweiligen Anwendung ab.

Def.: (Baum) Die Menge der (orientierten)[1] Bäume ist induktiv wie folgt definiert:

1. Wenn V einelementig ist, so ist der Graph G = (V, {}) ein Baum.
2. Sei G = (V, E) ein Baum, $v_1 \in V$, $v_2 \notin V$, dann ist G' = $(V \cup \{v_2\}, E \cup \{(v_1, v_2)\})$ ein Baum.

Knoten eines Baumes ohne Nachfolger heißen Blätter, der Knoten ohne Vorgänger (es gibt immer genau einen!) heißt Wurzel.

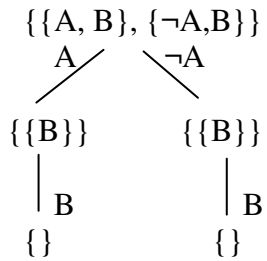
Dem obigen 2. Beispiel entspricht folgender Baum (Knoten entsprechen Aufrufen von Satisfiable; sie sind markiert mit der jeweils zu testenden Klauselmenge; die Kanten sind zusätzlich noch mit den Literalen markiert, die zum Reduzieren verwendet wurden):



[1] Es gibt auch nicht-orientierte Bäume, die aber in der Vorlesung keine Rolle spielen. Wir werden deshalb das Adjektiv „orientiert" weglassen und nur von Bäumen sprechen.

Die Formel ist erfüllbar, wenn mindestens 1 Blatt des Baumes mit der leeren Menge markiert ist. In diesem Fall kann man die erfüllende Belegung von der Beschriftung der Kanten ablesen.

Anmerkung: Es kann passieren, dass dieselbe Klauselmenge mehrmals im Baum auftritt. Das ist auch der Grund dafür, dass wir zwischen dem Knoten v und der Markierung des Knotens m(v) unterscheiden: jeder Knoten kann nur einmal auftreten, aber verschiedene Knoten können dieselbe Markierung haben. Man ist bei der Darstellung häufig nur an der Markierung interessiert und stellt dann nur diese im Baum dar. Beispiel (Baum mit 5 Knoten und 3 verschiedenen Markierungen):



2.8 Tableauverfahren

Tableau für Formel F:
Baum, dessen Knoten mit Formeln markiert sind, die Wurzel mit F.
Pfad von Wurzel zu Blatt repräsentiert die Konjunktion der Formeln an Knoten,
der gesamte Baum die Disjunktion dieser Konjunktionen.

Grundidee: F ist unerfüllbar, wenn für jeden Pfad von der Wurzel zu einem Blatt gilt: die entsprechende Konjunktion ist unerfüllbar.

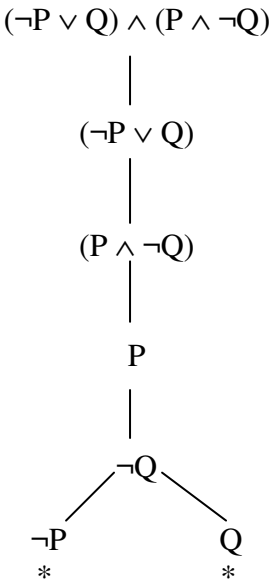
Erzeugungsregeln für Tableaus:

$\frac{\neg\neg H}{H}$	$\frac{G_1 \wedge G_2}{G_1 \\ G_2}$	$\frac{\neg(G_1 \wedge G_2)}{\neg G_1 \mid \neg G_2}$	$\frac{G_1 \vee G_2}{G_1 \mid G_2}$	$\frac{\neg(G_1 \vee G_2)}{\neg G_1 \\ \neg G_2}$

zu lesen:
wenn in Pfad $\neg\neg H$ vorkommt, erweitere ihn um H (Erweitern um H: Nachfolgerknoten einführen und mit H markieren)
wenn in Pfad $G_1 \wedge G_2$ vorkommt, erweitere ihn um G_1 und um G_2,
wenn in Pfad $\neg(G_1 \wedge G_2)$ vorkommt, verzweige und erweitere um linken Nachfolger $\neg G_1$ und rechten Nachfolger $\neg G_2$,
etc.

Beim Erzeugen des Tableaus werden die auftretenden Teilformeln immer einfacher. Die Idee dabei ist, dass sie irgendwann so einfach sind, dass man sofort „sehen“ kann, ob eine Inkonsistenz in einem Pfad vorliegt, d.h. ob die entsprechende Konjunktion unerfüllbar ist.

Beispiel:



Def.: (Tableau für F)
Die Menge der Tableaus für eine Formel F ist induktiv wie folgt definiert:

1) der Baum, der aus einem mit F markierten Knoten besteht, ist ein Tableau für F,
2) wenn T ein Tableau für F ist und T' durch Anwendung einer Erzeugungsregel aus T entsteht, so ist T' ein Tableau für F.

Def.: (Ast, abgeschlossen)
Ein Ast eines Tableaus ist ein Pfad von der Wurzel zu einem Blatt.
Ein Ast heißt abgeschlossen, wenn in ihm eine Formel und ihre Negation vorkommt.
Ein Tableau heißt abgeschlossen, wenn jeder Ast abgeschlossen ist.

Satz: F ist unerfüllbar gdw. es ein abgeschlossenes Tableau für F gibt.

Im Beispiel: linker Ast abgeschlossen, weil ¬P und P enthalten, rechter weil ¬Q und Q enthalten.

Vorteile der diskutierten Verfahren:

Resolution:	lässt sich relativ einfach auf Prädikatenlogik erweitern
Davis-Putnam:	für Aussagenlogik oft effizienter als Resolution
Tableaus:	keine Umformung in KNF nötig.