Konstruktiver Beweis des Lovász Local Lemma - Am Beispiel von Erfüllbarkeit von k-KNF Formeln

Naja v. Schmude

7. Juni 2011

1 Einleitung

Bei einer KNF Formel mit genau k Literalen pro Klausel (*k-KNF Formel*) wird bei einer zufälligen, gleichverteilten Belegung der Variablen eine Klausel mit Wahrscheinlichkeit $\frac{1}{2^k}$ verletzt. Hat eine k-KNF Formel $n < 2^k$ Klauseln, so ist die erwartete Anzahl von verletzten Klauseln

$$E(\text{Anz. verletzer Klauseln}) = \sum_{i=1}^{n} E(\text{Klausel } i \text{ verletzt}) = \sum_{i=1}^{n} \frac{1}{2^k} = \frac{n}{2^k} < 1 \ .$$

Daraus folgt, dass es eine Variablenbelegung geben muss, die die Formel erfüllt.

Bei dieser globalen Sicht auf die KNF kann sehr einfach probabilistisch an das Entscheidbarkeitsproblem gegangen werden, wie sieht es aber mit eine lokalen Sichtweise aus? Lokale Sicht bedeutet in diesem Fall, dass die *Nachbarschaft* einer Klausel betrachtet wird.

Definition 1 (Nachbarschaft einer Klausel)**.** *Die Nachbarschaft $\Gamma(C)$ einer Klausel C in einer k-KNF Formel F ist gegeben durch die Menge der Klauseln, welche gemeinsame Variablen mit C haben. Die Menge der Variablen, die in einer Klausel C auftreten, sei durch $vbl(C)$ gegeben.*

$$\Gamma(C) = \{D \in F \mid vbl(D) \cap vbl(C) \neq \emptyset\}$$

Im Folgenden soll gezeigt werden, dass man auch bei eingeschränkter Nachbarschaft die Erfüllbarkeit probabilistisch begründen kann, und zwar unter zu Hilfenahme des *Lovász Local Lemmas.*

2 Lovász Local Lemma

Theorem 1 (Lovász Local Lemma (symmetrische Version))**.** *Sei $\mathcal{A} = \{A_1, A_2, \dots A_m\}$ eine Menge von Zufallsereignissen eines Wahrscheinlichkeitsraums. Jedes Ereignis aus $\mathcal{A}$ hat eine Wahrscheinlichkeit von maximal p hat und ist maximal nur von d anderen Ereignissen aus $\mathcal{A}$ abhängig. Wenn $ep(d+1) \leq 1$ gilt, dann tritt mit positiver Wahrscheinlichkeit keines der Ereignisse aus $\mathcal{A}$ ein.*

Das Lovász Local Lemma wird nun im Bezug auf die Erfüllbarkeit von k-KNF Formeln umformuliert. Dazu definiert man das Ereignis, dass die i-te Klausel der k-KNF Formel F verletzt wird, als Zufallsereignisse A_i. Die Wahrscheinlichkeit, dass Ereignis A_i eintritt,

ist jeweils durch $Pr(A_i) = \frac{1}{2^k}$ gegeben. Damit eine k-KNF erfüllbar ist, folgt folgende Bedingung für die Größe der Nachbarschaft $d = |\Gamma(C)|$ einer Klausel:

$$d \leq \frac{2^k}{e} - 1$$

Theorem 2 (SAT Formulierung)**.** *Sei $k \in \mathbb{N}$ und F eine k-KNF Formel. Wenn $\forall C \in F : |\Gamma(C)| < \frac{2^k}{e} - 1$ gilt, dann ist F erfüllbar.*

Im Folgenden soll nun durch die Konstruktion einer erfüllenden Belegung der Variablen einer solchen k-KNF Formel das Theorem bewiesen werden.

3 Konstruktiver Beweis des Lovász Local Lemmas

Die Konstruktion einer erfüllenden Belegung geschieht über den Algorithmus 1.

Algorithmus 1 Konstruktionsalgorithmus

```
1: for all v ∈ F do
2:     v ← zufällige Belegung
3: end for
4: while ∃C ∈ F : C ist verletzt unter aktueller Variablenbelegung do
5:     C ← irgendeine verletzte Klausel aus F
6:     for all v ∈ vbl(C) do
7:         v ← zufällige Belegung
8:     end for
9: end while
```

Die Idee ist dabei immer die Klauseln mit einer neuen Variablenauswertung zu belegen, die verletzt worden sind. Dabei können dann durch diese neue Belegung maximal die benachbarten Klauseln verletzt werden. Da die Nachbarschaft eingeschränkt ist, sollte eine erfüllende Belegung gefunden werden können - und zwar in einer erwarteter Zeit, die polynomiell ist. Dies ist nun zu zeigen.

Definition 2 (Log des Algorithmus)**.** *Als Log L wird die Abbildung $L : \mathbb{N} \to F$ bezeichnet, die zur i-ten Iteration der While-Schleife die durch den Algorithmus ausgewählte Klausel C zuweist.*

Definition 3 (Witness-Tree)**.** *Ein Witness-Tree T ist ein Wurzelbaum mit einem Labeling $\sigma : V(T) \to F$ der Knoten des Baums zu den Klauseln von F. Bei einem durch ein Log L gegebenem Ablauf des Algorithmus wird der Witness-Tree zur i-ten Iteration folgendermaßen aufgebaut:*

- *Die Wurzel r erhält das Label $\sigma(r) = L(i)$.*
- *Für jede Iteration $j = i-1, i-2, \ldots, 0$ wird der Baum folgendermaßen erweitert*
  1. *Falls es einen Knoten v im bisher aufgebauten Baum gibt, sodass $L(j) \in \Gamma(\sigma(v))$, dann füge einen neuen Knoten mit Label $L(j)$ an einen der tiefsten Knoten an, die diese Bedingung erfüllen.*
  2. *Andernfalls fahre mit dem nächsten j fort. Bei $j = 0$ ist der Baum zu Ende konstruiert.*

Beispiel Gegeben sei folgende 3-KNF Formel (die Klauseln sind dabei aufsteigend durchnummeriert)

$$F = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_2 \vee x_3 \vee \neg x_4) \wedge (\neg x_1 \vee x_2 \vee \neg x_5) \wedge (x_5 \vee \neg x_6 \vee x_7)$$

und folgendes Log $L = 1, 4, 2, 3, 4, 1$. Der Witness-Tree für Iteration 5 sieht dann wie in Abbildung 1 gezeigt aus.

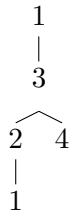

Abbildung 1: Beispiel Witness-Tree

Ein Witness-Tree dient als Rechtfertigung für einen Korrekturschritt bei der Ausführung des Algorithmuses. Durch diesen Baum kann der für den Korrekturschritt entscheidende Teil des Logs rekonstruiert werden, in dem man den Baum in einer umgekehrten Breitensuche von den Blättern zur Wurzel hin durchläuft. Man kann zudem aus dem Witness-Tree alle Werte ablesen, die eine Variable x zugewiesen bekommen hat, da man beim Traversieren des Baums die Anzahl der Klauseln bestimmen kann, in denen x vorkommt und somit wie oft diese Variable neu zugewiesen werden musste, da sie in vorangehenden Iterationen die entsprechende Klausel verletzt hatte.

Für einen gegebenen Witness-Tree T stellt sich nun die Frage, was die Wahrscheinlichkeit ist, dass genau T als Witness-Tree eines Korrekturschritts entstehen kann. Dabei hilft die gerade eben angestellte Überlegung, dass die Variablenbelegungen rekonstruiert werden können. Bei der Traversierung des Baums können für jeden Knoten genau k Variablen wiederhergestellt werden. Wenn T n Knoten hat, können insgesamt also nk Belegungen rekonstruiert werden. Die Wahrscheinlichkeit, dass der Algorithmus exakt diese Belegung erzeugt und somit den Baum T, beträgt $\frac{1}{2^{nk}}$.
Jetzt kann gezählt werden, wie viele Witness-Trees existieren können. Dazu wird eine Klausel C als Label der Wurzel fixiert und der unendliche Baum konstruiert, der dadurch entsteht, dass man alle benachbarten Klauseln als Label der Kinderknoten wählt usw. (siehe Beispiel in Abbildung 2 zur obigen Formel mit Klausel 1 als Wurzel).

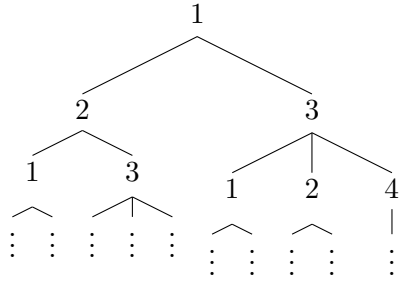

Abbildung 2: Unendlicher Witness-Tree

Ein so entstehender Baum ist maximal ein d-ärer Baum, welcher jeden Witness-Tree mit C in der Wurzel als Teilbaum besitzt. Ein unendlicher Witness-Tree hat maximal $(ed)^n$ Unterbäume der Größe n, entsprechend kann es also maximal $(ed)^n$ Witness-Trees der Größe n mit C als Label der Wurzel geben. Die Wahrscheinlichkeit, dass jeder der

Witness-Trees entstehen kann, wurde oben als $\frac{1}{2^{nk}}$ festgestellt, daher entspricht die erwartete Anzahl von entstehenden Witness-Trees

$$E(\text{Anzahl von entstehenden Witness-Trees}) = (ed)^n \cdot \frac{1}{2^{nk}}$$

Wenn man diesen Erwartungswert nun über alle möglichen $n \geq 1$ summiert, entsteht eine geometrische Reihe, welche zu einer Konstanten konvergiert. Somit gibt es maximal eine konstante Anzahl von Witness-Trees mit Wurzellabel C, die zu einem Korrekturschritt entstehen können.

Zurück zum eigentlichen Beweis. Wenn gezeigt werden kann, dass jede Klausel nur eine konstante Anzahl oft durch den Algorithmus resampled werden muss, findet man nach polynomieller Anzahl von Iterationen eine gültige Belegung der Variablen, was die Erfüllbarkeit beweist.
Eine Klausel C taucht $N(C)$ mal im Log L auf, und zwar in den Iterationen $i_1, i_2, \ldots, i_{N(C)}$. Zu jeder der Iterationen kann ein Witness-Tree geholt werden, der den Korrekturschritt begründet. Alle Witness-Trees zu den verschiedenen Iterationen sind unterschiedlich, da zwischen Iteration i_k und i_{k+1} mindestens ein Knoten zur Darstellung von Schritt i_{k+1} hinzugekommen sein muss. Da wie oben berechnet die Anzahl an Witness-Trees durch eine Konstante begrenzt ist, ist $N(C)$ ebenfalls durch eine Konstante nach oben begrenzt. Dies gilt für alle Klauseln, und somit terminiert der Algorithmus in einer erwarteten Laufzeit von polynomiell vielen Iterationen. □

4 General Lovász Local Lemma

Bisher wurde nur die vereinfachte (symmetrische) Variante des Lovász Local Lemmas verwendet. Die generelle Form lautet:

Theorem 3 (General Lovász Local Lemma)**.** *Sei $\mathcal{A}$ eine endliche Menge von Zufallsereignissen eines Wahrscheinlichkeitsraums. Für $A \in \mathcal{A}$ sei $\Gamma(A)$ die Teilmenge von $\mathcal{A}$, von der A abhängig ist. Falls eine Zurodnung $x : \mathcal{A} \to (0, 1)$ existiert, sodass*

$$\forall A \in \mathcal{A} : Pr(A) \leq x(A) \prod_{B \in \Gamma(A)} (1 - x(B)) \ ,$$

dann ist die Wahrscheinlichkeit, dass keins der Ereignisse aus $\mathcal{A}$ eintritt, maximal

$$\prod_{A \in \mathcal{A}} (1 - x(A)) \ .$$

Auch diese allgemeine Form kann konstruktiv bewiesen werden[2] und verläuft ähnlich wie der oben gezeigte am Beispiel von Erfüllbarkeit von k-KNF Formeln.

Literatur

[1] Gebauer, Heidi ; Moser, Robin ; Scheder, Dominik ; Welzl, Emo: The Lovász Local Lemma and Satisfiability. In: Albers, Susanne (Hrsg.) ; Alt, Helmut (Hrsg.) ; Näher, Stefan (Hrsg.): *Efficient Algorithms* Bd. 5760. Springer Berlin / Heidelberg, 2009, S. 30–54

[2] Moser, Robin A. ; Tardos, Gábor: A constructive proof of the general loász local lemma. In: *J. ACM* 57 (2010), February, S. 11:1–11:15. – ISSN 0004–5411