2.6 Davis-Putman-Algorithmen

- Erfüllbarkeits-Algorithmen
- Formeln in NNF ($\neg$, $\wedge$, $\vee$)
- Bottom-Up Verfahren - Festlegung einer erfüllenden Bewertung durch Auswahl der Werte der Atome

2.45 Definition

Sei A A-Form, $p \in \mathbb{V}$ definiere $A[p/1]$ (bzw. $A[p/0]$) als Ergebnis des folgenden Ersetzungsprozesses:

1. Ersetze in A jedes Vorkommen von p durch 1.
2. Tritt nun eine Teilform $\neg 1$ auf, ersetze sie durch 0,
   $\neg 0$ ersetze durch 1.
   Teilformeln $B \wedge 1$, sowie $B \vee 0$ werden durch B ersetzt,
   Teilformeln $B \vee 1$ durch 1 und
   Teilformeln $B \wedge 0$ durch 0 ersetzt.
3. Schritt 2 wird so lange durchgeführt, bis keine weitere Ersetzung möglich ist.

Analog für $A[p/0]$.

Allgemeiner verwende $A[l/1]$ bzw. $A[l/0]$ für Literale l.

Beachte: Für A in KNF und Literal l gilt: $A[l/1]$ entsteht aus A durch Streichen aller Klauseln, die das Literal l enthalten und durch Streichen aller Vorkommen des Literals $\neg l$ in allen anderen Klauseln.

$A[p/1]$ (bzw. $A[p/0]$) sind wohldefiniert. (Warum ?)

Davis-Putman-Algorithmen (Forts.)

Als Ergebnis des Ersetzungsprozesses $A[p/i]$ $i = 1, 0$ erhält man:

- eine A-Form (in NNF bzw. KNF wenn A diese Form hatte)
- 1 die „leere Formel“
- 0 die „leere Klausel“ ($\sqcup$, $\square$)

Die leere Formel wird als wahr interpretiert. Die leere Klausel als falsch (nicht erfüllbar), d. h. $A[p/i]$ kann als A-Form behandelt werden.

2.46 Lemma Für jedes Atom $p \in \mathbb{V}$ und $A \in F$ gilt:

1. $A[p/i]$ $i \in 1, 0$ ist entweder die leere Formel oder die leere Klausel oder eine A-Form in NNF in der p nicht vorkommt.
2. $A \wedge p \models\!\!=\!\!\models A[p/1] \wedge p$ $\qquad$ $A \wedge \neg p \models\!\!=\!\!\models A[p/0] \wedge \neg p$
3. A ist erfüllbar gdw $A[p/1] = 1$ oder $A[p/0] = 1$ oder eine der Formeln $A[p/1]$, $A[p/0]$ erfüllbar ist.

Durch Testen der Teilbewertungen $A[p/1]$ und $A[p/0]$ kann rekursiv die Erfüllbarkeit von A entschieden werden.

Regelbasierter Aufbau von DP-Algorithmen

2.47 Definition Regeln für Formeln in NNF

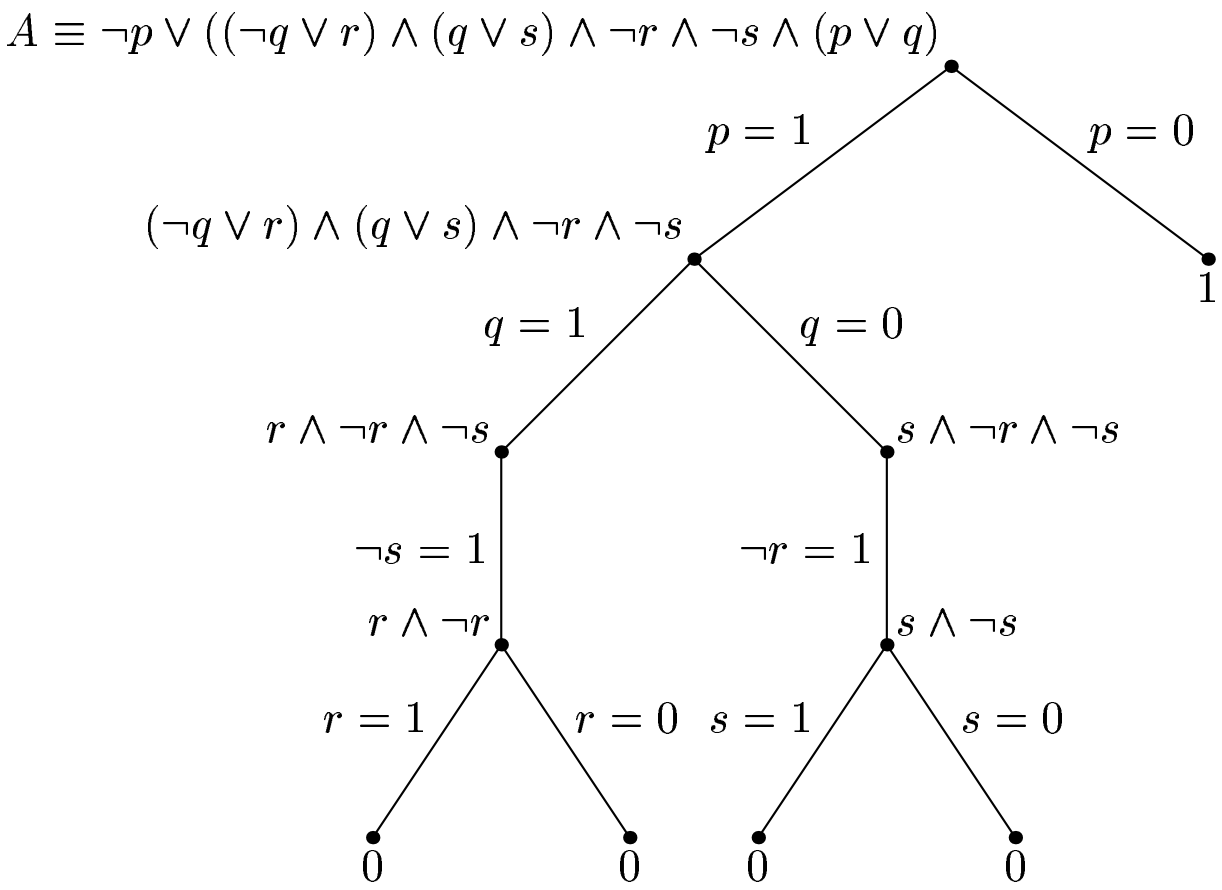
1. **Pure-Literal Regel**
   Kommt ein Atom $p \in \mathbb{V}$ in einer A-Form A nur positiv oder nur negativ vor, so können wir p mit 1 bzw. 0 belegen und die Formel dementsprechend kürzen.
   (Es gilt $A[p/0] \models A[p/1]$ bzw. $A[p/1] \models A[p/0]$), genauer A erfüllungsäquivalent $A[p/1]$ bzw. A erfüllungsäquivalent $A[p/0]$.
2. **Splitting-Regel**
   Kommt jedes Atom sowohl positiv als auch negativ vor, so wähle ein solches Atom p in A aus und bilde aus A die zwei A-Formen $A[p/1]$ und $A[p/0]$.
   Die Ausgangsformel A ist genau dann erfüllbar, wenn bei einer der Kürzungen der Wert 1 oder eine erfüllbare Formel als Ergebnis auftritt.

Regeln reduzieren das Erfüllbarkeitsproblem für eine Formel mit n Atomen auf EP für Formeln mit maximal $(n-1)$ Atomen.

Algorithmen, die mit Hilfe dieser beiden Regeln mit verschiedenen Heuristiken (zur Auswahl des splitting Atoms) und weiteren Verfeinerungen arbeiten, werden als **Davis-Putman-Algorithmen bezeichnet**.

Beispiel

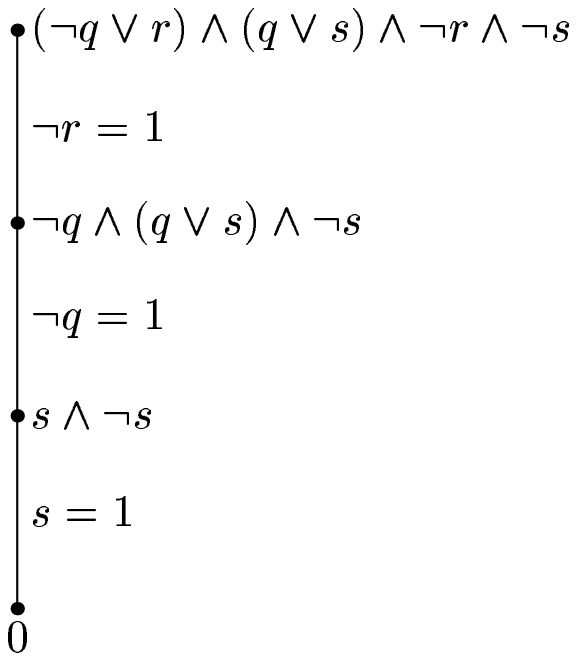
2.48 Beispiel : Darstellung der Abarbeitung als Baum



Weitere Verfeinerungen

2.49 Definition Regeln für Formeln in KNF

3. **Unit-Regel**
   Sei A in KNF und A enthält eine Unit Klausel $A_i \equiv l$. Bilde $A[l/1]$ (A ist erfüllbar gdw $A[l/1]$ erfüllbar).
   Da das Literal einer Unit-Klausel durch eine erfüllende Bewertung auf wahr gesetzt werden muss.



Seien A_1 und A_2 Klauseln. A_1 **subsumiert** A_2 ($A_1 \subseteq A_2$) gdw jedes Literal aus A_1 auch in A_2 auftritt.

Aus der Erfüllbarkeit einer Klausel folgt sofort die Erfüllbarkeit aller Klauseln, die sie subsumiert.

Verfahren DPA

4. **Subsumption-Rule**
   Sei A in KNF. Streiche aus A alle Klauseln, die von anderen subsumiert werden:: $SR(A)$.
   (Da in KNF alle Klauseln konjunktiv verknüpft sind, braucht man nur diejenigen zu berücksichtigen, die von keiner anderen subsumiert werden).

2.50 Definition Davis/Putman Algorithmen

Input: A in KNF
Output: Boolscher Wert für Erfüllbarkeit $(1, 0)$

begin
  if $A \equiv 1$ **then return**(A);
  if $A \equiv 0$ **then return**(A);
  $p :=$ **pure**(A, s); // *liefert Atom und Belegung, falls nur positiv oder nur negativ vorkommt, sonst* **null**
  if $p \neq$ **null then return**(DPA$(A[p/s])$);
  $p :=$ **unit**(A, s); //*Unit Klausel mit Belegung sonst* **null**
  if $p \neq$ **null then return**(DPA$(A[p/s])$);
  $A :=$ Subsumption_Reduce(A); //*entfernt subs. Klauseln*
  $p :=$ **split**(A); //*liefert Atom in* A
  if (DPA$(A[p/1]) = 1$) **then return**(1);
  return(DPA$(A[p/0])$);
end.

Auswahlkriterien für die Splitting Regel

- Wähle das erste in der Formel vorkommende Atom,
- wähle das Atom, welches am häufigsten vorkommt,
- $\cdots$ das in den kürzesten Klauseln am häufigsten vorkommt,
- wähle Atom mit $\sum_{p \text{ in } A_i} |A_i|$ minimal,
- berechne die Anzahl der positiven und negativen Vorkommen in den kürzesten Klauseln und wähle das Atom mit der größten Differenz.

2.7 Resolutions-Verfahren

Das **Resolutionskalkül** als Deduktionssystem operiert auf Klauselmengen, d. h. Formeln in KNF mit nur einer Schlussregel: Aus Klauseln $(A \vee l)$ und $(B \vee \neg l)$ kann eine neue Klausel $(A \vee B)$ erzeugt werden. **Ziel:** Leere Klausel zu erzeugen.

Klauseln als Mengen $(p \vee \neg q \vee p) \leftrightarrow \{p, \neg q\}$

$l \equiv p$ so $\neg l \equiv \neg p$ $\quad$ $l \equiv \neg p$ so $\neg l \equiv p$

2.51 Definition Resolutionsregel (Resolventenregel)

Seien A, B Klauseln, l ein Literal. l kommt in A und $\neg l$ kommt in B vor. Dann können A und B über l (bzw. $\neg l$) resolviert werden. Die **Resolvente** der Klauseln A und B ist die Klausel $(A \backslash \{l\}) \cup (B \backslash \{\neg l\})$. A und B sind die Elternklauseln der Resolvente

Schema $\dfrac{A \quad , \quad B}{Res_l(A, B) \equiv (A \backslash \{l\}) \cup (B \backslash \{\neg l\})}$ (Resolutionsregel)

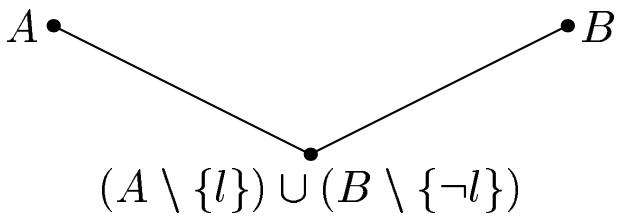
Beachte:

- Enthält die Resolvente ein Literal l', so muss dieses bereits in A oder B enthalten sein.
- Schreibe auch $A \vee l,\ B \vee \neg l \underset{Res}{\vdash} A \vee B$.
- $A \wedge B$ erfüllbar gdw $A \wedge B \wedge Res_l(A, B)$ erfüllbar. gdw $Res_l(A, B)$ erfüllbar.
- $A \wedge B \models Res(A, B)$.
- Resolvente kann leere Klausel $\sqcup$ sein.

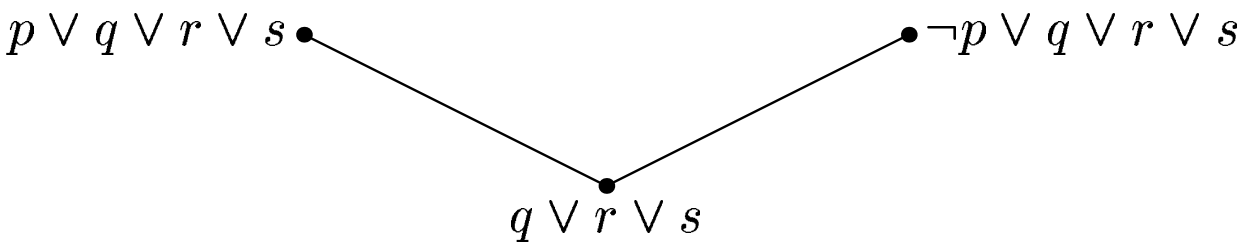
Darstellung - Beispiele

2.52 Beispiel

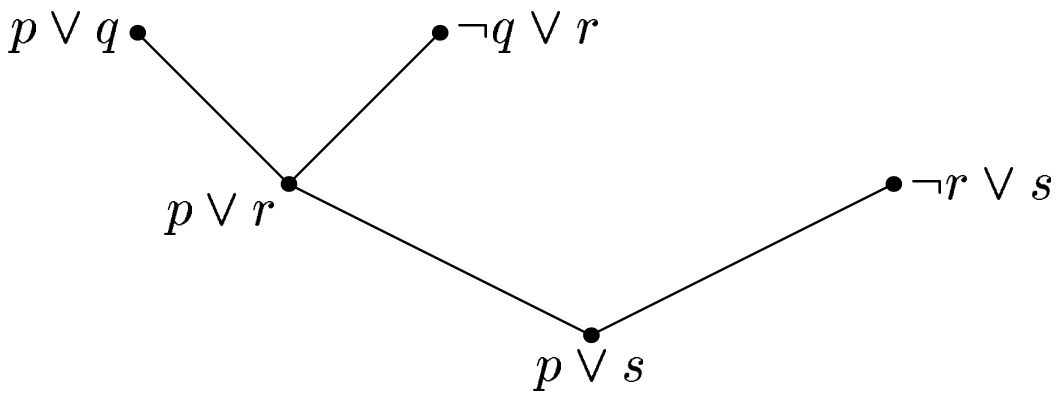
Darstellung für Klauseln A, B, die über l resolvieren



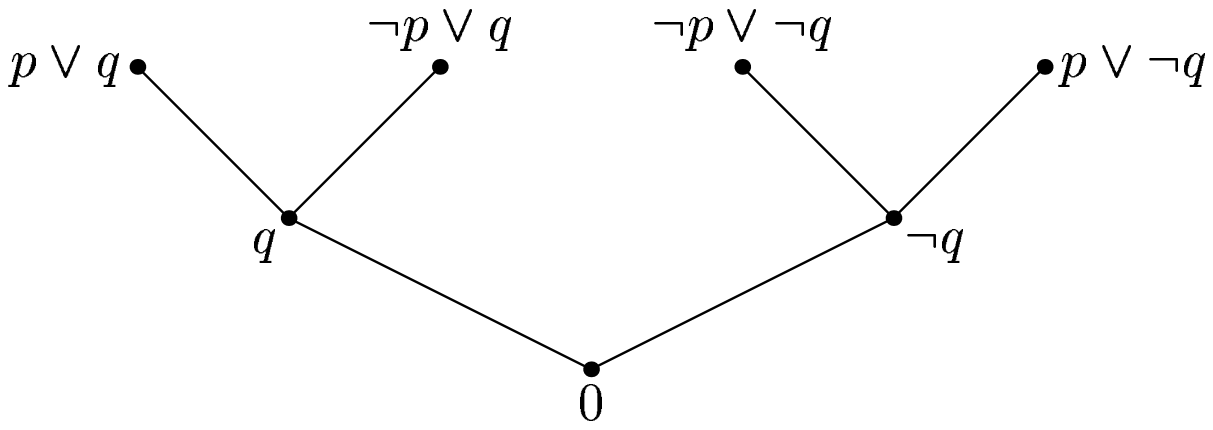
a) Formel $A = \{(p \vee q \vee r \vee s), (\neg p \vee q \vee r \vee s)\}$



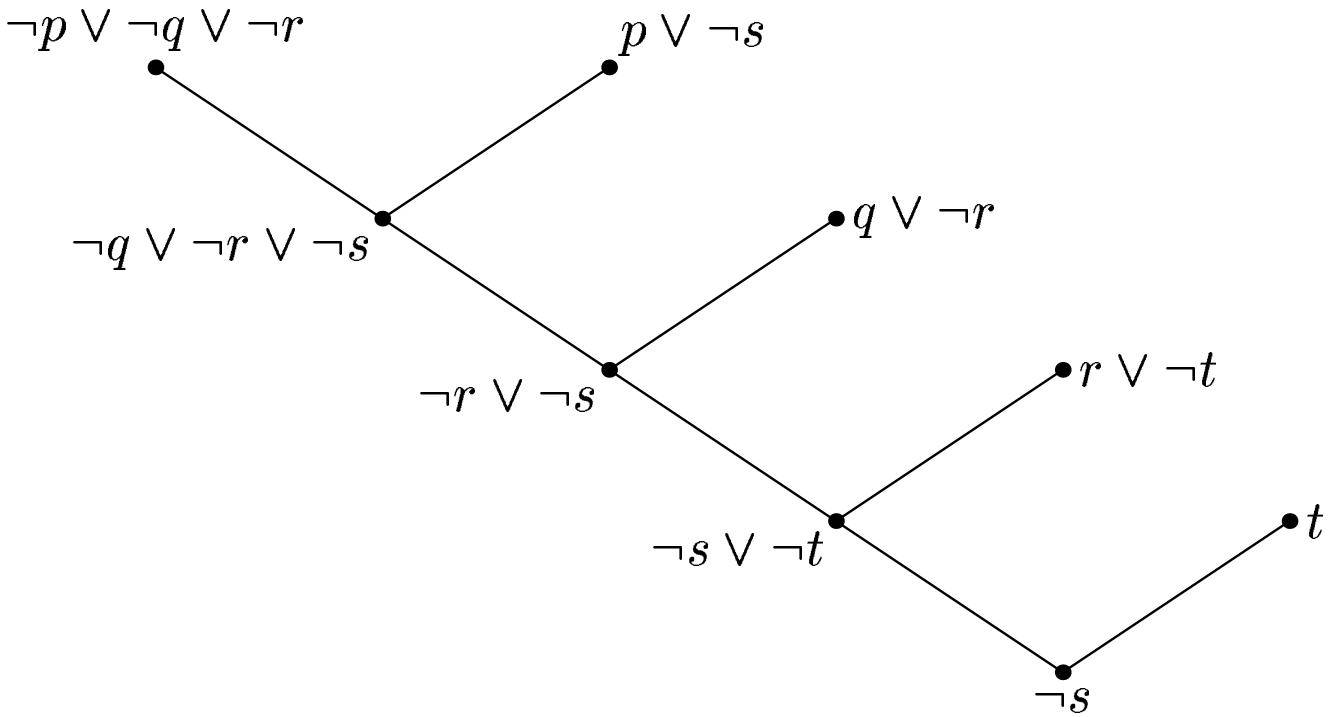
b) $A \equiv \{(p \vee q), (\neg q \vee r), (\neg r \vee s)\}$



c) $A \equiv \{(p \vee q), (\neg p \vee q), (p \vee \neg q), (\neg p \vee \neg q)\}$



d) $A \equiv \{(\neg p \vee \neg q \vee \neg r), (p \vee \neg s), (q \vee \neg r), (r \vee \neg t), t\}$
(Horn-Klauseln)



Ableitungen im Resolutionskalkül

2.53 Definition Herleitungen (Ableitungen)

Sei $A \equiv \{C_1, \ldots, C_n\}$ eine Formel in KNF und C ein Klausel. Eine Folge $D_1, \ldots, D_k$ von Klauseln ist eine **Herleitung** der Klausel C aus A. Wenn $C \equiv D_k$ und für alle j mit $1 \leq j \leq k$ Klauseln $E, F \in A \cup \{D_1, \ldots, D_{j-1}\}$ existieren mit $E, F \underset{Res}{\vdash} D_j$.

C ist (mit der Resolventenregel oder im Resolutionskalkül) **herleitbar** aus A

Schreibweise: $A \overset{+}{\underset{Res}{\vdash}} C$ (A Ausgangs-Klauseln)

k ist die Länge der Herleitung.

Minimale Herleitungen sind solche für die kein Schritt weggelassen werden kann.

Gilt $A \overset{+}{\underset{Res}{\vdash}} C_1$ und $A \overset{+}{\underset{Res}{\vdash}} C_2$, so schreibe $A \overset{+}{\underset{Res}{\vdash}} C_1, C_2$.

Darstellung von Herleitungen mit Hilfe von **DAG's**.

Korrektheit und Widerlegungsvollständigkeit

2.54 Satz

1. Der Resolutionskalkül ist korrekt.

   A in KNF, C Klausel dann $A \overset{+}{\underset{Res}{\vdash}} C$, so $A \models C$

2. Der Resolutionskalkül ist nicht vollständig.

   Es gibt A in KNF, C Klausel mit $A \models C$ aber nicht $A \overset{+}{\underset{Res}{\vdash}} C$

3. Der Resolutionskalkül ist widerlegungsvollständig.

   A in KNF, A widerspruchsvoll (unerfüllbar), so $A \overset{+}{\underset{Res}{\vdash}} \sqcup$

Beweis: 1. $\surd$, 2. $A \equiv p$, $C \equiv p \vee q \rightsquigarrow$ Behauptung.

3. Induktion nach Länge der Formel:

Kürzeste Formel:$\{(p), (\neg p)\}$, dann $p, \neg p \underset{Res}{\vdash} \sqcup$.

Verwende dabei: A widerspruchsvoll, so auch $A[p/1]$ und $A[p/0]$ widerspruchsvoll.

Sei A mit Länge $n+1$, A widerspruchsvoll. Es gibt ein Atom p in A das sowohl positiv als auch negativ vorkommt. Betrachte $A[p/1]$ und $A[\neg p/1]$, beide nicht erfüllbar. Angenommen nicht Wert 0.

Nach Ind.Vor.: $A[p/1] \overset{+}{\underset{Res}{\vdash}} \sqcup$, $A[p/0] \overset{+}{\underset{Res}{\vdash}} \sqcup$.

Korrektheit und Widerlegungsvollständigkeit (Forts.)

A in KNF. $A[p/1]$ entsteht durch Streichen der Klauseln, die p enthalten und durch Streichen von $\neg p$ aus Klauseln, die $\neg p$ enthalten. Fügt man in $A[p/1]$ die eliminierten Literale $\neg p$ und zu $A[\neg p/1]$ die Atome p wieder hinzu, so sind diese Formeln $A[p/1](\neg p)$ und $A[p/0](p)$ Teilformen von A.

Dann aber entweder $A[p/1](\neg p) \overset{+}{\underset{Res}{\vdash}} \sqcup$ bzw. $A[\neg p/1](p) \overset{+}{\underset{Res}{\vdash}} \sqcup$ auch aus A herleitbar oder
$A[p/1](\neg p) \overset{+}{\underset{Res}{\vdash}} \neg p$ und $A[\neg p/1](p) \overset{+}{\underset{Res}{\vdash}} p$.

Dann aber $\neg p, p \underset{Res}{\vdash} \sqcup$ und somit $A \overset{+}{\underset{Res}{\vdash}} \sqcup$.
Ergibt $A[p/1]$ oder $A[\neg p/1]$ den Wert 0, so enthält A eine Klausel p (falls $A[\neg p/1] = 0$) oder eine Klausel $\neg p$ (falls $A[p/1] = 0$). Dann analoger Schluss.

2.55 Lemma Sei A in KNF, C eine Klausel, dann gilt $A \models C$ gdw es gibt eine Teilklausel $C' \subseteq C$:

$$A \overset{+}{\underset{Res}{\vdash}} C'$$

Ist A erfüllbar und C Primimplikant von A, dann gilt $A \overset{+}{\underset{Res}{\vdash}} C$.

Resolventenmethode: Strategien/Heuristiken

Verfeinerungen der Methode

Starke Herleitungen: A in KNF, widerspruchsvoll.
Dann gibt es eine Herleitung $C_1, \ldots C_n \equiv \sqcup$ mit

1. in der Herleitung tritt keine Klausel mehrfach auf,
2. in der Herleitung tritt keine Tautologie auf,
3. in der Herleitung tritt keine schon subsumierte Klausel auf:
   Es gibt kein C_i, C_j, $j < i$, $C_j \subseteq C_i$.

Strategien, Heuristiken

- Stufenstrategie (Resolutionsabschluss)
  (Alle erfüllende Bewertungen)
- Stützmengenrestriktion
  (Set of Support, Unit-Klauseln bevorzugen)
- P-N Resolution
- Lineare Resolution (SL-Resolution)
  (PROLOG-Inferenzmaschine)

Beispiel: $A \equiv \{\neg p, \neg q, \neg r\}, \{p, \neg s\}, \{q, \neg r\}, \{r, \neg t\}, \{t\}\}$

Stufen:

0	1	2	3
1. $\{\neg p, \neg q, \neg r\}$	6. $\{\neg q, \neg r, \neg s\}$ (1,2)	1. $\{\neg r, \neg s\}$ (6,3)	21. $\{\neg s, \neg r\}$ (1,4)
2. $\{p, \neg s\}$	7. $\{\neg p, \neg r\}$ (1,3)	12. $\{\neg q, \neg s, \neg t\}$ (6,4)	22. $\{\neg s\}$ (11,10)
3. $\{q, \neg r\}$	8. $\{\neg p, \neg q, \neg t\}$ (1,4)	13. $\{\neg p, \neg t\}$ (7,4)	⋮
4. $\{r, \neg t\}$	9. $\{q, \neg t\}$ (3,4)	14. $\{\neg p, \neg r, \neg t\}$ (8,3)	
5. $\{t\}$	10. $\{r\}$ (4,5)	15. $\{\neg p, \neg q\}$ (8,5)	
		16. $\{q\}$ (10,3)	
		17. $\{\neg r, \neg s, \neg t\}$ (6,9)	
		18. $\{\neg q, \neg s\}$ (6,10)	
		19. $\{\neg p\}$ (7,10)	
		20. $\{\neg p, \neg t\}$ (8,9)	

$$\varphi(q) = 1, p(p) = 0, \varphi(s) = 0, \varphi(r) = 1, \varphi(t) = 1$$