

# Grundlagen der Künstlichen Intelligenz

## 32. Aussagenlogik: Lokale Suche und Ausblick

Malte Helmert

Universität Basel

4. Mai 2015

# Grundlagen der Künstlichen Intelligenz

4. Mai 2015 — 32. Aussagenlogik: Lokale Suche und Ausblick

32.1 Lokale Suche: GSAT

32.2 Lokale Suche: Walksat

32.3 Wie schwierig ist SAT?

32.4 Ausblick

32.5 Zusammenfassung

## Aussagenlogik: Überblick

Kapitelüberblick Aussagenlogik:

- ▶ 29. Grundlagen
- ▶ 30. Logisches Schliessen und Resolution
- ▶ 31. DPLL-Algorithmus
- ▶ 32. Lokale Suche und Ausblick

## 32.1 Lokale Suche: GSAT

## Lokale Suche für SAT

- ▶ Neben systematischen gibt es auch erfolgreiche **lokale Suchverfahren** für SAT.
- ▶ Diese sind im Normalfall nicht vollständig und können insbesondere nicht die **Unerfüllbarkeit** einer Formel zeigen.
- ▶ Oft ist dies aber verschmerzbar, wenn man dafür für schwierigere Probleme erfüllende Belegungen finden kann.
- ▶ Insgesamt waren DPLL-basierte systematische Verfahren allerdings in den letzten Jahren erfolgreicher.

## Lokale Suche für SAT: Ideen

Lokale Suchverfahren sind für SAT direkt anwendbar:

- ▶ **Zustände:** (vollständige) Belegungen
- ▶ **Zielzustände:** erfüllende Belegungen
- ▶ **Suchnachbarschaft:** ändere Belegung **einer** Variable
- ▶ **Heuristiken:** je nach Algorithmus; z.B. Anzahl unerfüllter Klauseln

## GSAT (Greedy SAT): Pseudo-Code

Hilfsfunktionen:

- ▶ **violated( $\Delta, I$ ):** Anzahl Klauseln in  $\Delta$ , die  $I$  nicht erfüllt
- ▶ **flip( $I, v$ ):** Die Belegung, die aus  $I$  entsteht, wenn man die Belegung der Aussagevariable  $v$  ändert

**function** GSAT( $\Delta$ ):

**repeat** *max-tries* **times:**

$I :=$  a random truth assignment

**repeat** *max-flips* **times:**

**if**  $I \models \Delta$ :

**return**  $I$

$V_{\text{greedy}} :=$  the set of variables  $v$  occurring in  $\Delta$   
for which  $\text{violated}(\Delta, \text{flip}(I, v))$  is minimal

randomly select  $v \in V_{\text{greedy}}$

$I := \text{flip}(I, v)$

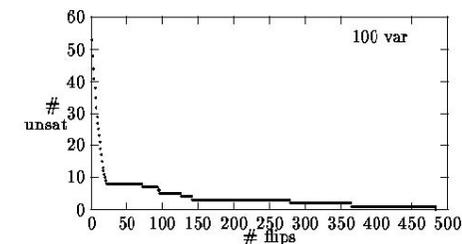
**return no solution found**

## GSAT: Diskussion

GSAT hat übliche Merkmale von lokalen Suchverfahren:

- ▶ Hill-Climbing
- ▶ Zufall (allerdings **relativ wenig!**)
- ▶ Neustarts

empirisch wird viel Zeit auf Plateaus verbracht:



## 32.2 Lokale Suche: Walksat

### Walksat: Pseudo-Code

$\text{lost}(\Delta, I, v)$ : #Klauseln in  $\Delta$ , die  $I$  erfüllt, aber  $\text{flip}(I, v)$  nicht

**function** Walksat( $\Delta$ ):

**repeat** *max-tries* **times**:

$I$  := a random truth assignment

**repeat** *max-flips* **times**:

**if**  $I \models \Delta$ :

**return**  $I$

$C$  := randomly chosen unsatisfied clause in  $\Delta$

**if** there is a variable  $v$  in  $C$  with  $\text{lost}(\Delta, I, v) = 0$ :

$V_{\text{choices}}$  := all such variables

**else** with probability  $p_{\text{noise}}$ :

$V_{\text{choices}}$  := all variables occurring in  $C$

**else**:

$V_{\text{choices}}$  := variables  $v$  in  $C$  that minimize  $\text{lost}(\Delta, I, v)$

randomly select  $v \in V_{\text{choices}}$

$I$  :=  $\text{flip}(I, v)$

**return no solution found**

### Walksat vs. GSAT

Vergleich GSAT vs. Walksat:

- ▶ sehr viel mehr Zufall in Walksat durch zufällige Wahl der betrachteten Klausel
  - ▶ auch „unintuitive“ Schritte, die die Zahl der verletzten Klauseln erst mal erhöhen, sind bei Walksat meistens möglich
- ↔ geringere Gefahr, in lokalen Minima stecken zu bleiben

## 32.3 Wie schwierig ist SAT?

## Wie schwierig ist SAT in der Praxis?

- ▶ SAT ist NP-vollständig
- ↪ Algorithmen wie DPLL benötigen im schlechtesten Fall exponentielle Zeit
- ▶ Wie sieht es im **Durchschnitt** aus?
- ▶ hängt davon ab, über **welche Probleminstanzen** der Durchschnitt gebildet wird

## SAT: polynomielle durchschnittliche Laufzeit

### Gute Nachrichten (Goldberg 1979)

Konstruierte zufällige KNF-Formeln mit  $n$  Variablen und  $k$  Klauseln wie folgt:

In jeder Klausel taucht jede Variable

- ▶ mit Wahrscheinlichkeit  $\frac{1}{3}$  positiv,
- ▶ mit Wahrscheinlichkeit  $\frac{1}{3}$  negativ,
- ▶ mit Wahrscheinlichkeit  $\frac{1}{3}$  gar nicht auf.

Dann ist die Laufzeit von DPLL im durchschnittlichen Fall polynomiell in  $n$  und  $k$ .

↪ leider kein sehr realistisches Modell für praktisch interessante KNF-Formeln (fast alle Zufallsformeln erfüllbar)

## Phasenübergänge

Wie finden wir **interessante** zufällige Probleme?

Vermutung von Cheeseman et al.:

Cheeseman et al., IJCAI 1991

Alle NP-vollständigen Probleme haben mindestens einen **Größenparameter**, für den die schwierigen Probleminstanzen in der Nähe eines **kritischen Werts** für diesen Parameter liegen.

Dieser so genannte **Phasenübergang** trennt zwei Problemregionen, z. B. eine zu stark eingeschränkte (**over-constrained**) von einer zu schwach eingeschränkten (**under-constrained**).

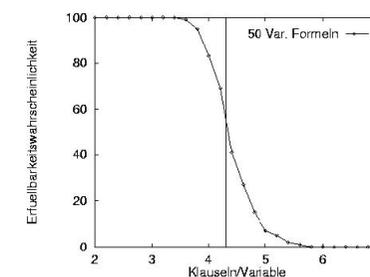
↪ bestätigt z. B. für Graphfärbung, Hamilton-Pfade und **SAT**

## Phasenübergänge für 3-SAT

Problemmodell von Mitchell et al., AAAI 1992

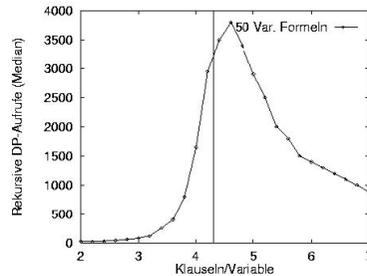
- ▶ feste Klausellänge 3
- ▶ wähle in jeder Klausel die Variablen zufällig
- ▶ Literale sind mit Wahrscheinlichkeit  $\frac{1}{2}$  positiv bzw. negativ

**kritischer Parameter:** Anz. Klauseln geteilt durch Anz. Variablen  
**Phasenübergang** bei Verhältnis von ca. 4.3



## Phasenübergang bei DPLL

DPLL zeigt hohe Laufzeit in der Nähe des Phasenübergangs:



## Phasenübergang: intuitive Erklärung

- ▶ Wenn es **sehr viele** Klauseln gibt, das Problem daher mit hoher Wahrscheinlichkeit unlösbar ist, wird das schnell durch Unit-Propagation nachgewiesen.
- ▶ Wenn es **sehr wenige** Klauseln gibt, gibt es sehr viele erfüllende Belegungen, und es ist leicht, eine zu finden.
- ▶ Nahe des **Phasenübergangs** gibt es viele „Fast-Lösungen“, die vom Suchalgorithmen verfolgt werden müssen.

## 32.4 Ausblick

## Stand der Wissenschaft

- ▶ SAT-Forschung allgemein:  
↪ <http://www.satlive.org/>
- ▶ SAT-Konferenzen seit 1996; seit 2000 jedes Jahr  
↪ <http://www.satisfiability.org/>
- ▶ Wettbewerbe für SAT-Algorithmen seit 1992  
↪ <http://www.satcompetition.org/>
  - ▶ grösste Instanzen haben mehr als 1'000'000 Literale
  - ▶ verschiedene Disziplinen (z. B. SAT vs. SAT+UNSAT; industrielle vs. zufällige Instanzen)

## Weiterführende Themen

### DPLL-basierte SAT-Algorithmen:

- ▶ effiziente Implementierungstechniken
- ▶ gute Variablenordnungen
- ▶ clause learning

### lokale Suchalgorithmen:

- ▶ effiziente Implementierungstechniken
- ▶ adaptive Suchverfahren („schwierige“ Klauseln werden mit der Zeit erkannt und priorisiert)

## 32.5 Zusammenfassung

## Zusammenfassung (1)

- ▶ **Lokale Suche** für SAT sucht im Raum der Interpretationen; Nachbarn: Belegungen, die nur in einer Variable anders sind
- ▶ haben typische Eigenschaften lokaler Suchverfahren: Bewertungsfunktionen, Randomisierung, Neustarts
- ▶ Beispiel: **GSAT** (Greedy SAT)
  - ▶ Hill-Climbing mit Heuristikfunktion: #unerfüllte Klauseln
  - ▶ Randomisierung durch Tie-Breaking und Neustarts
- ▶ Beispiel: **Walksat**
  - ▶ fokussiert in jeder Iteration auf **eine zufällig ausgewählte unerfüllte Klausel**
  - ▶ folgt nicht immer der Heuristik, sondern **injiziert Rauschen**
  - ▶ dadurch **mehr Randomisierung** als GSAT und weniger Gefahr, in lokalen Minima zu bleiben

## Zusammenfassung (2)

- ▶ **genauere Analyse** von SAT zeigt: das Problem ist NP-vollständig, aber nicht alle Instanzen sind schwer
- ▶ zufällig erzeugte 3SAT-Instanzen sind leicht zu erfüllen, wenn sie sehr wenige Klauseln beinhalten und leicht als unerfüllbar zu zeigen, wenn sie sehr viele Klauseln beinhalten
- ▶ dazwischen scharfer **Phasenübergang**