Vorlesung
– Automatisches Beweisen –
Kap. 4.2: AL-Entscheidungsverfahren
Kap. 4.2.1 Aussagenlogische Resolution

Prof. Dr. Wolfgang Küchlin

Dipl.-Inform., Dr. sc. techn. (ETH)

Arbeitsbereich Symbolisches Rechnen
Wilhelm-Schickard-Institut für Informatik
Fakultät für Informations- und Kognitionswissenschaften

Universität Tübingen

Steinbeis Transferzentrum
Objekt- und Internet-Technologien (OIT)

Wolfgang.Kuechlin@uni-tuebingen.de
http://www-sr.informatik.uni-tuebingen.de

SR

Kap 4.2

Aussagenlogische Entscheidungsverfahren

SR

Aussagenlogische Entscheidungsverfahren

- Resolution
- Davis-Putnam-Logemann-Loveland (DPLL)
- Ordered Binary Decision Diagrams (OBDDs)
- Semantische Tableaus
- Boolesche Polynome (fehlen noch)

Kalkül

- **Kalkül** (= formales Regelsystem) zum Herleiten neuer Formeln aus gegebener Formelmenge
  - Formeln werden „mechanisch“ erzeugt
  - Daher eignet sich ein Kalkül zur Implementierung
- Schreibweise: $\mathcal{F} \vdash F$

  Formel F wird aus der Formelmenge $\mathcal{F}$ hergeleitet
- Sinnvoll ist das nur, falls die neue Formel auch „gilt“

  d.h. falls aus $\mathcal{F} \vdash F$ folgt $\mathcal{F} \vDash F$
  - wir sagen: der Kalkül ist korrekt (sound).
- Wenn wir alles Wahre ableiten können, ist der Kalkül vollständig (complete): aus $\mathcal{F} \vDash F$ folgt $\mathcal{F} \vdash F$
- Widerlegungsvollständig: aus $\mathcal{F} \vDash \perp$ folgt $\mathcal{F} \vdash \perp$

Resolution

- **Kalkül** zum Herleiten (aussagenlogischer) Formeln in CNF
  - **Deduktionsverfahren:** gegeben $\mathcal{F}$, leite neue Formel F daraus her.
- Entwickelt 1965 von **John Alan Robinson**
  - Ursprünglich für Prädikatenlogik erster Stufe
  - A machine-oriented logic based on the resolution principle. *Journal of the ACM* **12**(1), 23—41.
- Aus Axiomensystem $\mathcal{F}$ (in CNF) können neue gültige Formeln (Resolventen) F_i hergeleitet werden, $\mathcal{F} \vdash F_i$
- Erste Idee: beweise $\mathcal{F} \vDash F$ durch direkte Herleitung $\mathcal{F} \vdash F$
  - geht nicht immer, da Resolution nur widerlegungsvollständig
- Üblicher Umweg:
  - $\mathcal{F} \vDash F$ gdw. es gilt nie $\mathcal{F} \land \neg F$.
  - es gilt nie $\mathcal{F} \land \neg F$ gdw. $\mathcal{F} \land \neg F \vDash \bot$
  - da Resolution widerlegungsvollständig: $\mathcal{F} \land \neg F \vDash \bot$ gdw. $\mathcal{F} \land \neg F \vdash \bot$
  - Insgesamt: $\mathcal{F} \vDash F$ gdw. $\mathcal{F} \land \neg F \vdash_{Res} \bot$

Resolution

- Anwendung: Beweise **Unerfüllbarkeit** einer Klauselmenge $\mathcal{F}$ durch Herleiten der (unerfüllbaren) leeren Klausel {⊥}, also $\mathcal{F} \vdash^* \bot$
- Methode:
  - Wende **Inferenzregel(n)** zur Herleitung der leeren Klausel an.

    (Leere Klausel { }, gleichbedeutend mit {⊥}, im Kalkül üblicherweise symbolisiert durch □, ist nicht erfüllbar,)
  - **Axiome**: Klauseln der Ausgangsformel
  - Falls sich leere Klausel herleiten lässt, ist ein **Beweis** der Unerfüllbarkeit gefunden.

J. A. Robinson (Quelle: Wikipedia)

- **John Alan Robinson** (* 1930 in Yorkshire, Großbritannien) ist ein englischer Philosoph und Logiker, der wichtige Beiträge zur Logikprogrammierung geleistet hat.
- Nach einem abgeschlossenen Studium der Klassischen Altertums-wissenschaft an der Uni Cambridge ging er 1952 in die USA. Dort studierte er zunächst Philosophie an der U. Oregon und promovierte 1956 in Princeton zum Ph.D. Danach arbeitete er beim Chemiekonzern DuPont, wo er Programmieren und Mathematik lernte. 1961 wechselte er an die Rice University, wo er sich weiter mit Mathematik beschäftigte.
- 1965 veröffentliche er mit „A machine-oriented logic based on the resolution principle“ [J.ACM 12(1)] wichtige Grundlagen zur automatisierbaren Resolution in der Logik. Auf ihn geht ein Algorithmus zur Unifikation von prädikatenlogischen Formeln zurück, der entscheidend beim Nachweis der Unerfüllbarkeit einer prädikatenlogischen Formel ist.

Resolution (2)

- (Einzige) Inferenzregel:

$$\frac{C \cup \{l\} \qquad D \cup \{\neg l\}}{C \cup D}\,\mathrm{Res}$$

wobei $C \cup \{\ell\}$, $D \cup \{\neg\ell\}$ Klauseln.

- Beispiele:

$$\frac{\{x, y, \neg z\} \quad \{u, \neg v, z\}}{\{x, y, u, \neg v\}} \qquad \frac{\{x, u, \neg v\} \quad \{\neg u\}}{\{x, \neg v\}}$$

SR

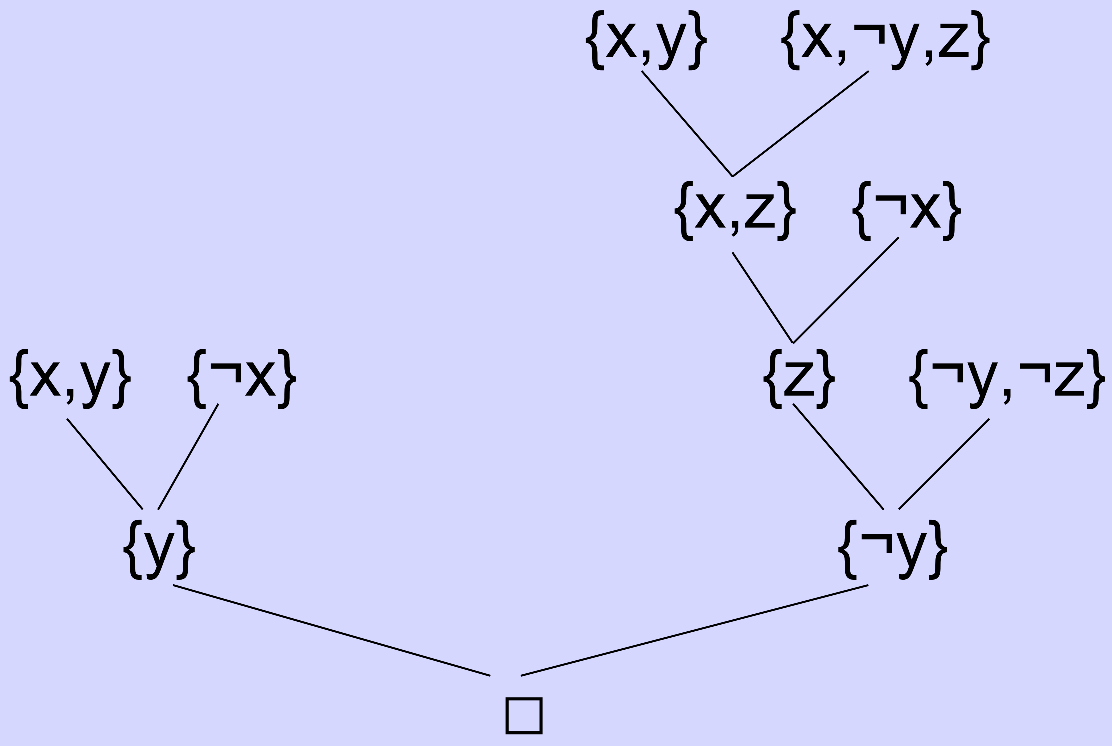
Resolution (3)

$$\frac{K_1 \quad K_2}{R}\,\text{Res}$$

- Begriffe allgemein bei Kalkül:
  - K_1, K_2: Prämissen
  - R: Konklusion
- Begriffe bei Resolutionsregel:
  - R: Resolvente *(resolvent)*
  - K_1, K_2: Eltern-Klauseln *(parent clauses)*
  - „R entsteht durch Resolution über ℓ aus K_1 und K_2“
  - Leere Klausel □ ist nicht erfüllbar, steht für $\{\perp\}$

Resolutionsbeweis: Beispiel

F3={{x,y},{x,¬y,z},{¬x},{¬y,¬z}}



Resolution (4)

- Herleitungsbegriff: $F \vdash C$
  - Aus der Klauselmenge F lässt sich durch eine endliche Anzahl von Anwendungen der Resolutionsregel die Klausel C herleiten
- Resolutionsabschluss:
  - $Res^0(F) = F$
  - $Res^1(F) = F \cup \{R \mid R$ (nicht-tautologische Resolvente zweier Klauseln K_1, K_2 aus F$\}$
  - $Res^{k+1}(F) = Res^1(Res^k(F))$ für $k \geq 1$
  - $Res^*(F) = \bigcup_{k \geq 0} Res^k(F)$

Subsumtion / Subsumption

- Eine Klausel C subsumiert *(subsumes)* eine Klausel D, genau dann wenn $C \subseteq D$.
  - subsumieren: unterordnen, unter etwas einordnen
  - Constraint C ist strenger als Constraint D
- Falls Klausel C die Klausel D subsumiert,

  dann gilt $C \models D$.
- Bei Erfüllbarkeitsprüfung und im Resolutionsbeweis können subsumierte Klauseln gelöscht werden.
  - sobald C erfüllt ist, ist auch D erfüllt
  - falls D nicht erfüllbar ist, so ist auch C nicht erfüllbar

Resolutionsverfahren

- Sei eine Menge von Axiomen (bzw. Constraints) C gegeben. Um einen Satz D zu beweisen, $C \vDash D$, verfahre wie folgt:

1. Negiere D. Bringe $C \cup \neg D$ in CNF. Nenne das Resultat F.
2. Wiederhole
   i. Bilde alle nicht-tautologischen Resolventen R von F
   ii. Falls $\square \in R$, return „proof“.
   iii. Lösche in R alle subsumierten Klauseln.
   iv. Falls R = { }, return „disproof“.
   v. Lösche in F alle durch R subsumierten Klauseln.
   vi. $F := F \cup R$

- Falls $C \vDash D$, so stoppt das Verfahren in (ii) wegen der Widerlegungs-Vollständigkeit (s.u.)
- Andernfalls stoppt das Verfahren in (iv), da es maximal 3^n subsumptionsfreie Klauseln in n Variablen (x, ¬x, don ‘t care x) gibt.

Resolution (5)

- Der Resolutionskalkül ist korrekt (sound). D.h. für alle Formeln F und alle Klauseln D gilt:

  $F \vdash^*_{Res} D$ impliziert $F \models D$

- Der Resolutionskalkül ist widerlegungsvollständig *(refutation complete)*. D.h. für alle Formeln F gilt:

  F unerfüllbar impliziert $F \vdash^*_{Res} \square$

  ($F \models \bot$ impliziert $F \vdash^*_{Res} \square$)

- Der Resolutionskalkül ist nicht vollständig.
  - Bsp: $x \wedge \neg x \models y$, aber y ist nicht durch Resolution aus $x \wedge \neg x$ ableitbar, denn es gibt nur eine einzige Resolvente $\square$.
  - wohl aber gilt $\{\{x\}, \{\neg x\}, \{\neg y\}\} \vdash^*_{Res} \square$

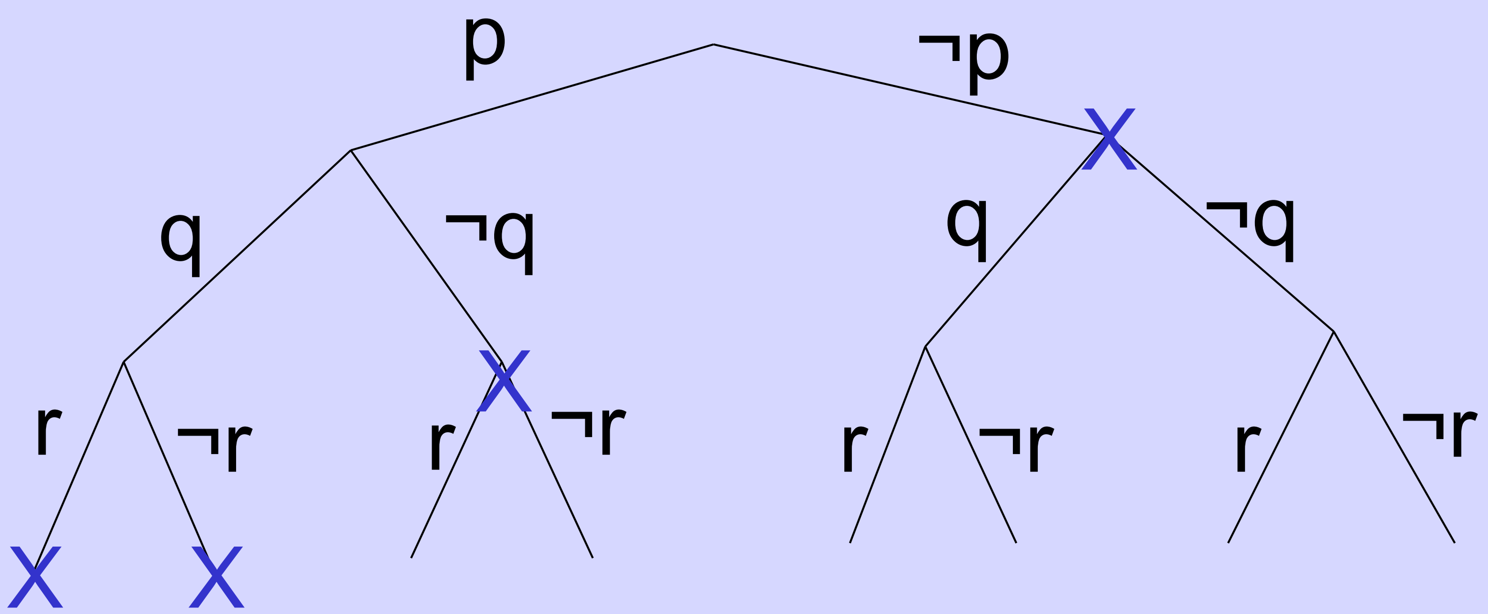
Resolution (6)

- Korrektheit:
  - Die Konklusion C∪D wird von den Prämissen C∪{ℓ}, D∪{¬ℓ} impliziert.
    - Jede Interpretation, die die Prämissen wahr macht, muss entweder ℓ oder ¬ℓ wahr machen.
    - Falls b(ℓ)=1, so b(D)=1; falls b(¬ℓ)=1, so b(C)=1.

      Also in jedem Fall b(C ∪ D)=1
  - Damit ist ein einzelner Beweisschritt korrekt. Korrektheit einer Ableitungskette folgt per Induktion
- Widerlegungs-Vollständigkeit
  - semantische Bäume: Repräsentation aller Belegungen als Baum
  - Auswirkungen eines Resolutionsschritts auf diesen Baum

Semantische Bäume

S = {{p}, {¬p,q}, {¬r}, {¬p,¬q,r}}



Belegung: Pfad im Baum
Fehlerknoten (X): Hier wird eine Klausel falsifiziert

Semantische Bäume (2)

- Jeder Zweig b in $\mathcal{T}$ definiert eine Interpretation β; gilt $\beta(F)=0$, dann heißt b geschlossen, sonst offen.
- $\mathcal{T}$ ist geschlossen, wenn alle Zweige geschlossen sind.

- $\mathcal{T}$ ist geschlossen, genau dann wenn F unerfüllbar.

Semantische Bäume (3)

- Ein Knoten eines Zweiges, der F falsifiziert und der Wurzel am nächsten ist, heißt Fehlerknoten.
- Eine Klausel, die durch einen Fehlerknoten falsifiziert wird, ist die mit diesem Fehlerknoten assoziierte Klausel.
- Ein Inferenzknoten ist ein Knoten, dessen Kinder beide Fehlerknoten sind.

Widerlegungsvollständigkeit der Resolution

- F unerfüllbar genau dann, wenn $\mathcal{T}$ geschlossen.
- In einem (nicht-trivialen) geschlossenen $\mathcal{T}$ gibt es mindestens einen Inferenzknoten n (mit Kindern n_1, n_2).
- Seien C_1 und C_2 die mit den Fehlerknoten n_1 und n_2 assoziierten Klauseln. Diese unterscheiden sich dann (mindestens) in dem unterhalb des Fehlerknotens belegten Literal. Wenn C_1 nach x=1 falsifiziert wird, dann ist $C_1 = C \cup \{\neg x\}$, entsprechend $D_1 = D \cup \{x\}$.
- Dann können C_1 und D_1 resolviert werden. Die partielle Interpretation in n falsifiziert die Resolvente $R = C \cup D$, da sowohl C als auch D schon am Knoten n falsifiziert werden.
- $F \cup \{R\}$ hat einen Fehlerknoten, der entweder n ist, oder ein Vorgänger von n, und R ist die mit diesem Knoten assoziierte Klausel.

Variante der Resolution: Unit-Resolution

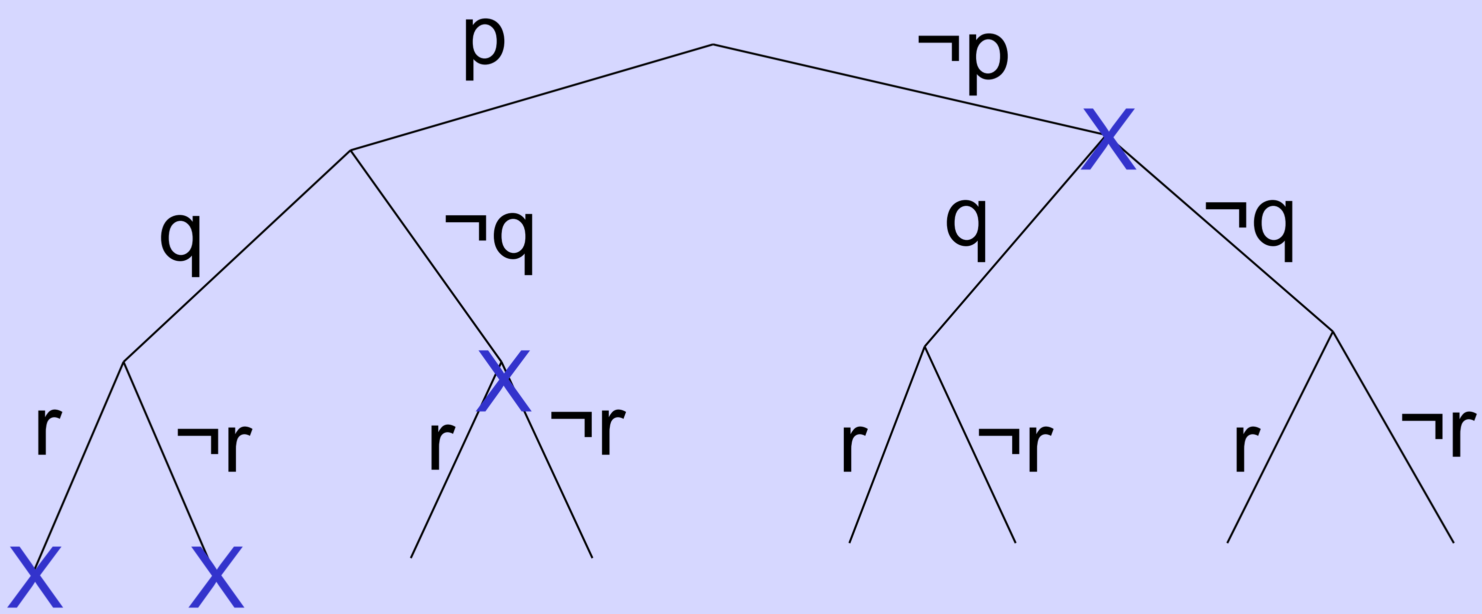
- Ziel: schränke Bildung von Resolventen ein ohne die Vollständigkeit zu verlieren
- Unit-Resolution
  - Mindestens eine Elternklausel ist Unit-Klausel
    - Aus dem Resolutionspartner wird ein Literal gelöscht
  - Unit-Resolution ist widerlegungsvollständig für Hornklauseln
  - $F \vdash_{\text{Unitres}} \square$ ist in linearer Zeit entscheidbar
  - Unit-Resolution ist nicht mehr widerlegungsvollständig
    - $\{\{a, b\}, \{a, \neg b\}, \{\neg a, b\}, \{\neg a, \neg b\}\} \models \square$

Varianten der Resolution: Ordered Resolution

- Geordnete Resolution
  - Voraussetzung: strikte, totale Ordnung $\prec$ auf Variablen
  - Literal, über das resolviert wird, muss in jeder Elternklausel maximal sein.
  - Geordnete Resolution ist widerlegungs-vollständig
    - Baue semantischen Baum mit kleinster Variable an der Wurzel, größter Variable an den Blättern
    - Falls $\mathcal{F}$ unerfüllbar gibt es einen Inferenzknoten n.
    - Die entsprechende Resolution an n ist zulässig, da sie über ein maximales Literal resolviert. (Die beteiligten Klauseln können kein größeres Literal beinhalten, da sie sonst nicht unmittelbar unterhalb n schon falsifiziert würden. Denn größere Literale werden weiter abwärts im Baum belegt und dort sind für beide möglichen Belegungen Zweige enthalten.)

Ordered Resolution: Vollständigkeit

S = {{p}, {¬p,q}, {¬r}, {¬p,¬q,r}}



Im ersten Schritt einzig zulässig: R({¬p,¬q,r},{¬r})={¬p,¬q}
Im zweiten Schritt einzig zulässig: R({¬p,¬q},{¬p,q})={¬p}
Im dritten Schritt einzig zulässig: R({¬p},{p})={}

SR

Varianten der Resolution: backward dual (BD) resolution

- Backward dual resolution (BD Resolution) arbeitet „rückwärts“ und auf Termen statt Klauseln (dual)
- Seien 2 Terme $T_1 = A \wedge x$ und $T_2 = B \wedge \neg x$ mit komplementären Literalen x und $\neg x$ gegeben. Dann ist $D = A \wedge B$ die ***BD-Resolvente*** von T_1 und T_2 und es gilt:
  - $D \vDash T_1 \vee T_2$.
  - Die BD-Resolvente von $T_1 = x$ und $T_2 = \neg x$ ist $\top$, denn $\top \vDash x \vee \neg x$.
- Beweis: Sei b eine Belegung mit $b(D)=1$.
  - falls $b(x)=1$, dann ist $b(T_1)=1$
  - falls $b(x)=0$, dann ist $b(T_2)=1$
- Satz: falls $A \vdash {}^*_{BDres} B$, dann $B \vDash A$
  - Beweis: Induktion.
- Korollar: falls $B \vdash {}^*_{BDres} \top$, dann $\vDash B$

BD-Resolution: Vollständigkeit

- BD Resolution ist tautologie-vollständig
  - Falls $\models D$, dann $D \vdash^*_{BDres} \top$
- Beweis über (duale) semantische Bäume
  - Sei D eine aussagenlogische Formel in DNF und $\models D$.
  - Bilde einen semantischen Baum. An jedem Ast des Baumes gibt es einen (höchsten) Knoten, an dem ein Term von D wahr wird. Dieser Term schließt den Ast ab.
    - Andernfalls gäbe es einen Ast (= eine Belegung), die keinen Term von D wahr macht, also D falsifiziert.
  - Entweder ist die Wurzel schon mit T markiert, fertig.
  - Oder es muss mindestens einen Knoten K mit 2 Kindern geben, die mit Termen K_1 und K_2 markiert sind. Zu diesen gibt es eine BD-Resolvente, die K oder einen Knoten oberhalb von K wahr macht.
  - Per Induktion ist nach endlich vielen Schritten die Markierung T der Wurzel erreicht.

BD-Resolution: Vollständigkeit

- BD-Resolution ist nicht vollständig.
  - Bsp: $x \wedge \neg x \vDash y$, aber $x \wedge \neg x$ ist nicht durch BD-Resolution aus y ableitbar, denn es gibt keine einzige Resolvente.
  - wohl aber gilt $\{x \vee \neg x \vee \neg y\} \vdash^*_{Res} \top$
- Def.: Ein **Implikant** einer Formel F ist ein Term D mit $D \vDash F$
- Def. (W. Quine): Ein **Prim-Implikant** P ist ein Implikant, in dem keine überflüssigen Literale enthalten sind. Streicht man in einem Prim-Implikanten P irgend ein Literal, so ist P kein Implikant mehr.
  - Jeder Implikant D ist prim oder hat einen Prim-Implikanten P als Teil-Term: $D = P \wedge Q$.
  - Die Terme einer DNF können durch Prim-Implikanten „abgedeckt“ werden
  - Bsp.: $F = A \wedge B \vee A \wedge \neg B \vee C$

    $A \wedge B$ und $A \wedge \neg B$ sind Implikanten ebenso wie $A \wedge B \wedge C$ und viele weitere

    A und C sind (die einzigen) Prim-Implikanten. A deckt $A \wedge B$ und $A \wedge \neg B$ ab.

    F kann vereinfacht werden zu $F = A \vee C$. (Denn $A \wedge B \vee A \wedge \neg B \vee C \equiv A \vee C$)

BD-Resolution: Vollständigkeit

- BD Resolution ist vollständig bezüglich Prim-Implikanten
  - Für jeden Prim-Implikanten D von F gilt $F \vdash {}^*_{BDres} D$
- Beweis über (duale) semantische Bäume
  - Sei D ein Prim-Implikant von F. Seien Var(D) die Variablen in D.
  - Bilde einen semantischen Baum S für F, in dem die Variablen in Var(D) höher stehen als die Variablen in Var(F)\Var(D). Dann gibt es genau einen Knoten K in S, an dem D wahr wird.
  - Jede Belegung, die D wahr macht, ist ein Ast, der durch den Knoten K geht. Da $D \vDash F$, gibt es auf jedem solchen Ast einen (höchsten) Knoten, an dem ein Term von DNF(F) wahr wird.
    - Im Teilbaum mit Wurzel K muss es also mindestens einen Knoten K´ mit 2 Kindern geben, die mit Termen T_1 und T_2 markiert sind. Zu diesen gibt es eine BD-Resolvente, die K´ oder einen Knoten oberhalb von K´ wahr macht.
  - Per Induktion ist nach endlich vielen Schritten die Markierung D der Wurzel K erreicht. Eine kürzere Markierung als D gibt es nicht, da D schon prim ist. (K kann nicht „übersprungen“ werden.)

DNF-Minimierung mit BD-Resolution

- Da man mit BD-Resolution alle Prim-Implikanten einer Formel in DNF bilden kann, erhält man mit BD-Resolution ein Minimierungsverfahren analog zu Quine-McCluskey.
  1. Bilde alle Prim-Implikanten von DNF(F).
  2. Wähle eine minimale Teilmenge der Prim-Implikanten so aus, dass jeder Term in DNF(F) durch ein Element der Teilmenge impliziert wird.
- Im Gegensatz zu Quine-McCluskey kann man mit einer beliebigen DNF von F starten; man benötigt keine KDNF.
- Bildung der Prim-Implikanten durch BD-Resolution in Verbindung mit Absorption, nach den Gleichungen
  - BD-Resolventenbildung: $(A \land \neg x) \lor (B \land \neg x) \equiv (A \land \neg x) \lor (B \land \neg x) \lor (A \land B)$
  - Absorption: $(A \land B) \lor A \equiv A$
  - Wiederhole: {Bilde Resolventen; Absorbiere;}

    bis keine neuen Terme mehr erzeugt werden.

DNF-Minimierung mit BD-Resolution

- Im Allgemeinen sind nicht alle Prim-Implikanten nötig, um alle Terme von DNF(F) abzudecken
  - Bsp.: Sei F = ab + a´c. Prim-Implikanten sind ab, a´c, bc. Es ist sowohl F ≡ ab + a´c als auch F ≡ bc.
- Gesucht ist eine minimale Teilmenge der Prim-Implikanten, so dass jeder Term in DNF(F) durch ein Element der Teilmenge impliziert wird.
  - dieses Problem ist NP-vollständig
- Greedy-Lösung mit Auswahl-Heuristik
  - wähle den Prim-Implikanten, der die meisten Terme von F impliziert und streiche diese weg. (siehe Bsp. oben)
  - wiederhole dies, bis keine Terme mehr übrig sind
    - Es können keine Terme übrig bleiben, denn diese wären ansonsten selbst Prim-Implikanten von F
  - Dann ist F als Disjunktion der benutzten Prim-Implikanten darstellbar.

Beispiel 1: DNF-Minimierung mit BD-Resolution

- Gegeben F = xyz + wx´ + wxy´ + wxyz´. („TI-Schreibweise“)
- Berechnung der BD-Resolventen
  - F_0: 1) xyz; 2) wx´; 3) wxy´; 4) wxyz´
  - F_1: 1,2) wyz; 1,3) wxz; 1,4) wxy; 2,3) wy´; 2,4) wyz´ 3,4) wxz´;

Beispiel 1: DNF-Minimierung mit BD-Resolution

- Gegeben F = xyz + wx´ + wxy´ + wxyz´. („TI-Schreibweise“)
- Berechnung der BD-Resolventen
  - F_0: 1) xyz; 2) wx´; 3) wxy´; 4) wxyz´
  - F_1: 1,2) wyz; 1,3) wxz; 1,4) wxy; 2,3) wy´; 2,4) wyz´ 3,4) wxz´;
    - Absorption: 1,4) absorbiert 4); 2,3) absorbiert 3); 2,4 absorbiert 4); 3,4 absorbiert 4)

Beispiel 1: DNF-Minimierung mit BD-Resolution

- Gegeben F = xyz + wx´ + wxy´ + wxyz´. („TI-Schreibweise“)
- Berechnung der BD-Resolventen
  - F_0: 1) xyz; 2) wx´; 3) wxy´; 4) wxyz´
  - F_1: 1,2) wyz; 1,3) wxz; 1,4) wxy; 2,3) wy´; 2,4) wyz´ 3,4) wxz´;
  - F_2: 1,[2,3]) wxz; 1,[2,4]) wxy; 1,[3,4]) wxy;
    2,[1,3]) wz; 2,[1,4]) wy; 2,[3,4]) wz´;
    [1,2],[2,3]) wz; [1,2],[2,4]) wy; [1,2],[3,4]) wxy;
    [1,3],[2,4]) wxy; [1,3],[3,4]) wx; [1,4],[2,3]) wx;
    [2,3],[2,4]) wz´;

Beispiel 1: DNF-Minimierung mit BD-Resolution

- Gegeben F = xyz + wx´ + wxy´ + wxyz´. („TI-Schreibweise“)
- Berechnung der BD-Resolventen
  - F_0: 1) xyz; 2) wx´; 3) wxy´; 4) wxyz´
  - F_1: 1,2) wyz; 1,3) wxz; 1,4) wxy; 2,3) wy´; 2,4) wyz´ 3,4) wxz´;
  - F_2: 1,[2,3]) wxz; 1,[2,4]) wxy; 1,[3,4]) wxy;

    2,[1,3]) wz; 2,[1,4]) wy; 2,[3,4]) wz´;

    [1,2],[2,3]) wz; [1,2],[2,4]) wy; [1,2],[3,4]) wxy;

    [1,3],[2,4]) wxy; [1,3],[3,4]) wx; [1,4],[2,3]) wx;

    [2,3],[2,4]) wz´;

    - Absorption: 1,3) absorbiert 1,[2,3]);

      1,4) absorbiert 1,[2,4]) und 1,[3,4]) und [1,2],[3,4]) und [1,3],[2,4])

      2,[1,3]) wz absorbiert 1,2) und 1,3) und [1,2],[2,3]);

      2,[1,4]) wy absorbiert 1,4) und 2,4) und [1,2],[2,4]); 2,[3,4]) wz´ absorbiert [2,3],[2,4]);

      [1,3],[3,4]) wx absorbiert 3,4) und [1,4],[2,3])

Beispiel 1: DNF-Minimierung mit BD-Resolution

- Gegeben F = xyz + wx´ + wxy´ + wxyz´. („TI-Schreibweise“)
- Berechnung der BD-Resolventen
  - F_0: 1) xyz; 2) wx´; 3) wxy´; 4) wxyz´
  - F_1: 1,2) wyz; 1,3) wxz; 1,4) wxy; 2,3) wy´; 2,4) wyz´ 3,4) wxz´;
  - F_2: 1,[2,3]) wxz; 1,[2,4]) wxy; 1,[3,4]) wxy;
    2,[1,3]) wz; 2,[1,4]) wy; 2,[3,4]) wz´;
    [1,2],[2,3]) wz; [1,2],[2,4]) wy; [1,2],[3,4]) wxy;
    [1,3],[2,4]) wxy; [1,3],[3,4]) wx; [1,4],[2,3]) wx;
    [2,3],[2,4]) wz´;
  - F_3: 1,[2,[3,4]]) wxz; 2,[[1,3],[3,4]]) w; ….

Beispiel 1: DNF-Minimierung mit BD-Resolution

- Gegeben F = xyz + wx´ + wxy´ + wxyz´. („TI-Schreibweise“)
- Berechnung der BD-Resolventen
  - F_0: 1) xyz; 2) wx´; 3) wxy´; 4) wxyz´
  - F_1: 1,2) wyz; 1,3) wxz; 1,4) wxy; 2,3) wy´; 2,4) wyz´ 3,4) wxz´;
  - F_2: 1,[2,3]) wxz; 1,[2,4]) wxy; 1,[3,4]) wxy;
    2,[1,3]) wz; 2,[1,4]) wy; 2,[3,4]) wz´;
    [1,2],[2,3]) wz; [1,2],[2,4]) wy; [1,2],[3,4]) wxy;
    [1,3],[2,4]) wxy; [1,3],[3,4]) wx; [1,4],[2,3]) wx;
    [2,3],[2,4]) wz´;
  - F_3: 1,[2,[3,4]]) wxz; 2,[[1,3],[3,4]]) w; ….
    - Absorption: w absorbiert jeden Term mit Variable w. Keine weitere Resolventen möglich.

Beispiel 1: DNF-Minimierung mit BD-Resolution

- Gegeben F = xyz + wx´ + wxy´ + wxyz´. („TI-Schreibweise“)
- Berechnung der BD-Resolventen
  - F_0: 1) xyz; 2) wx´; 3) wxy´; 4) wxyz´
  - F_1: 1,2) wyz; 1,3) wxz; 1,4) wxy; 2,3) wy´; 2,4) wyz´ 3,4) wxz´;
  - F_2: 1,[2,3]) wxz; 1,[2,4]) wxy; 1,[3,4]) wxy;
    2,[1,3]) wz; 2,[1,4]) wy; 2,[3,4]) wz´;
    [1,2],[2,3]) wz; [1,2],[2,4]) wy; [1,2],[3,4]) wxy;
    [1,3],[2,4]) wxy; [1,3],[3,4]) wx; [1,4],[2,3]) wx;
    [2,3],[2,4]) wz´;
  - F_3: 1,[2,[3,4]]) wxz; 2,[[1,3],[3,4]]) w; ….
    - Absorption: w absorbiert jeden Term mit Variable w. Keine weitere Resolventen möglich.
- Prim-Implikanten sind xyz und w. In diesem Fall werden beide gebraucht, um alle Terme von F abzudecken.
- Insgesamt ist F ≡ xyz + w.

Beispiel 2: DNF-Minimierung mit ordered BD-Resolution

- Gegeben F = xyz + wx´ + wxy´ + wxyz´. („TI-Schreibweise“)
- Berechnung der geordneten BD-Resolventen mit w<x<y<z
  - F_0: 1) xyz; 2) wx´; 3) wxy´; 4) wxyz´
  - F_1: 1,2) wyz; 1,3) wxz; 1,4) wxy; 2,3) wy´; 2,4) wyz´ 3,4) wxz´;

Beispiel 2: DNF-Minimierung mit ordered BD-Resolution

- Gegeben F = xyz + wx´ + wxy´ + wxyz´. („TI-Schreibweise“)
- Berechnung der geordneten BD-Resolventen mit w<x<y<z
  - F_0: 1) xyz; 2) wx´; 3) wxy´; 4) wxyz´
  - F_1: 1,2) wyz; 1,3) wxz; 1,4) wxy; 2,3) wy´; 2,4) wyz´ 3,4) wxz´;
    - Absorption: 1,4) absorbiert 4);

Beispiel 2: DNF-Minimierung mit ordered BD-Resolution

- Gegeben F = xyz + wx´ + wxy´ + wxyz´. („TI-Schreibweise“)
- Berechnung der geordneten BD-Resolventen mit w<x<y<z
  - F_0: 1) xyz; 2) wx´; 3) wxy´; 4) wxyz´
  - F_1: 1,2) wyz; 1,3) wxz; 1,4) wxy; 2,3) wy´; 2,4) wyz´ 3,4) wxz´;
  - F_2: 3,[1,4]) wx

Beispiel 2: DNF-Minimierung mit ordered BD-Resolution

- Gegeben F = xyz + wx´ + wxy´ + wxyz´. („TI-Schreibweise“)
- Berechnung der geordneten BD-Resolventen mit w<x<y<z
  - F_0: 1) xyz; 2) wx´; 3) wxy´; 4) wxyz´
  - F_1: 1,2) wyz; 1,3) wxz; 1,4) wxy; 2,3) wy´; 2,4) wyz´ 3,4) wxz´;
  - F_2: 3,[1,4]) wx
    - 3,[1,4]) absorbiert 3) und 1,4)

Beispiel 2: DNF-Minimierung mit ordered BD-Resolution

- Gegeben F = xyz + wx´ + wxy´ + wxyz´. („TI-Schreibweise“)
- Berechnung der geordneten BD-Resolventen mit w<x<y<z
  - F_0: 1) xyz; 2) wx´; 3) wxy´; 4) wxyz´
  - F_1: 1,2) wyz; 1,3) wxz; 1,4) wxy; 2,3) wy´; 2,4) wyz´ 3,4) wxz´;
  - F_2: 3,[1,4]) wx
  - F_3: 2,[3,[1,4]]) w

Beispiel 2: DNF-Minimierung mit ordered BD-Resolution

- Gegeben F = xyz + wx´ + wxy´ + wxyz´. („TI-Schreibweise")
- Berechnung der geordneten BD-Resolventen mit w<x<y<z
  - F_0: 1) xyz; 2) wx´; 3) wxy´; 4) wxyz´
  - F_1: 1,2) wyz; 1,3) wxz; 1,4) wxy; 2,3) wy´; 2,4) wyz´ 3,4) wxz´;
  - F_2: 3,[1,4]) wx
  - F_3: 2,[3,[1,4]]) w
    - 2,[3,[1,4]]) w absorbiert 2) und 3,[1,4])

Beispiel 2: DNF-Minimierung mit ordered BD-Resolution

- Gegeben F = xyz + wx´ + wxy´ + wxyz´. („TI-Schreibweise“)
- Berechnung der geordneten BD-Resolventen mit w<x<y<z
  - F_0: 1) xyz; 2) wx´; 3) wxy´; 4) wxyz´
  - F_1: 1,2) wyz; 1,3) wxz; 1,4) wxy; 2,3) wy´; 2,4) wyz´ 3,4) wxz´;
  - F_2: 3,[1,4]) wx
  - F_3: 2,[3,[1,4]]) w
- w ist offensichtlich Prim-Implikant.
- Geordnete BD-Resolution ist nicht vollständig bezgl. Prim-Implikanten-Bildung
  - Bsp.: Sei F = ab + a´c, sei a<b<c. Prim-Implikanten sind ab, a´c, bc; aber bc ist nicht durch geordnete Resolution herleitbar.
- Da aber in Bsp.2 auch keine allgemeinen BD-Resolventen mehr möglich sind ist auch xyz ein Prim-Implikant
- Insgesamt ist F ≡ xyz + w.

Bsp 3 zu BD-Resolution: ***Vollständigkeit eines Regelsystems***

- Expertensysteme sind regelbasierte Systeme (RBS) zur Entscheidungsfindung
- Boolesche RBS haben die Form $\{<B_i \rightarrow A_i>\}$
  - die B_i sind aussgenlog. Bedingungen, die A_i sind irgendwelche Aktionen
- Grundfrage: deckt das RBS den Entscheidungsraum vollständig ab? (Wird jedes Problem durch das RBS gelöst?)
- Beispiel: Finden der Teile zu einem bestellten Fahrzeug durch eine regelbasierte Stückliste
  - zu jeder geometrischen Einbauposition im Fahrzeug wird eine Regelmenge definiert, die die alternativen Teile (z.B. Radios) und deren Einbaubedingungen beschreibt (z.B.: falls Großbritannien, dann Digitalradio)
  - Jede Regel hat die Form <Einbaubedingung → Teil>
  - Die Frage ist: wird für jedes Fahrzeug ein Teil gefunden?

***Bsp 3 zu BD-Resolution:* Vollständigkeit eines Regelsystems**

- Wenn der Entscheidungsraum durch die Variablen $x_1,\ldots,x_n$ aufgespannt wird, dann ist die Frage:
  - Ist $x_1 \lor \ldots \lor x_n \equiv A_1 \lor \ldots \lor A_m$?
- Lösungsmöglichkeiten mit BD-Resolution
  - Beweise, dass $(x_1 \lor \ldots \lor x_n \leftrightarrow A_1 \lor \ldots \lor A_m)$ eine Tautologie ist.
  - Berechne die Prim-Implikanten von $(A_1 \lor \ldots \lor A_m)$. Dies müssen genau die Variablen $x_1,\ldots,x_n$ sein.
    - Andernfalls kann man $A=(A_1 \lor \ldots \lor A_m)$ minimieren. Diese Funktion zeigt, unter welchen Bedingungen Aktionen ausgeführt werden.
    - Die Negation $\neg A$ zeigt an, unter welchen Bedingungen keine Aktion erfolgt („Loch“ in der Entscheidungsfunktion).