Counting the Number of Satisfying Assignments

Ferienakademie im Sarntal — Kurs 1
Moderne Suchmethoden der Informatik: Trends und Potenzial

Stefan Ploner

Department Informatik
FAU Erlangen-Nürnberg

29. September 2014

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG

Technische Universität München

Universität Stuttgart

Outline

1. Einführung
   - Was ist $\#k$-SAT?
2. Monte-Carlo Counting
   - MC im Allgemeinen
   - MC für $\#k$-SAT
3. ℓ-Cuts
   - Eliminationsbäume
   - ℓ-Cuts
4. Eliminationsbaum zurechtstutzen
   - Verbesserung der Laufzeit zum Finden von ℓ-Cuts
5. Unabhängige Klauseln und Rekursion
   - Wie man das Universum verkleinert...
   - Rekursion und Berechnung von ψ
6. Zusammenfassung

1 Einführung
Was ist $\#k$-SAT?
2 Monte-Carlo Counting
MC im Allgemeinen
MC für $\#k$-SAT
3 ℓ-Cuts
Eliminationsbäume
ℓ-Cuts
4 Eliminationsbaum zurechtstutzen
Verbesserung der Laufzeit zum Finden von ℓ-Cuts
5 Unabhängige Klauseln und Rekursion
Wie man das Universum verkleinert...
Rekursion und Berechnung von ψ
6 Zusammenfassung

Satisfiability (SAT)

$$(x_1 \vee x_2) \wedge (x_1 \vee x_3 \vee \overline{x}_4) \wedge (x_3)$$

- Erfüllbarkeit einer boolschen Formel Φ in konjunktiver Normalform über den Variablen $x_1, \ldots, x_n$

- Konjunktive Normalform (KNF / CNF):

Literale:	$l = \{x_i, \overline{x}_i \mid i \in \{1, \ldots, n\}\}$	$\overline{x}_1$
Klauseln:	$C_j = l_1 \vee \ldots \vee l_{k_j}$	$(x_1 \vee \overline{x}_2)$
KNF:	$\Phi = C_1 \wedge \ldots \wedge C_m$	$(x_1 \vee x_2) \wedge (\overline{x}_3 \vee x_2)$

- SAT ist NP-vollständig
- Brute-Force-Methode: Alle Variablenbelegungen testen
  $\rightarrow$ Laufzeit in $\mathcal{O}^*(2^n)$ $\mathcal{O}^*(.)$ ignoriert polynomielle Faktoren

k-SAT

Bsp. 2-SAT: $(x_1 \vee x_2) \wedge (\overline{x}_1 \vee x_3) \wedge (x_3) \wedge (x_3 \vee \overline{x}_4)$

- Jede Klausel enthält maximal k Literale

- 2-SAT $\in \mathcal{P}$
- $k \geq 3$: k-SAT $\in \mathcal{NP}$

- Laufzeiten randomisierter k-SAT Solver

	Laufzeit	
$k = 3$	1.30704^n	
$k = 4$	1.46899^n	

k-SAT

Bsp. 2-SAT: $(x_1 \vee x_2) \wedge (\overline{x}_1 \vee x_3) \wedge (x_3) \wedge (x_3 \vee \overline{x}_4)$

- Jede Klausel enthält maximal k Literale

- 2-SAT $\in \mathcal{P}$
- $k \geq 3$: k-SAT $\in \mathcal{NP}$

- Laufzeiten randomisierter k-SAT Solver

	Laufzeit	β_k
$k = 3$	$1.30704^n = 2^{0.3864n}$	0.3864
$k = 4$	$1.46899^n = 2^{0.5548n}$	0.5548

#k-SAT

- Gesucht ist die Anzahl erfüllender Belegungen einer k-SAT Instanz
- Brute-Force-Methode: Belegungen durchprobieren und mitzählen
  $\rightarrow$ Laufzeit wieder $\mathcal{O}^*(2^n)$

#k-SAT

- Gesucht ist die Anzahl erfüllender Belegungen einer k-SAT Instanz
- Brute-Force-Methode: Belegungen durchprobieren und mitzählen
  $\rightarrow$ Laufzeit wieder $\mathcal{O}^*(2^n)$

- #k-SAT ist für $k \geq 2$ #P-vollständig,
  die Lösung ist nicht in polynomieller Zeit verifizierbar
- #2-SAT ist exakt in $\mathcal{O}^*(1.2377^n)$ berechenbar
  ("Wahlström's Algorithmus")

Randomisiertes Zähl-Approximationsschema

- Liefert mit einer bestimmten Wahrscheinlichkeit eine Näherungslösung

$$Pr[|\#(I) - A(I, \varepsilon)| \leq \varepsilon \cdot \#(I)] \geq \frac{3}{4}$$

- I: Instanz des Zählproblems
- $\#(I)$: exakte Lösung
- $A(I, \varepsilon)$: berechnete Lösung
- ε: obere Schranke der Abweichung

Ziel des Vortrags

Algorithmus zum effizienten Lösen von #k-SAT

Eigenschaften:

- Randomisiert
- Hybrid bestehend aus 3 Komponenten:

1. Monte-Carlo Counting
2. Eliminationsbäume / ℓ-Cuts
3. Rekursion

- Kein Algorithmus kann alle Probleminstanzen gut lösen
- Nimm den Algorithmus, der für die Instanz am besten geeignet ist

1 Einführung
Was ist $\#k$-SAT?

2 Monte-Carlo Counting
MC im Allgemeinen
MC für $\#k$-SAT

3 ℓ-Cuts
Eliminationsbäume
ℓ-Cuts

4 Eliminationsbaum zurechtstutzen
Verbesserung der Laufzeit zum Finden von ℓ-Cuts

5 Unabhängige Klauseln und Rekursion
Wie man das Universum verkleinert...
Rekursion und Berechnung von ψ

6 Zusammenfassung

Monte-Carlo Counting, Beispiel

1. Rate x und y gleichverteilt im Intervall $[-1, 1]$
2. Prüfe, ob (x, y) im Kreis um den Ursprung mit Radius 1 liegt

Monte-Carlo Counting, Beispiel

1. Rate x und y gleichverteilt im Intervall $[-1, 1]$
2. Prüfe, ob (x, y) im Kreis um den Ursprung mit Radius 1 liegt
3. Wiederhole die vorigen Schritte endlos
4. Sei R der Anteil an Treffern. Die Ausgabe ist $R \cdot 4$.

Wie lautet das Ergebnis?

Monte-Carlo Counting, Beispiel

1. Rate x und y gleichverteilt im Intervall $[-1, 1]$
2. Prüfe, ob (x, y) im Kreis um den Ursprung mit Radius 1 liegt
3. Wiederhole die vorigen Schritte endlos
4. Sei R der Anteil an Treffern. Die Ausgabe ist $R \cdot 4$.

Wie lautet das Ergebnis? $\Pr[\text{Trefferwskt.}] \cdot 4 = \frac{1^2\pi}{2 \cdot 2} \cdot 4 = \pi$

Realität: Mehr Wiederholungen führen zu einer genaueren Annäherung.

Monte-Carlo Counting, Allgemein

$\mathcal{S}_I$: Menge zulässiger Lösungen
$\mathcal{U}_I$: Stichprobenraum, "Universum", $\mathcal{U}_I \supset \mathcal{S}_I$

Monte-Carlo Counting, Allgemein

$\mathcal{S}_I$: Menge zulässiger Lösungen (Kreis)
$\mathcal{U}_I$: Stichprobenraum, "Universum", $\mathcal{U}_I \supset \mathcal{S}_I$ (Quadrat)
$|\mathcal{U}_I|$ muss bekannt sein, $|\mathcal{S}_I| = \#(I)$ ist gesucht.

Monte-Carlo Counting, Allgemein

$\mathcal{S}_I$: Menge zulässiger Lösungen (Kreis)
$\mathcal{U}_I$: Stichprobenraum, "Universum", $\mathcal{U}_I \supset \mathcal{S}_I$ (Quadrat)
$|\mathcal{U}_I|$ muss bekannt sein, $|\mathcal{S}_I| = \#(I)$ ist gesucht.

Algorithmus MC:

- Wiederhole T mal
  - Rate unabhängig und gleichverteilt ein Element x aus $\mathcal{U}_I$
  - Prüfe, ob $x \in \mathcal{S}_I$ (Treffer?)
- Sei L die Anzahl der Treffer. Dann ist $R = \frac{L}{T}$ eine Abschätzung für den *Anteil* richtiger Lösungen. (Trefferwskt.)
- Dann ist $R \cdot |\mathcal{U}_I|$ eine Abschätzung von $\#(I)$

Estimator Theorem

Satz: Sei $\varepsilon > 0$. Mit $T \geq \frac{4}{\varepsilon^2}(\frac{|\mathcal{U}_I|}{|\mathcal{S}_I|} - 1)$ ist MC ein RASC, es gilt also:

$$\Pr[|MC(T(\varepsilon)) - \#(I)| \leq \varepsilon \cdot \#(I)] \geq \frac{3}{4}$$

Tschebyscheff-Ungleichung:

$$\Pr\left[|Z - \mathrm{E}[Z]| \leq \varepsilon \cdot \mathrm{E}[Z]\right] \geq 1 - \frac{1}{\varepsilon^2} \cdot \frac{\mathrm{Var}[Z]}{\mathrm{E}[Z]^2}$$

Estimator Theorem

Satz: Sei $\varepsilon > 0$. Mit $T \geq \frac{4}{\varepsilon^2}(\frac{|\mathcal{U}_I|}{|\mathcal{S}_I|} - 1)$ ist MC ein RASC, es gilt also:

$$\Pr[|MC(T(\varepsilon)) - \#(I)| \leq \varepsilon \cdot \#(I)] \geq \frac{3}{4}$$

Tschebyscheff-Ungleichung:

$$\Pr\left[|Z - \mathrm{E}[Z]| \leq \varepsilon \cdot \mathrm{E}[Z]\right] \geq 1 - \frac{1}{\varepsilon^2} \cdot \frac{\mathrm{Var}[Z]}{\mathrm{E}[Z]^2}$$

$$\mathrm{E}[Z] = \ldots = \#(I)$$

Zufallsvariable für einen Treffer in der i. Stichprobe:

$$X_i = \begin{cases} 1 & x_i \in \mathcal{S}_I \\ 0 & \text{sonst} \end{cases}$$

Zufallsvariable für einen Treffer in der i. Stichprobe:

$$X_i = \begin{cases} 1 & x_i \in \mathcal{S}_I \\ 0 & \text{sonst} \end{cases}$$

Rechenregeln der Varianz:

$$\begin{aligned} \text{Var}[X] &= \text{E}[X^2] - \text{E}[X]^2 \\ \text{Var}[cX] &= c^2 \cdot \text{Var}[X] \\ \text{Var}\left[\sum_{i=1}^{T} X_i\right] &= \sum_{i=1}^{T} \text{Var}[X_i], \qquad \text{wenn } X_i \text{ unabhängig} \end{aligned}$$

Zufallsvariable für einen Treffer in der i. Stichprobe:

$$X_i = \begin{cases} 1 & x_i \in \mathcal{S}_I \\ 0 & \text{sonst} \end{cases}$$

Rechenregeln der Varianz:

$$\begin{aligned} \text{Var}[X] &= \text{E}[X^2] - \text{E}[X]^2 \\ \text{Var}[cX] &= c^2 \cdot \text{Var}[X] \\ \text{Var}\left[\sum_{i=1}^{T} X_i\right] &= \sum_{i=1}^{T} \text{Var}[X_i], \qquad \text{wenn } X_i \text{ unabhängig} \end{aligned}$$

$$\text{Var}[Z] = \ldots = \frac{|\mathcal{S}_I|}{T} \cdot (|\mathcal{U}_I| - |\mathcal{S}_I|)$$

Tschebyscheff-Ungleichung:

$$\Pr\left[|Z - \mathrm{E}[Z]| \leq \varepsilon \cdot \mathrm{E}[Z]\right] \geq 1 - \frac{1}{\varepsilon^2} \cdot \frac{\mathrm{Var}[Z]}{\mathrm{E}[Z]^2}$$

Satz: Sei $\varepsilon > 0$. Mit $T \geq \frac{4}{\varepsilon^2}\left(\frac{|\mathcal{U}_I|}{|\mathcal{S}_I|} - 1\right)$ ist MC ein RASC, es gilt also:
$\Pr[|MC(T(\varepsilon)) - \#(I)| \leq \varepsilon \cdot \#(I)] \geq \frac{3}{4}$

Tschebyscheff-Ungleichung:

$$\Pr\left[|Z - \mathrm{E}[Z]| \leq \varepsilon \cdot \mathrm{E}[Z]\right] \geq 1 - \frac{1}{\varepsilon^2} \cdot \frac{\mathrm{Var}[Z]}{\mathrm{E}[Z]^2} \overset{!}{\geq} \frac{3}{4}$$

Satz: Sei $\varepsilon > 0$. Mit $T \geq \frac{4}{\varepsilon^2}\left(\frac{|\mathcal{U}_I|}{|\mathcal{S}_I|} - 1\right)$ ist MC ein RASC, es gilt also:
$\Pr[|MC(T(\varepsilon)) - \#(I)| \leq \varepsilon \cdot \#(I)] \geq \frac{3}{4}$

Tschebyscheff-Ungleichung:

$$\Pr\left[|Z - \mathrm{E}[Z]| \leq \varepsilon \cdot \mathrm{E}[Z]\right] \geq 1 - \frac{1}{\varepsilon^2} \cdot \frac{\mathrm{Var}[Z]}{\mathrm{E}[Z]^2} \overset{!}{\geq} \frac{3}{4}$$

$$\frac{1}{\varepsilon^2} \cdot \frac{\mathrm{Var}[Z]}{\mathrm{E}[Z]^2} \leq \frac{1}{4}$$

Satz: Sei $\varepsilon > 0$. Mit $T \geq \frac{4}{\varepsilon^2}\left(\frac{|\mathcal{U}_I|}{|\mathcal{S}_I|} - 1\right)$ ist MC ein RASC, es gilt also:
$\Pr[|MC(T(\varepsilon)) - \#(I)| \leq \varepsilon \cdot \#(I)] \geq \frac{3}{4}$

Tschebyscheff-Ungleichung:

$$\Pr\left[|Z - \mathrm{E}[Z]| \leq \varepsilon \cdot \mathrm{E}[Z]\right] \geq 1 - \frac{1}{\varepsilon^2} \cdot \frac{\mathrm{Var}[Z]}{\mathrm{E}[Z]^2} \stackrel{!}{\geq} \frac{3}{4}$$

$$\frac{1}{\varepsilon^2} \cdot \frac{\mathrm{Var}[Z]}{\mathrm{E}[Z]^2} \leq \frac{1}{4}$$

$$\frac{1}{\varepsilon^2} \cdot \frac{\frac{|\mathcal{S}_I|}{T} \cdot (|\mathcal{U}_I| - |\mathcal{S}_I|)}{|\mathcal{S}_I|^2} \leq \frac{1}{4}$$

Satz: Sei $\varepsilon > 0$. Mit $T \geq \frac{4}{\varepsilon^2}\left(\frac{|\mathcal{U}_I|}{|\mathcal{S}_I|} - 1\right)$ ist MC ein RASC, es gilt also:
$\Pr[|MC(T(\varepsilon)) - \#(I)| \leq \varepsilon \cdot \#(I)] \geq \frac{3}{4}$

Tschebyscheff-Ungleichung:

$$\Pr\left[|Z - \mathrm{E}[Z]| \leq \varepsilon \cdot \mathrm{E}[Z]\right] \geq 1 - \frac{1}{\varepsilon^2} \cdot \frac{\mathrm{Var}[Z]}{\mathrm{E}[Z]^2} \stackrel{!}{\geq} \frac{3}{4}$$

$$\frac{1}{\varepsilon^2} \cdot \frac{\mathrm{Var}[Z]}{\mathrm{E}[Z]^2} \leq \frac{1}{4}$$

$$\frac{1}{\varepsilon^2} \cdot \frac{\frac{|\mathcal{S}_I|}{T} \cdot (|\mathcal{U}_I| - |\mathcal{S}_I|)}{|\mathcal{S}_I|^2} \leq \frac{1}{4}$$

$$\frac{4}{\varepsilon^2}\left(\frac{|\mathcal{U}_I|}{|\mathcal{S}_I|} - 1\right) \leq T$$

Satz: Sei $\varepsilon > 0$. Mit $T \geq \frac{4}{\varepsilon^2}\left(\frac{|\mathcal{U}_I|}{|\mathcal{S}_I|} - 1\right)$ ist MC ein RASC, es gilt also:
$\Pr[|MC(T(\varepsilon)) - \#(I)| \leq \varepsilon \cdot \#(I)] \geq \frac{3}{4}$

In unserem Beispiel...

$$T_I(\varepsilon) \geq \frac{4}{\varepsilon^2}\left(\frac{|\mathcal{U}_I|}{|\mathcal{S}_I|} - 1\right)$$

$$T(\varepsilon) \geq \frac{4}{\varepsilon^2}\left(\frac{2 \cdot 2}{1^2\pi} - 1\right) \cong 1.1 \cdot \varepsilon^{-2}$$

$$T(10^{-3}) \geq 1.1 \cdot 10^{(-3)\cdot(-2)} = 1.1 \cdot 10^6$$

In unserem Beispiel...

$$T_I(\varepsilon) \geq \frac{4}{\varepsilon^2}\left(\frac{|\mathcal{U}_I|}{|\mathcal{S}_I|} - 1\right)$$

$$T(\varepsilon) \geq \frac{4}{\varepsilon^2}\left(\frac{2 \cdot 2}{1^2\pi} - 1\right) \cong 1.1 \cdot \varepsilon^{-2}$$

$$T(10^{-3}) \geq 1.1 \cdot 10^{(-3)\cdot(-2)} = 1.1 \cdot 10^6$$

Problem:

In unserem Beispiel...

$$T_I(\varepsilon) \geq \frac{4}{\varepsilon^2}\left(\frac{|\mathcal{U}_I|}{|\mathcal{S}_I|} - 1\right)$$

$$T(\varepsilon) \geq \frac{4}{\varepsilon^2}\left(\frac{2 \cdot 2}{1^2\pi} - 1\right) \cong 1.1 \cdot \varepsilon^{-2}$$

$$T(10^{-3}) \geq 1.1 \cdot 10^{(-3)\cdot(-2)} = 1.1 \cdot 10^6$$

Problem: $|\mathcal{S}_I|$ ist eigentlich die Unbekannte.
Lösung: Verwende untere Grenze $\ell \leq |S_I|$.

$$T_I(\varepsilon) \geq \frac{4}{\varepsilon^2}\left(\frac{|\mathcal{U}_I|}{\ell} - 1\right) \geq \frac{4}{\varepsilon^2}\left(\frac{|\mathcal{U}_I|}{|\mathcal{S}_I|} - 1\right)$$

Monte-Carlo Counting für #k-SAT

$$I = \Phi = (\overline{x}_1 \vee x_2) \wedge (\overline{x}_2 \vee x_3)$$

- $\mathcal{S}_\Phi$:

Monte-Carlo Counting für $\#k$-SAT

$$I = \Phi = (\overline{x}_1 \vee x_2) \wedge (\overline{x}_2 \vee x_3)$$

- $\mathcal{S}_\Phi$: Menge der erfüllenden Belegungen
- $\mathcal{U}_\Phi$

Monte-Carlo Counting für #k-SAT

$$I = \Phi = (\overline{x}_1 \vee x_2) \wedge (\overline{x}_2 \vee x_3)$$

- $\mathcal{S}_\Phi$: Menge der erfüllenden Belegungen
- $\mathcal{U}_\Phi = \{0,1\}^n$: Belegungen der n Variablen von Φ, $|\mathcal{U}_\Phi| = 2^n$

Monte-Carlo Counting für #k-SAT

$$I = \Phi = (\overline{x}_1 \vee x_2) \wedge (\overline{x}_2 \vee x_3)$$

- $\mathcal{S}_\Phi$: Menge der erfüllenden Belegungen
- $\mathcal{U}_\Phi = \{0,1\}^n$: Belegungen der n Variablen von Φ, $|\mathcal{U}_\Phi| = 2^n$

- Wiederhole T mal
  - Für *jede* Variable
    - Mit Wahrscheinlichkeit $\frac{1}{2}$: $x_i := 1$, sonst $x_i := 0$
  - Werte Φ aus

$$X_t = \begin{cases} 1 & \text{erfüllende Belegung} \\ 0 & \text{sonst} \end{cases}$$

- $R = \frac{1}{T} \cdot \sum_{t=1}^{T} X_t$
- Gebe $R \cdot |\mathcal{U}_\Phi|$ aus.

Monte-Carlo Counting für #k-SAT

$$I = \Phi = (\overline{x}_1 \vee x_2) \wedge (\overline{x}_2 \vee x_3)$$

- $\mathcal{S}_\Phi$: Menge der erfüllenden Belegungen
- $\mathcal{U}_\Phi = \{0,1\}^n$: Belegungen der n Variablen von Φ, $|\mathcal{U}_\Phi| = 2^n$

- Wiederhole T mal — $\mathcal{O}((n+mk) \cdot T)$
  - Für *jede* Variable — $\mathcal{O}(n)$
    - Mit Wahrscheinlichkeit $\frac{1}{2}$: $x_i := 1$, sonst $x_i := 0$
  - Werte Φ aus — $\mathcal{O}(mk)$

$$X_t = \begin{cases} 1 & \text{erfüllende Belegung} \\ 0 & \text{sonst} \end{cases}$$

- $R = \frac{1}{T} \cdot \sum_{t=1}^{T} X_t$
- Gebe $R \cdot |\mathcal{U}_\Phi|$ aus.

Berechnung der Laufzeit

$$T_\Phi(\varepsilon) \geq \frac{4}{\varepsilon^2}\left(\frac{|\mathcal{U}_\Phi|}{\ell} - 1\right) = \frac{4}{\varepsilon^2}\left(\frac{2^n}{\ell} - 1\right)$$

Berechnung der Laufzeit

$$T_\Phi(\varepsilon) \geq \frac{4}{\varepsilon^2}\left(\frac{|\mathcal{U}_\Phi|}{\ell} - 1\right) = \frac{4}{\varepsilon^2}\left(\frac{2^n}{\ell} - 1\right)$$

$$LZ = (n + mk) \cdot \frac{4}{\varepsilon^2} \cdot \left(\frac{2^n}{\ell} - 1\right) \in \mathcal{O}^*\left(\frac{1}{\varepsilon^2} \cdot \frac{2^n}{\ell}\right)$$

Berechnung der Laufzeit

$$T_\Phi(\varepsilon) \geq \frac{4}{\varepsilon^2}\left(\frac{|\mathcal{U}_\Phi|}{\ell} - 1\right) = \frac{4}{\varepsilon^2}\left(\frac{2^n}{\ell} - 1\right)$$

$$LZ = (n + mk) \cdot \frac{4}{\varepsilon^2} \cdot \left(\frac{2^n}{\ell} - 1\right) \in \mathcal{O}^*\left(\frac{1}{\varepsilon^2} \cdot \frac{2^n}{\ell}\right)$$

Wie wählen wir ℓ?

Berechnung der Laufzeit

$$T_\Phi(\varepsilon) \geq \frac{4}{\varepsilon^2}\left(\frac{|\mathcal{U}_\Phi|}{\ell} - 1\right) = \frac{4}{\varepsilon^2}\left(\frac{2^n}{\ell} - 1\right)$$

$$LZ = (n + mk) \cdot \frac{4}{\varepsilon^2} \cdot \left(\frac{2^n}{\ell} - 1\right) \in \mathcal{O}^*\left(\frac{1}{\varepsilon^2} \cdot \frac{2^n}{\ell}\right)$$

Wie wählen wir ℓ?

Idee: Ein Löser für k-SAT liefert uns die Untergrenze $\ell = 1$ in $o(2^n)$

$$LZ \in \mathcal{O}^*\left(\frac{1}{\varepsilon^2} \cdot 2^n\right)$$

Berechnung der Laufzeit

$$T_\Phi(\varepsilon) \geq \frac{4}{\varepsilon^2}\left(\frac{|\mathcal{U}_\Phi|}{\ell} - 1\right) = \frac{4}{\varepsilon^2}\left(\frac{2^n}{\ell} - 1\right)$$

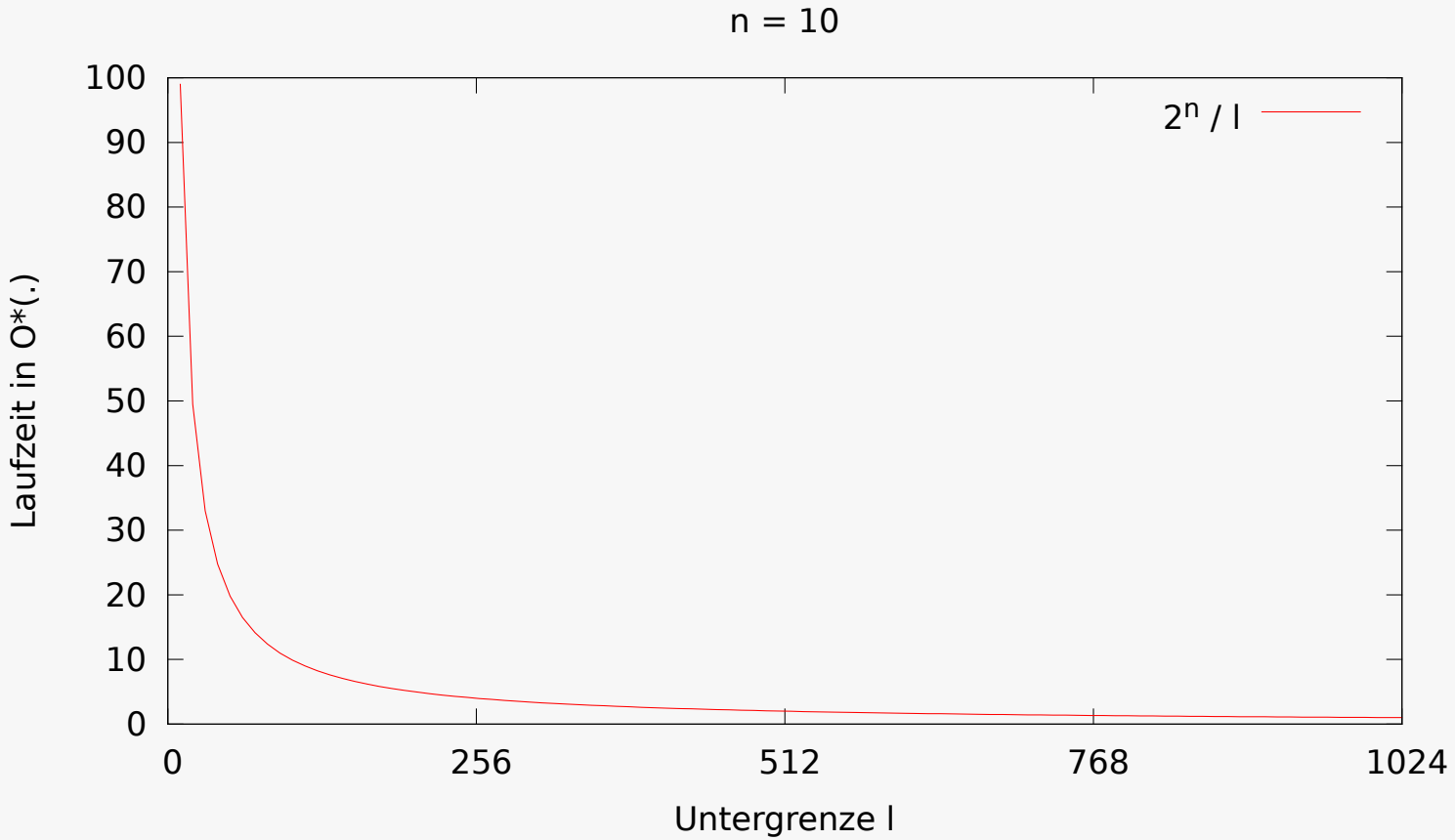
$$LZ = (n + mk) \cdot \frac{4}{\varepsilon^2} \cdot \left(\frac{2^n}{\ell} - 1\right) \in \mathcal{O}^*\left(\frac{1}{\varepsilon^2} \cdot \frac{2^n}{\ell}\right)$$

Wie wählen wir ℓ?
Idee: Ein Löser für k-SAT liefert uns die Untergrenze $\ell = 1$ in $o(2^n)$

$$LZ \in \mathcal{O}^*\left(\frac{1}{\varepsilon^2} \cdot 2^n\right)$$

Die triviale Laufzeit war auch $\mathcal{O}^*(2^n)$.
Wir sollten also noch ein wenig nachbessern...

Hat die Methode überhaupt Potenzial?



Mit einer größeren Untergrenze ℓ
ließe sich die Laufzeit deutlich reduzieren.

1 Einführung
Was ist #k-SAT?

2 Monte-Carlo Counting
MC im Allgemeinen
MC für #k-SAT

3 ℓ-Cuts
Eliminationsbäume
ℓ-Cuts

4 Eliminationsbaum zurechtstutzen
Verbesserung der Laufzeit zum Finden von ℓ-Cuts

5 Unabhängige Klauseln und Rekursion
Wie man das Universum verkleinert...
Rekursion und Berechnung von ψ

6 Zusammenfassung

Elimination Trees

- Die Lösungen einer beliebigen SAT-Instanz können als Baum dargestellt werden

Elimination Trees

- Die Lösungen einer beliebigen SAT-Instanz können als Baum dargestellt werden
- Beispiel: $(\overline{x}_1 \vee x_2) \wedge (\overline{x}_2 \vee x_3)$ (x_2, x_1, x_3)

Elimination Trees

- Die Lösungen einer beliebigen SAT-Instanz können als Baum dargestellt werden
- Beispiel: $(\overline{x}_1 \vee x_2) \wedge (\overline{x}_2 \vee x_3)$ (x_2, x_1, x_3)
- Die Anzahl erfüllbarer Blätter entspricht der gesuchten Anzahl an Lösungen $\#\Phi$.

ℓ-Cuts

- Ein ℓ-Cut ist ein Teilbaum des Eliminationsbaumes, der die Wurzel enthält und ℓ Blätter hat. Er enthält nur Knoten, die erfüllbare Formeln repräsentieren.

ℓ-Cuts

- Ein ℓ-Cut ist ein Teilbaum des Eliminationsbaumes, der die Wurzel enthält und ℓ Blätter hat. Er enthält nur Knoten, die erfüllbare Formeln repräsentieren.

ℓ-Cuts

- Ein ℓ-Cut ist ein Teilbaum des Eliminationsbaumes, der die Wurzel enthält und ℓ Blätter hat. Er enthält nur Knoten, die erfüllbare Formeln repräsentieren.
- Berechnung: Von der Wurzel des Eliminationsbaums aus überprüfen, ob die teilweise belegte Formel erfüllbar ist, bis ℓ Blätter gefunden wurden

ℓ-Cuts

- Ein ℓ-Cut ist ein Teilbaum des Eliminationsbaumes, der die Wurzel enthält und ℓ Blätter hat. Er enthält nur Knoten, die erfüllbare Formeln repräsentieren.
- Berechnung: Von der Wurzel des Eliminationsbaums aus überprüfen, ob die teilweise belegte Formel erfüllbar ist, bis ℓ Blätter gefunden wurden
- Beispiel: $(\overline{x}_1 \vee x_2) \wedge (\overline{x}_2 \vee x_3)$ (x_2, x_1, x_3)

ℓ-Cuts

- Ein ℓ-Cut ist ein Teilbaum des Eliminationsbaumes, der die Wurzel enthält und ℓ Blätter hat. Er enthält nur Knoten, die erfüllbare Formeln repräsentieren.
- Berechnung: Von der Wurzel des Eliminationsbaums aus überprüfen, ob die teilweise belegte Formel erfüllbar ist, bis ℓ Blätter gefunden wurden
- Beispiel: $(\overline{x}_1 \vee x_2) \wedge (\overline{x}_2 \vee x_3)$ (x_2, x_1, x_3)

Fakten zu ℓ-Cuts:

- Ein ℓ-Cut hat maximal $\#\Phi$ Blätter
- ℓ-Cut existiert $\Rightarrow \#\Phi \geq \ell$
- Ein ℓ-Cut enthält maximal $n \cdot \ell$ Knoten

Laufzeit zum Ermitteln eines ℓ-Cuts

ℓ-Cut, max. Anzahl Knoten:	$\ell \cdot n$
Binärbaum, Höhe:	$\log_2(\#Knoten)$
Binärbaum, Anzahl Knoten pro Level:	2^i
Laufzeit SAT-Solver:	$2^{\beta_k \cdot \#Variablen}$
n-te Partialsumme der geometrischen Reihe:	$\sum_{k=0}^{n} q^k = \frac{q^{n+1}-1}{q-1}$

Laufzeit zum Ermitteln eines ℓ-Cuts

ℓ-Cut, max. Anzahl Knoten:	$\ell \cdot n$
Binärbaum, Höhe:	$\log_2(\#Knoten)$
Binärbaum, Anzahl Knoten pro Level:	2^i
Laufzeit SAT-Solver:	$2^{\beta_k \cdot \#Variablen}$
n-te Partialsumme der geometrischen Reihe:	$\sum_{k=0}^{n} q^k = \frac{q^{n+1}-1}{q-1}$

$$T_{\text{cut}} \in \mathcal{O}^*\left(\sum_{i=0}^{\lceil \log(n\cdot\ell)\rceil} 2^i \cdot 2^{\beta_k\cdot(n-i)}\right) = \mathcal{O}^*\left(2^{\beta_k\cdot n} \cdot \sum_{i=0}^{\lceil \log(n\cdot\ell)\rceil} 2^{(1-\beta_k)\cdot i}\right)$$

$$= \mathcal{O}^*(2^{\beta_k\cdot n} \cdot \ell \underbrace{^{1-\beta_k}}_{\in[0;1)})$$

Laufzeit zum Ermitteln eines ℓ-Cuts

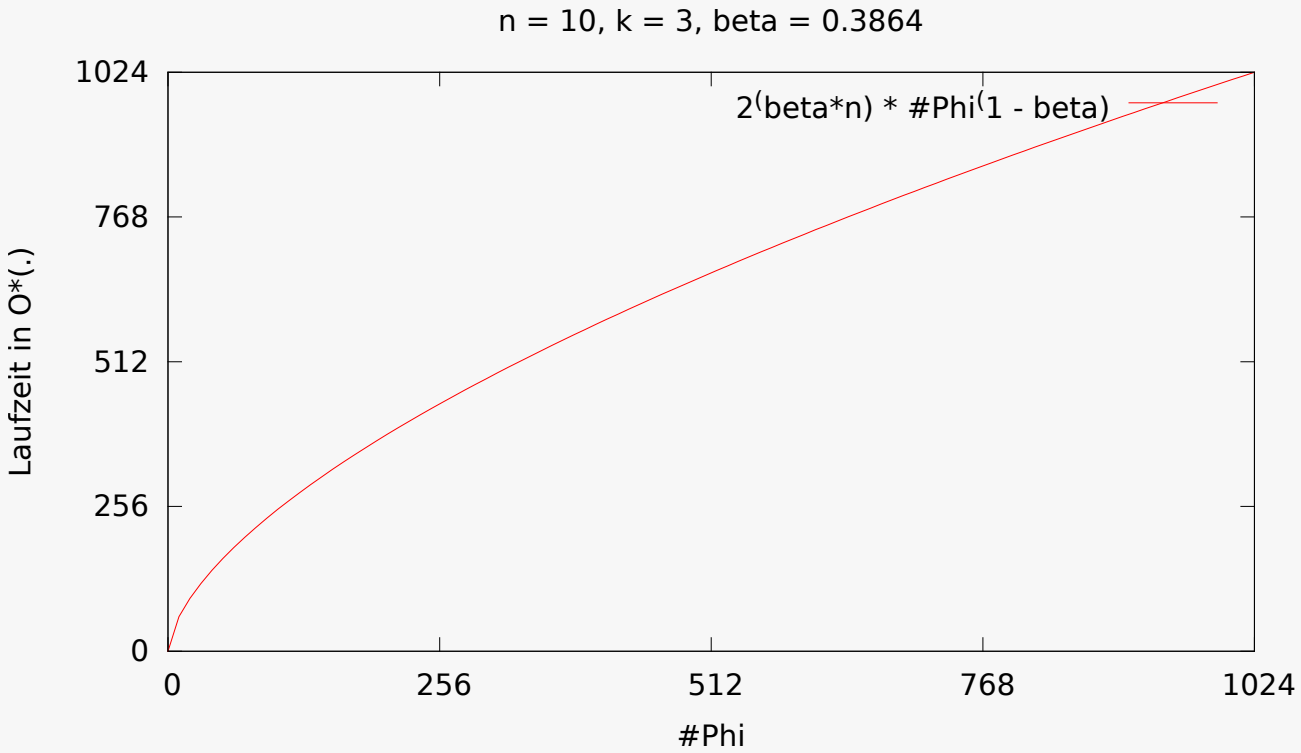
ℓ-Cut, max. Anzahl Knoten:	$\ell \cdot n$
Binärbaum, Höhe:	$\log_2(\#Knoten)$
Binärbaum, Anzahl Knoten pro Level:	2^i
Laufzeit SAT-Solver:	$2^{\beta_k \cdot \#Variablen}$
n-te Partialsumme der geometrischen Reihe:	$\sum_{k=0}^{n} q^k = \frac{q^{n+1}-1}{q-1}$

$$T_{\text{cut}} \in \mathcal{O}^*\left(\sum_{i=0}^{\lceil \log(n\cdot\ell)\rceil} 2^i \cdot 2^{\beta_k\cdot(n-i)}\right) = \mathcal{O}^*\left(2^{\beta_k\cdot n} \cdot \sum_{i=0}^{\lceil \log(n\cdot\ell)\rceil} 2^{(1-\beta_k)\cdot i}\right)$$
$$= \mathcal{O}^*(2^{\beta_k\cdot n} \cdot \ell \underbrace{^{1-\beta_k}}_{\in[0;1)})$$

Bemerkung: $\ell^{1-\beta_k}$ ist im Allgemeinen *kein* polynomieller Faktor, $\ell \leq 2^n$.

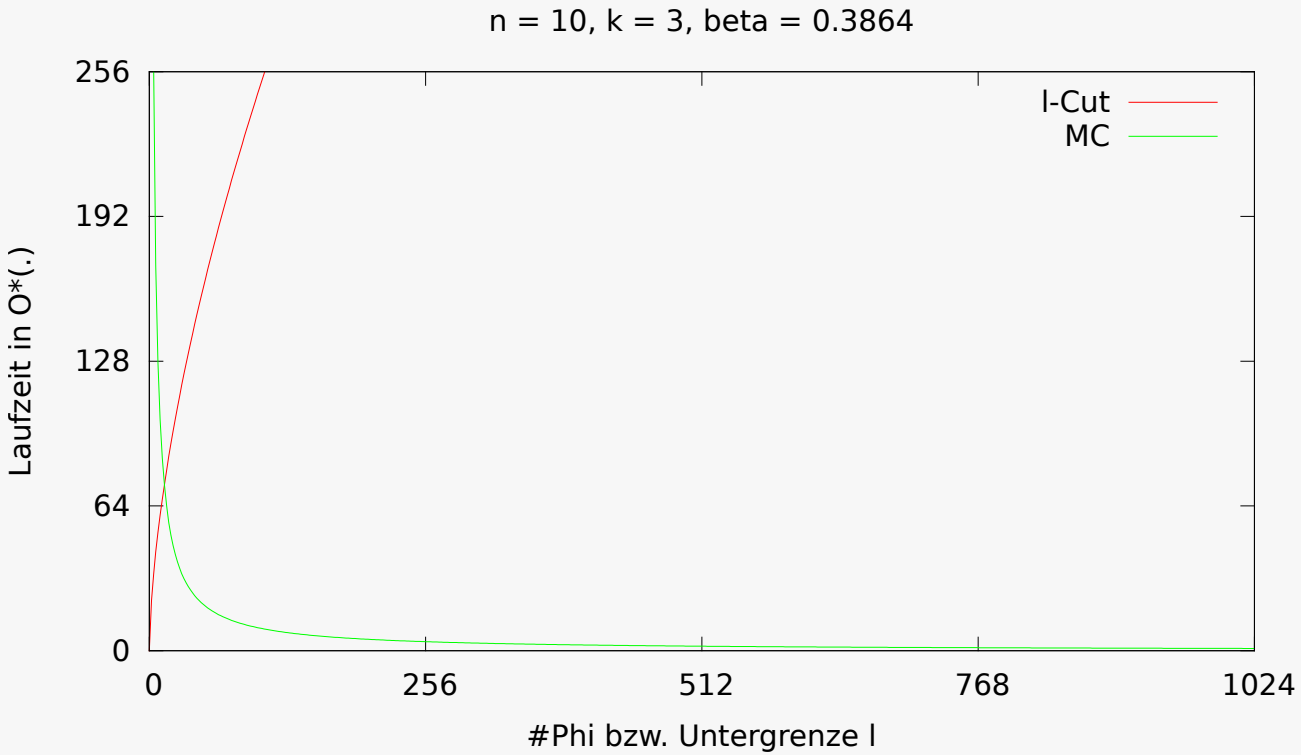
Laufzeit zum Ermitteln eines ℓ-Cuts, Abhängigkeit von $\#\Phi$

- Bricht die Berechnung eines ℓ-Cuts vorzeitig ab, weil keine erfüllbaren Äste mehr existieren, so ist $\#\Phi$ gleich der aktuellen Anzahl Blätter des Schnitts.



- Je größer $\#\Phi$, desto aufwendiger ist die Berechnung des ℓ-Cuts

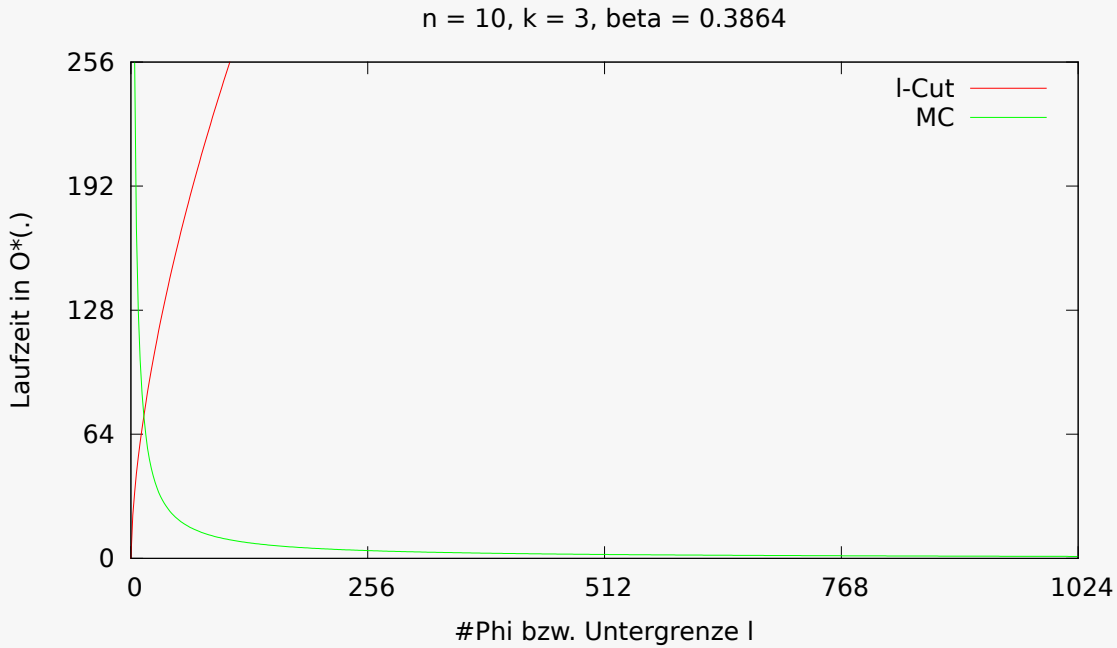
Laufzeit von ℓ-Cut und Monte-Carlo



- Die Stärken beider Algorithmen ergänzen sich

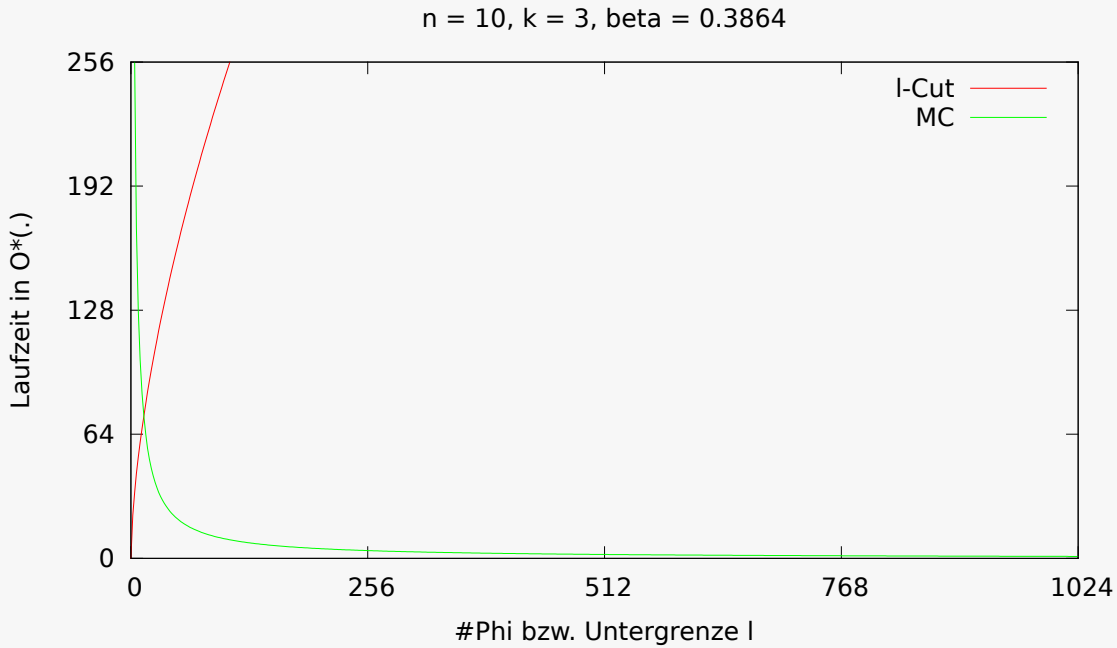
Hybrider Algorithmus

- Berechne zuerst einen ℓ-Cut
  - ℓ-Cut existiert nicht: $\#\Phi$ entspricht der Anzahl der Blätter im Schnitt
  - ℓ-Cut existiert: ℓ ist eine Untergrenze der Lösungen von Φ
- Starte MC mit der Untergrenze ℓ und dem gewünschten ε



Hybrider Algorithmus

- Berechne zuerst einen ℓ-Cut
  - ℓ-Cut existiert nicht: $\#\Phi$ entspricht der Anzahl der Blätter im Schnitt
  - ℓ-Cut existiert: ℓ ist eine Untergrenze der Lösungen von Φ
- Starte MC mit der Untergrenze ℓ und dem gewünschten ε



- Welcher Wert von ℓ ist der optimale Umschaltpunkt?

Optimales ℓ für die kleinste Worst-Case Laufzeit

$$\mathsf{T}_{\mathsf{Cut}} \in \mathcal{O}^*(2^{\beta_k \cdot n} \cdot \ell^{1-\beta_k}) \qquad \mathsf{T}_{\mathsf{MC}} \in \mathcal{O}^*\left(\frac{1}{\varepsilon^2} \cdot \frac{2^n}{\ell}\right)$$

$$\ell^* = \mathsf{arg\text{-}min}_\ell \{\mathsf{T}_{\mathsf{Cut}} + \mathsf{T}_{\mathsf{MC}}\}$$

Optimales ℓ für die kleinste Worst-Case Laufzeit

$$\mathsf{T}_{\mathsf{Cut}} \in \mathcal{O}^*(2^{\beta_k \cdot n} \cdot \ell^{1-\beta_k}) \qquad \mathsf{T}_{\mathsf{MC}} \in \mathcal{O}^*\left(\frac{1}{\varepsilon^2} \cdot \frac{2^n}{\ell}\right)$$

$$\ell^* = \text{arg-min}_\ell \{\mathsf{T}_{\mathsf{Cut}} + \mathsf{T}_{\mathsf{MC}}\}$$

$$\frac{\partial}{\partial \ell}(\mathsf{T}_{\mathsf{Cut}} + \mathsf{T}_{\mathsf{MC}}) \stackrel{!}{=} 0$$

Optimales ℓ für die kleinste Worst-Case Laufzeit

$$\mathsf{T}_{\mathsf{Cut}} \in \mathcal{O}^*(2^{\beta_k \cdot n} \cdot \ell^{1-\beta_k}) \qquad \mathsf{T}_{\mathsf{MC}} \in \mathcal{O}^*\left(\frac{1}{\varepsilon^2} \cdot \frac{2^n}{\ell}\right)$$

$$\ell^* = \text{arg-min}_\ell \{\mathsf{T}_{\mathsf{Cut}} + \mathsf{T}_{\mathsf{MC}}\}$$

$$\frac{\partial}{\partial \ell}(\mathsf{T}_{\mathsf{Cut}} + \mathsf{T}_{\mathsf{MC}}) \stackrel{!}{=} 0$$

$$\ell^* = 2^{\frac{1-\beta_k}{2-\beta_k} \cdot n}$$

Resultierende Worst-Case Laufzeit:

Optimales ℓ für die kleinste Worst-Case Laufzeit

$$\mathsf{T}_{\mathsf{Cut}} \in \mathcal{O}^*(2^{\beta_k \cdot n} \cdot \ell^{1-\beta_k}) \qquad \mathsf{T}_{\mathsf{MC}} \in \mathcal{O}^*\left(\frac{1}{\varepsilon^2} \cdot \frac{2^n}{\ell}\right)$$

$$\ell^* = \text{arg-min}_\ell \{\mathsf{T}_{\mathsf{Cut}} + \mathsf{T}_{\mathsf{MC}}\}$$

$$\frac{\partial}{\partial \ell}(\mathsf{T}_{\mathsf{Cut}} + \mathsf{T}_{\mathsf{MC}}) \stackrel{!}{=} 0$$

$$\ell^* = 2^{\frac{1-\beta_k}{2-\beta_k} \cdot n}$$

Resultierende Worst-Case Laufzeit:

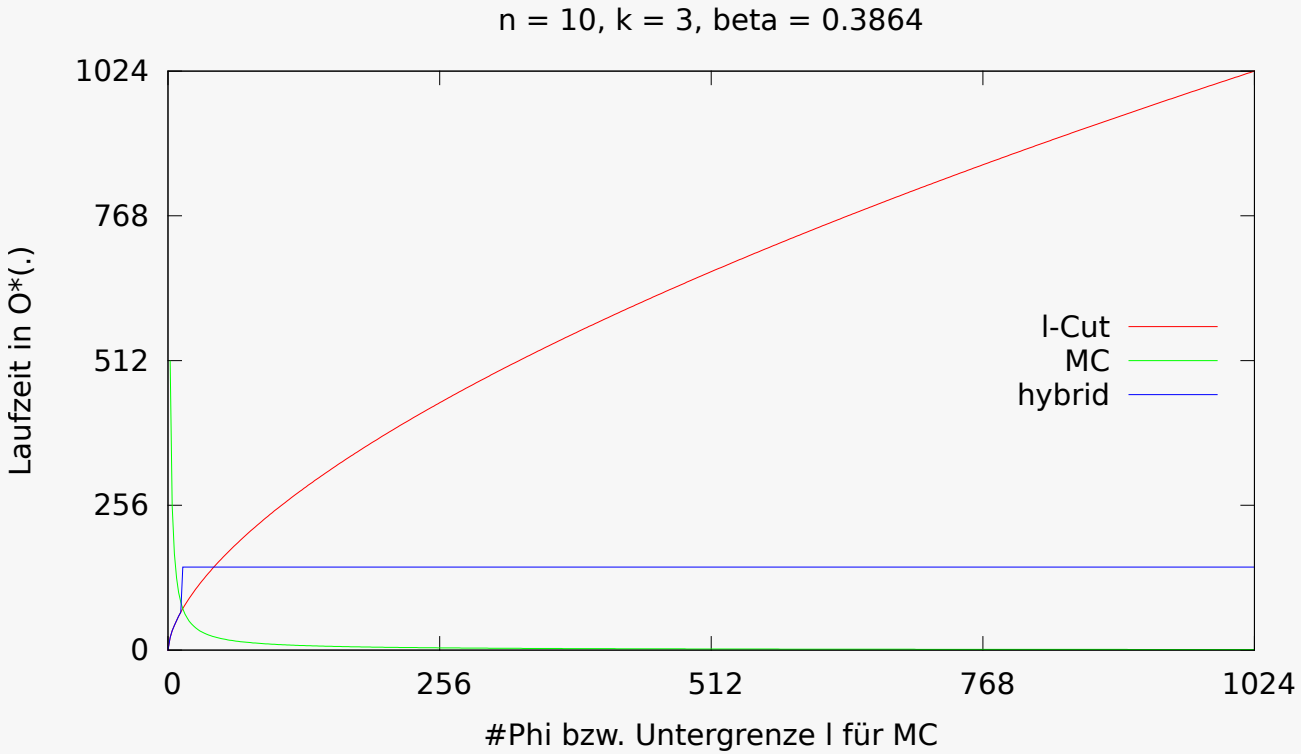
$$\mathcal{O}^*\left(\frac{1}{\varepsilon^2} \cdot 2^{\frac{1}{2-\beta_k} \cdot n}\right)$$

Worst-Case Laufzeiten des hybriden Algorithmus

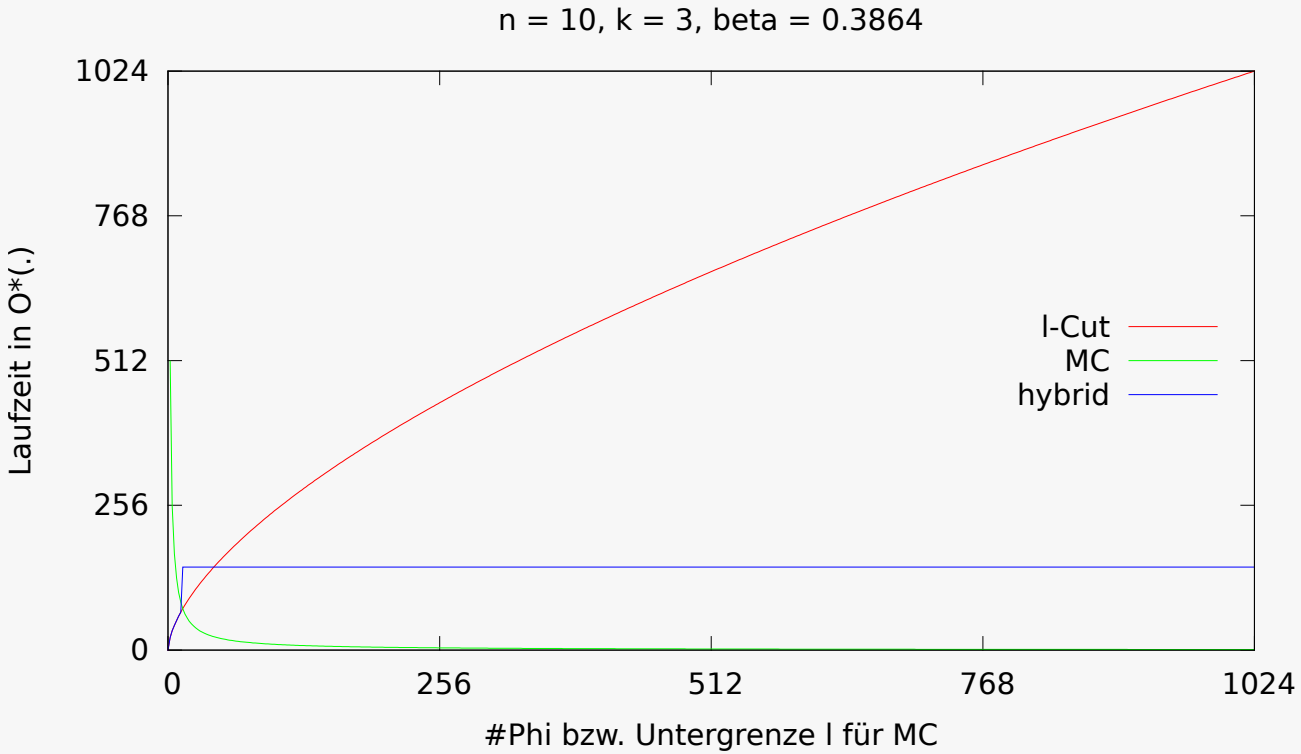
$$\mathcal{O}^*\left(\frac{1}{\varepsilon^2}\cdot 2^{\frac{1}{2-\beta_k}\cdot n}\right)$$

- $k = 3$: $\mathcal{O}^*(\varepsilon^{-2}\cdot 1.5366^n)$ $\beta_k = 0.3864$
- $k = 4$: $\mathcal{O}^*(\varepsilon^{-2}\cdot 1.6155^n)$ $\beta_k = 0.5548$
- $k \geq 5$: für β_k, siehe Literatur

Laufzeit des hybriden Algorithmus in Abhängigkeit von #Φ



Laufzeit des hybriden Algorithmus in Abhängigkeit von #Φ



Starke Verbesserung ohne in die Formel selbst hineinzuschauen!

1 Einführung
Was ist #k-SAT?

2 Monte-Carlo Counting
MC im Allgemeinen
MC für #k-SAT

3 ℓ-Cuts
Eliminationsbäume
ℓ-Cuts

4 Eliminationsbaum zurechtstutzen
Verbesserung der Laufzeit zum Finden von ℓ-Cuts

5 Unabhängige Klauseln und Rekursion
Wie man das Universum verkleinert...
Rekursion und Berechnung von ψ

6 Zusammenfassung

Grad des Eliminationsbaums reduzieren - Fall $\kappa = k$

$$\Phi = (x_1 \vee x_2) \wedge (\overline{x}_2 \vee x_3) \wedge (\overline{x}_3 \vee x_4) \wedge (\overline{x}_1 \vee x_2)$$

- Sei ϕ die von einem Knoten im Eliminationsbaum repräsentierte k-CNF Formel
- Sei C eine Klausel in ϕ mit $\kappa = k$ Literalen
- Es gibt genau eine Belegung der κ Variablen, die C nicht erfüllt

$$C = (x_1 \vee x_2) \stackrel{!}{=} (\text{false} \vee \text{false})$$
$$x_1 \leftarrow \text{false},$$
$$x_2 \leftarrow \text{false}$$

Grad des Eliminationsbaums reduzieren - Fall $\kappa = k$

$$\Phi = (x_1 \vee x_2) \wedge (\overline{x}_2 \vee x_3) \wedge (\overline{x}_3 \vee x_4) \wedge (\overline{x}_1 \vee x_2)$$

- Sei ϕ die von einem Knoten im Eliminationsbaum repräsentierte k-CNF Formel
- Sei C eine Klausel in ϕ mit $\kappa = k$ Literalen
- Es gibt genau eine Belegung der κ Variablen, die C nicht erfüllt

$$C = (x_1 \vee x_2) \stackrel{!}{=} (\text{false} \vee \text{false})$$
$$x_1 \leftarrow \text{false},$$
$$x_2 \leftarrow \text{false}$$

- Belegen wir die k Variablen von C, kann der Ast, der die Klausel nicht erfüllt, weggelassen werden

Grad des Eliminationsbaums reduzieren - $\kappa \leq k$

$$\Phi = (x_1 \vee x_2) \wedge (\overline{x}_2 \vee x_3) \wedge (\overline{x}_3 \vee x_4) \wedge (\overline{x}_1 \vee x_2)$$

- Sei C eine Klausel in ϕ mit $\kappa \leq k$ Literalen

Grad des Eliminationsbaums reduzieren - $\kappa \leq k$

$$\Phi = (x_1 \vee x_2) \wedge (\overline{x}_2 \vee x_3) \wedge (\overline{x}_3 \vee x_4) \wedge (\overline{x}_1 \vee x_2)$$

$$\downarrow \; x_1 \leftarrow \text{true}, x_2 \leftarrow \text{true}$$

$$(x_3) \wedge (\overline{x}_3 \vee x_4)$$

- Sei C eine Klausel in ϕ mit $\kappa \leq k$ Literalen

Grad des Eliminationsbaums reduzieren - $\kappa \leq k$

$$\Phi = (x_1 \vee x_2) \wedge (\overline{x}_2 \vee x_3) \wedge (\overline{x}_3 \vee x_4) \wedge (\overline{x}_1 \vee x_2)$$

$$\downarrow \; x_1 \leftarrow \text{true}, x_2 \leftarrow \text{true}$$

$$(x_3) \wedge (\overline{x}_3 \vee x_4)$$

- Sei C eine Klausel in ϕ mit $\kappa \leq k$ Literalen
- Belegen wir k Variablen einschließlich der in C enthaltenen, können **alle** Äste, die C nicht erfüllen, weggelassen werden

Grad des Eliminationsbaums reduzieren - $\kappa \leq k$

$$\Phi = (x_1 \vee x_2) \wedge (\overline{x}_2 \vee x_3) \wedge (\overline{x}_3 \vee x_4) \wedge (\overline{x}_1 \vee x_2)$$

$$\downarrow \; x_1 \leftarrow \text{true}, x_2 \leftarrow \text{true}$$

$$(x_3) \wedge (\overline{x}_3 \vee x_4)$$

$(x_3 \leftarrow \text{false}, x_4 \leftarrow \text{true})$ und $(x_3 \leftarrow \text{false}, x_4 \leftarrow \text{false})$ fallen weg.

- Sei C eine Klausel in ϕ mit $\kappa \leq k$ Literalen
- Belegen wir k Variablen einschließlich der in C enthaltenen, können **alle** Äste, die C nicht erfüllen, weggelassen werden
- Ersparnis: $2^{k-\kappa} \geq 1$ Äste

Grad des Eliminationsbaums reduzieren - $\kappa \leq k$

$$\Phi = (x_1 \vee x_2) \wedge (\overline{x}_2 \vee x_3) \wedge (\overline{x}_3 \vee x_4) \wedge (\overline{x}_1 \vee x_2)$$

$$\downarrow \; x_1 \leftarrow \text{true}, x_2 \leftarrow \text{true}$$

$$(x_3) \wedge (\overline{x}_3 \vee x_4)$$

$(x_3 \leftarrow \text{false}, x_4 \leftarrow \text{true})$ und $(x_3 \leftarrow \text{false}, x_4 \leftarrow \text{false})$ fallen weg.

- Sei C eine Klausel in ϕ mit $\kappa \leq k$ Literalen
- Belegen wir k Variablen einschließlich der in C enthaltenen, können **alle** Äste, die C nicht erfüllen, weggelassen werden
- Ersparnis: $2^{k-\kappa} \geq 1$ Äste

- Der Knoten hat also $2^k - 2^{k-\kappa} \leq 2^k - 1$ Kinder
- Die Höhe des Baumes reduziert sich auf $\lceil \frac{n}{k} \rceil$

Verbesserte Laufzeit zum Ermitteln eines ℓ-Cuts

verbesserter ℓ-Cut, max. Anzahl Knoten:	$\ell \cdot \lceil \frac{n}{k} \rceil$
Höhe eines Baumes mit Grad $2^k - 1$:	$\log_{2^k-1}(\#Knoten)$
Anzahl Knoten pro Level:	$(2^k - 1)^i$
Laufzeit SAT-Solver, Level i:	$2^{\beta_k \cdot (n - k \cdot i)}$
n-te Partialsumme der geometrischen Reihe:	$\sum_{k=0}^{n} q^k = \frac{q^{n+1}-1}{q-1}$

$$T_{\text{cut}} \in \mathcal{O}^* \left(\sum_{i=0}^{\lceil \log_{2^k-1}(\ell \cdot \frac{n}{k}) \rceil} (2^k - 1)^i \cdot 2^{\beta_k \cdot (n - k \cdot i)} \right)$$

$$= \mathcal{O}^*(2^{\beta_k \cdot n} \cdot \ell^{1 - \beta_k \cdot \frac{k}{\log(2^k - 1)}})$$

Verbesserte Laufzeit zum Ermitteln eines ℓ-Cuts

verbesserter ℓ-Cut, max. Anzahl Knoten:	$\ell \cdot \lceil \frac{n}{k} \rceil$
Höhe eines Baumes mit Grad $2^k - 1$:	$\log_{2^k-1}(\#Knoten)$
Anzahl Knoten pro Level:	$(2^k - 1)^i$
Laufzeit SAT-Solver, Level i:	$2^{\beta_k \cdot (n - k \cdot i)}$
n-te Partialsumme der geometrischen Reihe:	$\sum_{k=0}^{n} q^k = \frac{q^{n+1}-1}{q-1}$

$$T_{\text{cut}} \in \mathcal{O}^* \left(\sum_{i=0}^{\lceil \log_{2^k-1}(\ell \cdot \frac{n}{k}) \rceil} (2^k - 1)^i \cdot 2^{\beta_k \cdot (n - k \cdot i)} \right)$$

$$= \mathcal{O}^*(2^{\beta_k \cdot n} \cdot \ell^{1 - \beta_k \cdot \frac{k}{\log(2^k - 1)}})$$

$$\frac{k}{\log(2^k - 1)} = \frac{\log(2^k)}{\log(2^k - 1)} > 1$$

Änderung der Laufzeit des Hybriden Algorithmus

Laufzeit ℓ-Cut	$\mathcal{O}^*\left(2^{\beta_k \cdot n} \cdot \ell^{1-\beta_k \cdot \frac{k}{\log(2^k-1)}}\right)$	wird kleiner
Umschaltpunkt	$\ell = \exp\left(\frac{1-\beta_k}{2-\beta_k \cdot \frac{k}{\log(2^k-1)}} \cdot n\right)$	wird größer
Laufzeit Gesamt	$\mathcal{O}^*\left(\frac{1}{\varepsilon^2} \cdot 2^{p_k n}\right)$	

Änderung der Laufzeit des Hybriden Algorithmus

Laufzeit ℓ-Cut $\quad \mathcal{O}^*\left(2^{\beta_k \cdot n} \cdot \ell^{1-\beta_k \cdot \frac{k}{\log(2^k-1)}}\right)$ wird kleiner

Umschaltpunkt $\quad \ell = \exp\left(\frac{1-\beta_k}{2-\beta_k \cdot \frac{k}{\log(2^k-1)}} \cdot n\right)$ wird größer

Laufzeit Gesamt $\quad \mathcal{O}^*\left(\frac{1}{\varepsilon^2} \cdot 2^{p_k n}\right)$

$$p_k = \frac{1 - \beta_k \cdot \left(\frac{k}{\log(2^k-1)} - 1\right)}{2 - \beta_k \cdot \frac{k}{\log(2^k-1)}}$$

wird unübersichtlich

Änderung der Laufzeit des Hybriden Algorithmus

Laufzeit ℓ-Cut	$\mathcal{O}^*\left(2^{\beta_k \cdot n} \cdot \ell^{1-\beta_k \cdot \frac{k}{\log(2^k-1)}}\right)$	wird kleiner
Umschaltpunkt	$\ell = \exp\left(\frac{1-\beta_k}{2-\beta_k \cdot \frac{k}{\log(2^k-1)}} \cdot n\right)$	wird größer
Laufzeit Gesamt	$\mathcal{O}^*\left(\frac{1}{\varepsilon^2} \cdot 2^{p_k n}\right)$	wird kleiner

$$p_k = \frac{1 - \beta_k \cdot \left(\frac{k}{\log(2^k-1)} - 1\right)}{2 - \beta_k \cdot \frac{k}{\log(2^k-1)}}$$

Änderung der Laufzeit des Hybriden Algorithmus

Laufzeit ℓ-Cut	$\mathcal{O}^*\left(2^{\beta_k \cdot n} \cdot \ell^{1-\beta_k \cdot \frac{k}{\log(2^k-1)}}\right)$	wird kleiner
Umschaltpunkt	$\ell = \exp\left(\frac{1-\beta_k}{2-\beta_k \cdot \frac{k}{\log(2^k-1)}} \cdot n\right)$	wird größer
Laufzeit Gesamt	$\mathcal{O}^*\left(\frac{1}{\varepsilon^2} \cdot 2^{p_k n}\right)$	wird kleiner

$$p_k = \frac{1 - \beta_k \cdot \left(\frac{k}{\log(2^k-1)} - 1\right)}{2 - \beta_k \cdot \frac{k}{\log(2^k-1)}}$$

	Verbesserter ℓ-Cut	ursprünglicher ℓ-Cut
$k = 3$	$\mathcal{O}^*(\varepsilon^{-2} \cdot 1.5298^n)$	$\mathcal{O}^*(\varepsilon^{-2} \cdot 1.5366^n)$
$k = 4$	$\mathcal{O}^*(\varepsilon^{-2} \cdot 1.6122^n)$	$\mathcal{O}^*(\varepsilon^{-2} \cdot 1.6155^n)$

1 Einführung
Was ist #k-SAT?

2 Monte-Carlo Counting
MC im Allgemeinen
MC für #k-SAT

3 ℓ-Cuts
Eliminationsbäume
ℓ-Cuts

4 Eliminationsbaum zurechtstutzen
Verbesserung der Laufzeit zum Finden von ℓ-Cuts

5 Unabhängige Klauseln und Rekursion
Wie man das Universum verkleinert...
Rekursion und Berechnung von ψ

6 Zusammenfassung

Erinnerung

- Laufzeit des Monte-Carlo Algorithmus

$$\mathcal{O}^*\left(\frac{1}{\varepsilon^2}\cdot\frac{|\mathcal{U}_\Phi|}{\ell}\right)$$

- Bisheriger Fokus: Untergrenze ℓ

Erinnerung

- Laufzeit des Monte-Carlo Algorithmus

$$\mathcal{O}^*\left(\frac{1}{\varepsilon^2}\cdot\frac{|\mathcal{U}_\Phi|}{\ell}\right)$$

- Bisheriger Fokus: Untergrenze ℓ
- Neuer Ansatz: Reduzierung von $|\mathcal{U}_\Phi|$

Unabhängigkeit

$$\underbrace{(x_1 \vee x_2) \wedge (x_3 \vee x_4)}$$

- Zwei Klauseln sind unabhängig, wenn sie keine gemeinsamen Variablen enthalten.

Unabhängigkeit

$$\Phi = \underbrace{(x_1 \vee x_2) \wedge (x_3 \vee x_4)}_{\psi} \wedge (x_1 \vee x_5)$$

- Zwei Klauseln sind unabhängig, wenn sie keine gemeinsamen Variablen enthalten.
- Betrachte Φ als Menge von Klauseln.
  Eine Teilformel $\psi \subseteq \Phi$ ist unabhängig, wenn die enthaltenen Klauseln paarweise unabhängig sind.

Unabhängigkeit

$$\Phi = \underbrace{(x_1 \vee x_2) \wedge (x_3 \vee x_4)}_{\psi} \wedge (x_1 \vee x_5)$$

- Zwei Klauseln sind unabhängig, wenn sie keine gemeinsamen Variablen enthalten.
- Betrachte Φ als Menge von Klauseln.
  Eine Teilformel $\psi \subseteq \Phi$ ist unabhängig, wenn die enthaltenen Klauseln paarweise unabhängig sind.
- ψ ist maximal, wenn jede Klausel aus Φ mindestens eine Variable enthält, die in einer Klausel von ψ enthalten ist.

$$\Phi = \underbrace{(x_1 \vee x_2) \wedge (x_3 \vee x_4)}_{\psi} \wedge (x_1 \vee x_5)$$

- ψ ist Teil von Φ
- Φ enthält weitere Klauseln, ist also restriktiver

$$\Phi = \underbrace{(x_1 \vee x_2) \wedge (x_3 \vee x_4)}_{\psi} \wedge (x_1 \vee x_5)$$

- ψ ist Teil von Φ
- Φ enthält weitere Klauseln, ist also restriktiver

$$\mathcal{S}_\psi \supseteq \mathcal{S}_\Phi$$

$$\Phi = \underbrace{(x_1 \vee x_2) \wedge (x_3 \vee x_4)}_{\psi} \wedge (x_1 \vee x_5)$$

- ψ ist Teil von Φ
- Φ enthält weitere Klauseln, ist also restriktiver

$$\{0,1\}^n \supsetneq \mathcal{S}_\psi \supseteq \mathcal{S}_\Phi$$

Ein neues Universum

$$\Phi = \underbrace{(x_1 \vee x_2) \wedge (x_3 \vee x_4)}_{\psi} \wedge (x_1 \vee x_5)$$

- ψ ist Teil von Φ
- Φ enthält weitere Klauseln, ist also restriktiver

$$\{0,1\}^n \supsetneq \mathcal{S}_\psi \supseteq \mathcal{S}_\Phi$$

- Verwende $\mathcal{S}_\psi$ als Universum für die Approximation von $|\mathcal{S}_\Phi| = \#\Phi$
  $\Rightarrow$ weniger Wiederholungen

Ein neues Universum

$$\Phi = \underbrace{(x_1 \vee x_2) \wedge (x_3 \vee x_4)}_{\psi} \wedge (x_1 \vee x_5)$$

- ψ ist Teil von Φ
- Φ enthält weitere Klauseln, ist also restriktiver

$$\{0,1\}^n \supsetneq \mathcal{S}_\psi \supseteq \mathcal{S}_\Phi$$

- Verwende $\mathcal{S}_\psi$ als Universum für die Approximation von $|\mathcal{S}_\Phi| = \#\Phi$
  $\Rightarrow$ weniger Wiederholungen

- Warum verwenden wir nicht gleich $\mathcal{S}_\Phi$?

Ein neues Universum

$$\Phi = \underbrace{(x_1 \vee x_2) \wedge (x_3 \vee x_4)}_{\psi} \wedge (x_1 \vee x_5)$$

- ψ ist Teil von Φ
- Φ enthält weitere Klauseln, ist also restriktiver

$$\{0,1\}^n \supsetneq \mathcal{S}_\psi \supseteq \mathcal{S}_\Phi$$

- Verwende $\mathcal{S}_\psi$ als Universum für die Approximation von $|\mathcal{S}_\Phi| = \#\Phi$
  $\Rightarrow$ weniger Wiederholungen

- Warum verwenden wir nicht gleich $\mathcal{S}_\Phi$? Stichproben ziehen $\in \mathcal{NP}$

Ziehen von uniform verteilten Stichproben aus $|\mathcal{S}_\psi|$

$$\Phi = \underbrace{\overbrace{(x_1 \vee x_2)} \wedge \overbrace{(x_3 \vee x_4)}}_{\psi} \wedge (x_1 \vee \overbrace{x_5})$$

Ziehen von uniform verteilten Stichproben aus $|\mathcal{S}_\psi|$

$$\Phi = \underbrace{\overbrace{(x_1 \vee x_2)}^{(1)} \wedge \overbrace{(x_3 \vee x_4)}^{(1)}}_{\psi} \wedge (x_1 \vee \overbrace{x_5}^{})$$

1. Für jede Klausel aus ψ, wähle uniform eine erfüllende Belegung der enthaltenen Variablen

Ziehen von uniform verteilten Stichproben aus $|\mathcal{S}_\psi|$

$$\Phi = \underbrace{\overbrace{(x_1 \vee x_2)}^{(1)} \wedge \overbrace{(x_3 \vee x_4)}^{(1)}}_{\psi} \wedge (x_1 \vee \overbrace{x_5}^{(2)})$$

1. Für jede Klausel aus ψ, wähle uniform eine erfüllende Belegung der enthaltenen Variablen
2. Für alle anderen Variablen, wähle uniform eine Belegung

Monte-Carlo Counting im neuen Universum

$$\Phi = \underbrace{(x_1 \vee x_2) \wedge (x_3 \vee x_4)}_{\psi} \wedge (x_1 \vee x_5)$$

- Wiederhole T mal
  - Für jede Variable
    - Mit Wahrscheinlichkeit $\frac{1}{2}$: $x_i := 1$, sonst $x_i := 0$
  - Werte Φ aus

$$X_t = \begin{cases} 1 & \text{erfüllende Belegung} \\ 0 & \text{sonst} \end{cases}$$

- $R = \frac{1}{T} \cdot \sum_{t=1}^{T} X_t$
- Das Ergebnis ist $R \cdot |\mathcal{S}_\psi|$

Monte-Carlo Counting im neuen Universum

$$\Phi = \underbrace{(x_1 \vee x_2) \wedge (x_3 \vee x_4)}_{\psi} \wedge (x_1 \vee x_5)$$

- Wiederhole T mal
  - Für jede Klausel in ψ
    - Würfle von 1 bis $2^k - 1$, setze die Literale entsprechend der Bits
  - Für jede übrige Variable
    - Mit Wahrscheinlichkeit $\frac{1}{2}$: $x_i := 1$, sonst $x_i := 0$
  - Werte Φ aus

$$X_t = \begin{cases} 1 & \text{erfüllende Belegung} \\ 0 & \text{sonst} \end{cases}$$

- $R = \frac{1}{T} \cdot \sum_{t=1}^{T} X_t$
- Das Ergebnis ist $R \cdot |\mathcal{S}_\psi|$

Laufzeit Monte-Carlo mit kleinerem Universum

$$|\mathcal{S}_\psi| \leq (2^k - 1)^{|\psi|} \cdot 2^{n-k\cdot|\psi|}$$

- $|\psi|$ = number of clauses in ψ

Laufzeit Monte-Carlo mit kleinerem Universum

$$|\mathcal{S}_\psi| \leq (2^k - 1)^{|\psi|} \cdot 2^{n-k\cdot|\psi|} = \underbrace{2^n}_{=|\mathcal{U}_\Phi|} \cdot \underbrace{(1-2^{-k})^{|\psi|}}_{<1}$$

- $|\psi|$ = number of clauses in ψ

Laufzeit Monte-Carlo mit kleinerem Universum

$$|\mathcal{S}_\psi| \leq (2^k - 1)^{|\psi|} \cdot 2^{n-k\cdot|\psi|} = \underbrace{2^n}_{=|\mathcal{U}_\Phi|} \cdot \underbrace{(1-2^{-k})^{|\psi|}}_{<1}$$

- $|\psi|$ = number of clauses in ψ
- Es ergibt sich die Laufzeit

$$\mathcal{O}^*\left(\frac{1}{\varepsilon^2} \cdot \frac{2^n}{\ell} \cdot (1-2^{-k})^{|\psi|}\right)$$

Was bringt uns das neue Universum $\mathcal{S}_\psi$ im Worst-Case?

$$\mathcal{O}^*\left(\frac{1}{\varepsilon^2}\cdot\frac{2^n}{\ell}\cdot(1-2^{-k})^{|\psi|}\right)$$

Was bringt uns das neue Universum $\mathcal{S}_\psi$ im Worst-Case?

$$\mathcal{O}^*\left(\frac{1}{\varepsilon^2}\cdot\frac{2^n}{\ell}\cdot(1-2^{-k})^{|\psi|}\right)$$

$$\Phi = (x_1 \vee x_2) \wedge (x_1 \vee x_3) \wedge (\overline{x}_1 \vee x_4) \wedge (\overline{x}_1 \vee x_5)$$

Was bringt uns das neue Universum $\mathcal{S}_\psi$ im Worst-Case?

$$\mathcal{O}^*\left(\frac{1}{\varepsilon^2}\cdot\frac{2^n}{\ell}\cdot(1-2^{-k})^{|\psi|}\right)$$

$$\Phi = (x_1 \vee x_2) \wedge (x_1 \vee x_3) \wedge (\overline{x}_1 \vee x_4) \wedge (\overline{x}_1 \vee x_5) \qquad |\psi| = 1$$

- $(1-2^{-k})^1 \overset{k:=2}{=} \frac{3}{4}$ ist im Worst-Case ein konstanter Faktor
- Die Ersparnis wird in $\mathcal{O}$-Notation komplett vernachlässigt!

$$\Phi = (x_1 \vee x_2) \wedge (x_1 \vee x_3) \wedge (\overline{x}_1 \vee x_4) \wedge (\overline{x}_1 \vee x_5)$$

Rekursionsidee

$$\Phi = (x_1 \vee x_2) \wedge (x_1 \vee x_3) \wedge (\overline{x}_1 \vee x_4) \wedge (\overline{x}_1 \vee x_5)$$

$$= \#\Phi|_{x_1=\text{true}} + \#\Phi|_{x_1=\text{false}} \qquad \text{(vgl. Elim.-Baum)}$$

Rekursionsidee

$$\Phi = (x_1 \vee x_2) \wedge (x_1 \vee x_3) \wedge (\overline{x}_1 \vee x_4) \wedge (\overline{x}_1 \vee x_5)$$
$$= \#\Phi|_{x_1=\text{true}} + \#\Phi|_{x_1=\text{false}} \qquad \text{(vgl. Elim.-Baum)}$$

$$\Phi|_{x_1=\text{true}} = (\text{true} \vee x_2) \wedge (\text{true} \vee x_3) \wedge (\text{false} \vee x_4) \wedge (\text{false} \vee x_5)$$

$$\Phi|_{x_1=\text{false}} = (\text{false} \vee x_2) \wedge (\text{false} \vee x_3) \wedge (\text{true} \vee x_4) \wedge (\text{true} \vee x_5)$$

Rekursionsidee

$$\begin{aligned}\Phi &= (x_1 \vee x_2) \wedge (x_1 \vee x_3) \wedge (\overline{x}_1 \vee x_4) \wedge (\overline{x}_1 \vee x_5) \\ &= \#\Phi|_{x_1=\text{true}} + \#\Phi|_{x_1=\text{false}} \qquad \text{(vgl. Elim.-Baum)}\end{aligned}$$

$$\begin{aligned}\Phi|_{x_1=\text{true}} &= (\text{true} \vee x_2) \wedge (\text{true} \vee x_3) \wedge (\text{false} \vee x_4) \wedge (\text{false} \vee x_5) \\ &= (x_4) \wedge (x_5) \\ \Phi|_{x_1=\text{false}} &= (\text{false} \vee x_2) \wedge (\text{false} \vee x_3) \wedge (\text{true} \vee x_4) \wedge (\text{true} \vee x_5) \\ &= (x_2) \wedge (x_3)\end{aligned}$$

Rekursionsidee

$$\Phi = (x_1 \vee x_2) \wedge (x_1 \vee x_3) \wedge (\overline{x}_1 \vee x_4) \wedge (\overline{x}_1 \vee x_5) \qquad \in \#\text{2-SAT}$$
$$= \#\Phi|_{x_1=\text{true}} + \#\Phi|_{x_1=\text{false}} \qquad \text{(vgl. Elim.-Baum)}$$

$$\begin{aligned}
\Phi|_{x_1=\text{true}} &= (\text{true} \vee x_2) \wedge (\text{true} \vee x_3) \wedge (\text{false} \vee x_4) \wedge (\text{false} \vee x_5) \\
&= (x_4) \wedge (x_5) \quad \in \#\text{1-SAT} \\
\Phi|_{x_1=\text{false}} &= (\text{false} \vee x_2) \wedge (\text{false} \vee x_3) \wedge (\text{true} \vee x_4) \wedge (\text{true} \vee x_5) \\
&= (x_2) \wedge (x_3) \quad \in \#\text{1-SAT}
\end{aligned}$$

Mehr, aber deutlich einfachere Instanzen!

Rekursion im Allgemeinen

$$\Phi = (x_1 \vee x_2) \wedge (x_1 \vee x_3) \wedge (\overline{x}_1 \vee x_4) \wedge (\overline{x}_1 \vee x_5)$$

- Wenn $|\psi|$ klein ist
  - Belege *alle* Variablen aus ψ in allen erfüllbaren Kombinationen
  - Löse die $2^{k \cdot |\psi|}$ Subprobleme der Größe $n' \leq n - k \cdot |\psi|$ aus $(k-1)$-#SAT

Rekursion im Allgemeinen

$$\Phi = (x_1 \vee x_2) \wedge (x_1 \vee x_3) \wedge (\overline{x}_1 \vee x_4) \wedge (\overline{x}_1 \vee x_5)$$

- Wenn $|\psi|$ klein ist
  - Belege *alle* Variablen aus ψ in allen erfüllbaren Kombinationen
  - Löse die $2^{k \cdot |\psi|}$ Subprobleme der Größe $n' \leq n - k \cdot |\psi|$ aus $(k-1)$-#SAT
- Im Worst-Case (nur unabhängige Klauseln) würde das Verfahren $(2^k - 1)^m < 2^n$ Subprobleme erzeugen

Hybrider(er) Algorithmus

- Berechne ψ nach dem Greedy-Prinzip
- Wenn $|\psi| < \hat{m}$:
  - Wenn $k - 1 = 2$: Verwende Wahlström's Algorithmus
  - Sonst: Berechne rekursiv die Summe von $\#(k-1)$-SAT aller ψ-erfüllenden Belegungen
- Sonst:

Hybrider(er) Algorithmus

- Berechne ψ nach dem Greedy-Prinzip
- Wenn $|\psi| < \hat{m}$:
  - Wenn $k - 1 = 2$: Verwende Wahlström's Algorithmus
  - Sonst: Berechne rekursiv die Summe von $\#(k-1)$-SAT aller ψ-erfüllenden Belegungen
- Sonst:
  - Berechne zuerst einen ℓ-Cut
    - ℓ-Cut existiert nicht: $\#\Phi$ entspricht der Anzahl der Blätter im Schnitt
    - ℓ-Cut existiert: ℓ ist eine Untergrenze der Lösungen von Φ
  - Starte MC mit der Untergrenze ℓ und dem Universum $\mathcal{U}_\psi$

Hybrider(er) Algorithmus

- Berechne ψ nach dem Greedy-Prinzip
- Wenn $|\psi| < \hat{m}$:
  - Wenn $k-1=2$: Verwende Wahlström's Algorithmus
  - Sonst: Berechne rekursiv die Summe von $\#(k-1)$-SAT aller ψ-erfüllenden Belegungen
- Sonst:
  - Berechne zuerst einen ℓ-Cut
    - ℓ-Cut existiert nicht: $\#\Phi$ entspricht der Anzahl der Blätter im Schnitt
    - ℓ-Cut existiert: ℓ ist eine Untergrenze der Lösungen von Φ
  - Starte MC mit der Untergrenze ℓ und dem Universum $\mathcal{U}_\psi$

Es gilt:

- Für die max. Anzahl rekursiver Aufrufe: $(2^k-1)^{|\psi|} \leq (2^k-1)^{\hat{m}-1}$

Hybrider(er) Algorithmus

- Berechne ψ nach dem Greedy-Prinzip
- Wenn $|\psi| < \hat{m}$:
  - Wenn $k-1=2$: Verwende Wahlström's Algorithmus
  - Sonst: Berechne rekursiv die Summe von $\#(k-1)$-SAT aller ψ-erfüllenden Belegungen
- Sonst: $|\psi| \geq \hat{m}$
  - Berechne zuerst einen ℓ-Cut
    - ℓ-Cut existiert nicht: $\#\Phi$ entspricht der Anzahl der Blätter im Schnitt
    - ℓ-Cut existiert: ℓ ist eine Untergrenze der Lösungen von Φ
  - Starte MC mit der Untergrenze ℓ und dem Universum $\mathcal{U}_\psi$

Es gilt:

- Für die max. Anzahl rekursiver Aufrufe: $(2^k-1)^{|\psi|} \leq (2^k-1)^{\hat{m}-1}$
- Für die min. Reduzierung des Universums: $(1-2^{-k})^{|\psi|} \geq (1-2^{-k})^{\hat{m}}$

$$|\mathcal{U}_\psi| = 2^n \cdot (1-2^k)^{|\psi|}$$

Optimierung von $\hat{m}$ und ℓ

$$\text{Wahlström} \in \mathcal{O}^*(1.2377^n)$$

$$\text{Rekursion} \in \mathcal{O}^*\left((2^k-1)^{\hat{m}} \cdot \mathsf{T}(\#(k-1)\text{-SAT})\right)$$

$$\ell\text{-Cut} \in \mathcal{O}^*\left(2^{\beta_k \cdot n} \cdot \ell^{1-\beta_k \frac{k}{\log(2^k-1)}}\right)$$

$$\text{MC} \in \mathcal{O}^*\left(\frac{1}{\varepsilon^2} \cdot \frac{2^n}{\ell} \cdot (1-2^{-k})^{\hat{m}}\right)$$

- 2-dimensionales Optimierungsproblem
- Die Parameter $\hat{m}$ und ℓ können wieder durch Null-Setzen der partiellen Ableitungen berechnet werden

Worst-Case Laufzeiten des hybrid(er)en Algorithmus

	Unabh. Klauseln	Verbesserter ℓ-Cut	ursprünglicher ℓ-Cut
$k = 3$	$\mathcal{O}^*(\varepsilon^{-2} \cdot 1.5181^n)$	$\mathcal{O}^*(\varepsilon^{-2} \cdot 1.5298^n)$	$\mathcal{O}^*(\varepsilon^{-2} \cdot 1.5366^n)$
$k = 4$	$\mathcal{O}^*(\varepsilon^{-2} \cdot 1.6105^n)$	$\mathcal{O}^*(\varepsilon^{-2} \cdot 1.6122^n)$	$\mathcal{O}^*(\varepsilon^{-2} \cdot 1.6155^n)$

1 Einführung
Was ist $\#k$-SAT?

2 Monte-Carlo Counting
MC im Allgemeinen
MC für $\#k$-SAT

3 ℓ-Cuts
Eliminationsbäume
ℓ-Cuts

4 Eliminationsbaum zurechtstutzen
Verbesserung der Laufzeit zum Finden von ℓ-Cuts

5 Unabhängige Klauseln und Rekursion
Wie man das Universum verkleinert...
Rekursion und Berechnung von ψ

6 Zusammenfassung

Zusammenfassung

Ausblick

- Wir haben die Verwendung der randomisierten SAT-Solver nicht exakt analysiert.
- Deren Fehlerwahrscheinlichkeit kann mit Hilfe der "Median of Means"-Methode, also durch eine polynomielle Anzahl zusätzlicher Wiederholungen, klein genug gehalten werden.
- Dies hat keine Auswirkung auf die Laufzeit in $\mathcal{O}^*$-Notation

Ausblick

- Wir haben die Verwendung der randomisierten SAT-Solver nicht exakt analysiert.
- Deren Fehlerwahrscheinlichkeit kann mit Hilfe der "Median of Means"-Methode, also durch eine polynomielle Anzahl zusätzlicher Wiederholungen, klein genug gehalten werden.
- Dies hat keine Auswirkung auf die Laufzeit in $\mathcal{O}^*$-Notation

- Für die Rekursion und die Verkleinerung des MC-Universums ist es nicht notwendig, ausschließlich Klauseln zu verwenden
- Solange die Länge der verwendeten Teilformeln konstant ist, kann die Anzahl an Lösungen ohne zusätzlichen asymptotischen Aufwand berechent werden

Ausblick

- Wir haben die Verwendung der randomisierten SAT-Solver nicht exakt analysiert.
- Deren Fehlerwahrscheinlichkeit kann mit Hilfe der "Median of Means"-Methode, also durch eine polynomielle Anzahl zusätzlicher Wiederholungen, klein genug gehalten werden.
- Dies hat keine Auswirkung auf die Laufzeit in $\mathcal{O}^*$-Notation

- Für die Rekursion und die Verkleinerung des MC-Universums ist es nicht notwendig, ausschließlich Klauseln zu verwenden
- Solange die Länge der verwendeten Teilformeln konstant ist, kann die Anzahl an Lösungen ohne zusätzlichen asymptotischen Aufwand berechent werden

- Beim Erstellen eines ℓ-Cuts muss der SAT-Solver sehr ähnliche Formeln lösen. Möglicherweise ließe sich das zugunsten einer besseren Laufzeit ausnutzen.

Vielen Dank für die Aufmerksamkeit!

Fragen?