Semantik der Aussagenlogik (Kurzform)

Eine **Interpretation** ist eine Abbildung der Aussagevariablen je in {*true*,*false*} (entspr. „Wahrheit“ und „Falschheit“, abk. 0,1)

Interpretation zusammengesetzter Formeln **definiere** induktiv über die Interpretationen ihrer Teilformeln.
Hier abgekürzt mit **Wahrheitstafeln** für Teilformeln α,β:

α	β	$\neg\alpha$	$\alpha \wedge \beta$	$\alpha \vee \beta$	$\alpha \Rightarrow \beta$	$\alpha \Leftrightarrow \beta$
0	0	Δ1	Δ0	Δ0	1	1
0	1	Δ1	Δ0	Δ1	1	0
1	0	Δ0	Δ0	Δ1	0	0
1	1	Δ0	Δ1	Δ1	1	1

(Das Δ erinnere: der jeweilige Wert wird hier *definiert*! Später verwenden wir Wahrheitstafeln, um Interpretationen von Formeln *berechnen*!)

Modelle und Folgerung

Ein **Modell** einer Formel (bzw. einer Formelmenge) ist eine Interpretation, unter der die Formel (bzw. alle Formeln der Menge) auf *true* abbildet.

Beispiele: Formel: $(P \wedge Q)$

$I_1 = \{P \mapsto 1, Q \mapsto 1\}$ Modell

$I_2 = \{P \mapsto 1, Q \mapsto 0\}$ kein Modell

Eine Formel β **folgt** aus einer Formel α (bzw. einer Formelmenge A), gdw. alle Modelle von α (bzw. alle Modelle aller Formeln von A) auch Modelle von β sind.
(α *entails* β, Notation α |= β)

Beispiele: $(P \wedge Q) \models P$ $\{(P \Rightarrow Q), P\} \models Q$

Mechanisches Prüfen der Folgerbarkeit

Frage: Gilt die behauptete Folgerung {(P⇒Q), P} |= Q ?

Entscheidungsverfahren: Prüfen in der Wahrheitstafel

alle Modelle von (P⇒Q),

die auch Modell von P sind,

sind auch Modelle von Q!

P	Q	$P \Rightarrow Q$
0	0	1
0	1	1
1	0	0
1	1	1

Aussagelogische Folgerbarkeit ist entscheidbar in $O(2^{|Variable|})$ Zeit

Model Checking in allen Interpretationen

```
function TT-ENTAILS?(KB, α) returns true or false

    symbols ← a list of the proposition symbols in KB and α
    return TT-CHECK-ALL(KB, α, symbols, [])

---

function TT-CHECK-ALL(KB, α, symbols, model) returns true or false
    if EMPTY?(symbols) then
        if PL-TRUE?(KB, model) then return PL-TRUE?(α, model)
        else return true
    else do
        P ← FIRST(symbols); rest ← REST(symbols)
        return TT-CHECK-ALL(KB, α, rest, EXTEND(P, true, model)) and
                    TT-CHECK-ALL(KB, α, rest, EXTEND(P, false, model))
```

Effizientere Implementierungen folgen später

Äquivalenz

Zwei Formeln α und β sind **äquivalent**, Notation: $\alpha \equiv \beta$, gdw. sie dieselben Modelle haben.

$\alpha \wedge \beta$	$\equiv$	$\beta \wedge \alpha$	Kommutativität (auch für $\vee$)
$(\alpha \wedge \beta) \wedge \chi$	$\equiv$	$\alpha \wedge (\beta \wedge \chi)$	Assoziativität (auch für $\vee$)
$\neg(\neg\alpha)$	$\equiv$	α	Doppelte Negation
$\alpha \Rightarrow \beta$	$\equiv$	$\neg\beta \Rightarrow \neg\alpha$	Kontraposition
$\neg(\alpha \wedge \beta)$	$\equiv$	$\neg\beta \vee \neg\alpha$	deMorgansche Regel (analog für $\vee$)
$\alpha \wedge (\beta \vee \chi)$	$\equiv$	$(\alpha\wedge\beta) \vee (\alpha\wedge\chi)$	Distributivität $\wedge$ über $\vee$ (analog $\vee$, $\wedge$)

Beweis je durch Aufstellen der W'tafeln (gleiche Einträge je in allen Zeilen)

Erfüllbarkeit, Allgemeingültigkeit, Inkonsistenz

Eine Formel(menge), die mindestens 1 Modell hat, ist **erfüllbar**

Beispiele: P $(P \wedge Q)$ $(P \vee \neg P)$

Satz aus Theo.Inf.: Entscheidung der Erf'barkeit ist NP-vollständig

Eine Formel(menge), die in allen Interpretationen wahr ist, ist **(allgemein)gültig** (tautologisch)

Beispiele: $true$ $(P \vee \neg P)$

Eine Formel(menge), die kein Modell hat, ist **inkonsistent** (widersprüchlich)

Beispiele: $false$ $(P \wedge \neg P)$

Hilfreiche/Fundamentale Sätze

Deduktionssatz

Für beliebige α und β: $\alpha \models \beta$ gdw. $\alpha \Rightarrow \beta$ allgemeingültig ist

Beweis direkt aus Definitionen der Folgerung und der Implikation

Satz vom Widerspruchsbeweis

Für beliebige α und β: $\alpha \models \beta$ gdw. $\alpha \wedge \neg\beta$ inkonsistent ist

Beweis: Umformung des Deduktionstheorems

Monotoniesatz

Für beliebige KB, α und β: wenn $\mathbf{KB} \models \alpha$ dann $\mathbf{KB} \cup \beta \models \alpha$

Beweis über Betrachtung der Modelle

Endlichkeitssatz

Eine (möglicherweise unendliche) Formelmenge ist inkonsistent, gdw. sie eine endliche inkonsistente Teilmenge hat.

Beweis Richtung „$\rightarrow$“ nichttrivial! (s, z.B. Schöning)

KNF und DNF

Ein **Literal** ist eine negierte oder nicht negierte Aussagevariable.
Eine **Klausel** ist eine Disjunktion von Literalen

Beispiele: P Literal; $\neg Q$ Literal; $\neg\neg P$ kein Literal; $(\neg Q \vee P)$ Klausel

Eine Formel ist in **konjunktiver** (bzw. **disjunktiver**) **Normalform** (**KNF**, bzw. **DNF**), wenn sie eine Konjunktion von Disjunktionen von Literalen (bzw. Disjunktion von Konjunktionen) ist.
KNF bezeichnet man auch als **Klauselnormalform**.

Beispiele: KNF: $\neg P \wedge (\neg Q \vee R) \wedge (\neg R \vee S \vee \neg P)$
DNF: $(\neg Q \wedge P \wedge P) \vee \neg P \vee (\neg P \wedge S \wedge R)$

Notation: Die leere Klausel (enthält 0 Literale) schreiben wir $\square$

KNF und DNF sind duale Formen. Im Folgenden Konzentration auf KNF!

Umformung von Formeln in KNF

… durch Verwendung der Äquivalenzen Folie 120 nach Rezept:

1. Löse $\Leftrightarrow$ und $\Rightarrow$ auf
   (s. Folie 116: $P \Leftrightarrow Q$ zu $(P \Rightarrow Q) \wedge (Q \Rightarrow P)$ und $P \Rightarrow Q$ zu $\neg P \vee Q$)
2. Bringe alle Negationszeichen direkt vor die Variablen;
   (deMorgansche Regeln)
   löse dabei doppelte Negation immer auf
3. Multipliziere aus und fasse zusammen, bis KNF fertig
   (Distributivität)

Beispiel: Überführe in KNF: $(P \wedge (Q \Rightarrow R)) \Rightarrow S$ an der Tafel

Tafelbeispiel

$$
\begin{aligned}
\big(P \wedge (Q \Rightarrow R)\big) \Rightarrow S \quad &\equiv \quad \big(P \wedge (\neg Q \vee R)\big) \Rightarrow S \\
&\equiv \quad \neg\big(P \wedge (\neg Q \vee R)\big) \vee S \\
&\equiv \quad \big(\neg P \vee \neg(\neg Q \vee R)\big) \vee S && \text{de M.} \\
&\equiv \quad \big(\neg P \vee (\neg\neg Q \wedge \neg R)\big) \vee S && \text{de M.} \\
&\equiv \quad \big(\neg P \vee (Q \wedge \neg R)\big) \vee S \\
&\equiv \quad \big((\neg P \vee Q) \wedge (\neg P \vee \neg R)\big) \vee S && \text{Distr.} \\
&\equiv \quad \big(S \vee (\neg P \vee Q)\big) \wedge \big(S \vee (\neg P \vee \neg R)\big) && \text{Komm.} + \text{Distr.} \\
&\equiv \quad (S \vee \neg P \vee Q) \wedge (S \vee \neg P \vee \neg R)
\end{aligned}
$$

Mehr zur KNF

Existenz

Zu jeder AL-Formel gibt es äquivalente Formeln in KNF.

Beweis Induktion über Formelaufbau nach BNF-Syntax; verwende „Rezept“

Uneindeutigkeit

Die äquivalente KNF zu einer gegebenen Formel ist uneindeutig.

Beispiele: $P \wedge Q$ und $Q \wedge P$ sind äquivalente, ungleiche KNF

P und $(P \vee R) \wedge (P \vee \neg R)$ äquivalente, ungleiche KNF

Minimierung

Es gibt effiziente Verfahren zur Minimierung einer gegebenen KNF.

Mengennotation („Klauselmenge“)

Im Folgenden werden wir Formeln in KNF wie folgt notieren:

aus $(L_{1,1} \vee \ldots \vee L_{1,n1}) \wedge \ldots \wedge (L_{m,1} \vee \ldots \vee L_{m,nm})$

wird $\{\{L_{1,1}, \ldots, L_{1,n1}\}, \ldots, \{L_{m,1}, \ldots, L_{m,nm}\}\}$

Nutzen der KNF

- Normalformen erlauben effizientere Verfahren
  - Beispiel Davis/Putnam (gleich)
  - Beispiel Resolution (später)
- Lokale Entscheidung der Erfüllbarkeit:
  - Ein einziges Exemplar von $\square$ signalisiert Inkonsistenz;
  - ist eine Interpretation lokal Modell jeder einzelnen Klausel, ist sie Modell

Abkürzen der Erfüllbarkeitsprüfung

1. Entferne Tautologien (Klauseln, in denen P und $\neg P$ vorkommen)
2. Terminiere sofort, wenn jede Klausel ein *true* bewertetes Literal enthält (erf'bar) oder wenn eine Klausel endgültig *false* bewertet (inkonsistent);
3. Bewerte Variablen, die als Literal *pur* vorkommen, so, dass das Literal wahr ist
   (*pur*: kommt überall nur negiert oder nur unnegiert vor)
4. Bewerte Variablen, die als *Einsklauseln* vorkommen, so, dass das Einsklausel-Literal wahr ist
   (*Einsklausel*: Klausel, die aus genau einem Literal besteht)

Alle vier Regeln sind korrekt!

Erfüllbarkeitsprüfung nach Davis/Putnam

function DPLL-SATISFIABLE?(*s*) **returns** *true* or *false*

inputs: *s*, a sentence in propositional logic
clauses ← the clause form of *s* with tautologies deleted
symbols ← the list of proposition symbols in *s*
return DPLL (*clauses, symbols,*[])

---

function DPLL(*clauses, symbols, model*) **returns** *true* or *false*

if every clause in *clauses* is true in *model* **then return** *true*
if some clause in *clauses* is false in *model* **then return** *false*
P, value ← FIND-PURE-SYMBOL(*symbols, clauses, model*)
if *P* is non-null **then return** DPLL(*clauses, symbols–P,* EXTEND(*P, value, model*))
P, value ← FIND-UNIT-CLAUSE(*clauses, model*)
if *P* is non-null **then return** DPLL(*clauses, symbols–P,* EXTEND(*P, value, model*))
P ← FIRST(*symbols*); *rest* ← REST(*symbols*)
return DPLL(*clauses, rest,* EXTEND(*P, true, model*)) **or**
DPLL(*clauses, rest,* EXTEND(*P, false, model*))

Beispiel Davis/Putnam

Prüfe auf Erf'barkeit: $\{P,\neg Q\}, \{\neg P, Q\}, \{Q,\neg R\}, \{\neg Q,R\}$

$true, \{Q\}, \{Q,\neg R\}, \{\neg Q,R\}$ $\qquad$ $\{\neg Q\}, true, \{Q,\neg R\}, \{\neg Q,R\}$

$\{P \mapsto 1, Q \mapsto 1\}$ | **Einsklausel**

$true, true, true, \{R\}$

$\{P \mapsto 1, Q \mapsto 1, R \mapsto 1\}$ | **pur**

$true, true, true, true$ ➥ erfüllbar!

Model Checking durch lokale Suche: WALKSAT

function WALKSAT (*clauses*,*p*,*max_flips*) **returns** a model or failure

inputs: *clauses*, a propositional clause set
p, the probability of taking random assignment flips
max_flips, the number of flips allowed before giving up

model ← a random assignment true/false to the symbols in *clauses*

for *i*=1 **to** *max_flips* **do**
  if *model* satisfies *clauses* **then return** *model*
  clause ← a randomly chosen clause from *clauses* that is false in *model*
  with probability *p* flip the value in *model* of a randomly selected variable from *clause*
  else flip the value in *model* of whichever variable in *clauses* that maximizes the number of satisfied clauses

return failure

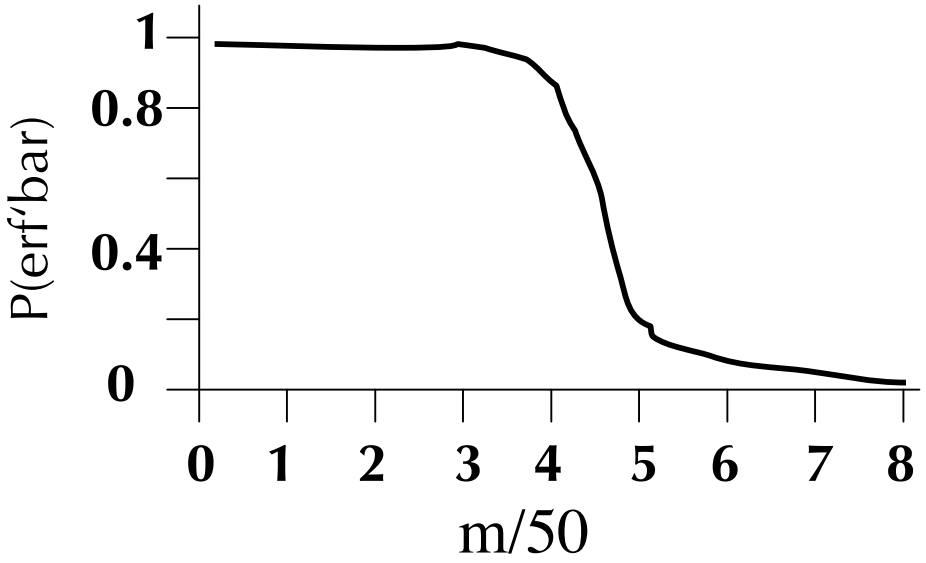
Laufzeit und „Modelldichte“

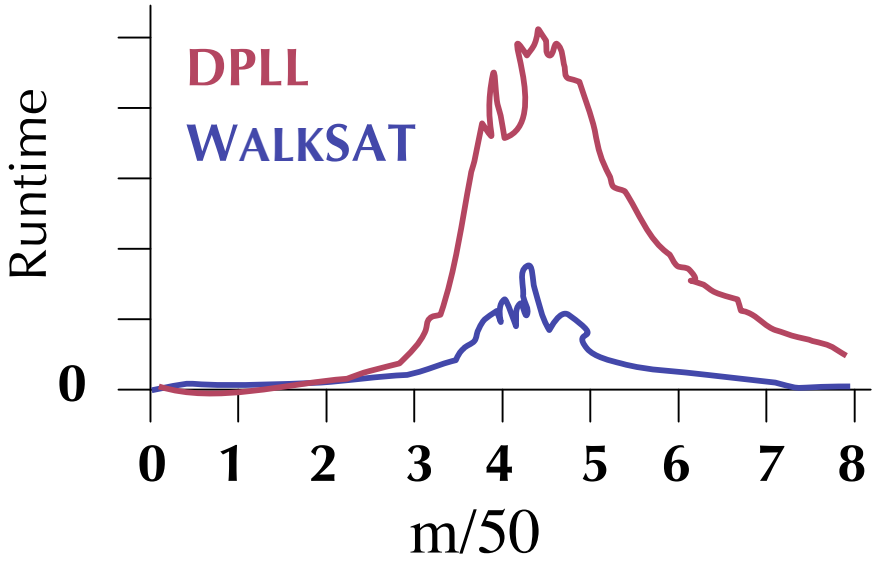
Laufzeit bis Finden eines Modells hängt beim *model checking* ab von „Dichte“ der Modelle unter allen Interpretationen

Damit korrelierter Parameter:

$$\frac{|\text{Klauseln}|}{|\text{Aus.-Variablen}|} = \frac{m}{n}$$

(viele Klauseln ➥ inkonsistent;
wenige Klauseln ➥ erfüllbar)





Fazit Erfüllbarkeitsprüfung

- Erfüllbarkeit kann in der Aussagenlogik konstruktiv geprüft werden durch den Versuch, ein Modell zu erstellen
- Wie in der Suche und in CSPs gibt es dazu systematische und lokale Verfahren
- Für endliche Modellmengen sind Verfahren zur Erfüllbarkeitsprüfung Stand der Technik
  (mit repräsentationen durch binäre Entscheidungsdiagramme (binary decision diagrams, BDDs) kann man Zustandsmengen bis in die Ordnung von 10^{200} Zuständen analysieren)
- Für unendliche Modellmengen oder ausdrucksstärkere Logiken braucht man allgemeinere Kalküle