Lovász Local Lemma: an algorithmic approach

David Oertel

2. März 2011

Seminar Algorithmentechnik, WS 10/11
KIT – Institut für Theoretische Informatik, Prof. Wagner
Betreuer: Dr. Martin Nöllenburg

Zusammenfassung

Seit vielen Jahren wird die „klassische“ Erfüllbarkeit Boolescher Formeln, insbesondere in k-Konjunktiver-Normal-Form (k-KNF), unter einschränkenden Voraussetzungen untersucht. Einen wichtigen Beitrag hierzu lieferten in den 70er Jahren Lovász und Erdős mit dem *Lovász Local Lemma* [EL75], mithilfe dessen ohne größeren Aufwand die Erfüllbarkeit einer Booleschen Formel in k-KNF bestätigt werden kann, falls diverse Voraussetzungen für die Klauseln erfüllt sind. Problematisch für die Praxis ist dabei, dass das Lovász Local Lemma in seiner ursprünglichen Form eine rein existenzielle Aussage bleibt und man keine konkrete erfüllende Belegung erhält. Abhilfe hierfür schafft ein randomisierter Algorithmus von Moser und Tardos [Mos09, MT10], der erwartet in polynomieller Zeit abläuft und in diesem Paper vorgestellt wird.

1 Einleitung und Notation

Mithilfe des Lovász Local Lemma [EL75] kann unter gewissen Voraussetzungen die Erfüllbarkeit Boolescher Formeln in k-Konjunktiver Normal-Form (k-KNF, das heißt jede Klausel enthält k Literale) bestätigt werden, ohne jedoch – in der Grundform – eine konkrete erfüllende Belegung zu erhalten. Mit dem im Folgenden vorgestellten randomisierten Algorithmus kann man diesem Umstand entgegenwirken, um das Lovász Local Lemma auch praktisch nutzbar zu machen. Hierbei stehe stets $F = C_1, \ldots, C_m$ als eine m-elementige Menge von Klauseln für eine Boolesche Formel in k-KNF über den Variablen $\mathcal{X} = \{x_1, \ldots, x_n\}$. Des Weiteren bezeichne $\mathrm{vbl}(C) \subseteq \mathcal{X}$ die Variablen von Klausel C, für welche die *Nachbarschaft* mit $\Gamma(C) := \{D \in F \mid D \neq C, \mathrm{vbl}(C) \cap \mathrm{vbl}(D) \neq \emptyset\}$ sowie die inklusive Nachbarschaft mit $\Gamma^+(C) := \Gamma(C) \cup \{C\}$ bezeichnet sei. Eine *Belegung* der Variablen werde dabei mit $\alpha : \mathcal{X} \to \{0, 1\}$ benannt, zu welcher stets eine Menge an unerfüllten Klauseln $\mathrm{vlt}(F, \alpha) \subseteq F$ existiert.

2 Lovász Local Lemma

2.1 Grundidee

Die Grundaussage des Lovász Local Lemma ist folgende: Überschneiden/widersprechen sich nur „wenige“ Klauseln von einer k-KNF-Formel F, dann ist F erfüllbar. „überschneiden“ oder „widersprechen“ bedeutet hier, dass Klauseln Variablen teilen, unter Umständen sogar mit entgegengesetzten Vorzeichen.
Das Lovász Local Lemma ist in seiner ursprünglichen Form ein Lemma der Stochastik über endlich viele Ereignisse eines Wahrscheinlichkeitsraumes. Mit der passenden Modellierung lässt es sich auch auf Erfüllbarkeit Boolescher Formeln übertragen, indem die Erfüllbarkeit als Zufallsexperiment auf einem diskretem Wahrscheinlichkeits-Raum (Ω, P) betrachtet wird, wobei $\Omega = \{0,1\}^n$ die Variablen und ihre möglichen Belegungen repräsentiert. Diese Belegungen sollen gleichverteilt sein, also $P(\omega) = \frac{1}{|\Omega|} = \frac{1}{2^n} \quad \forall \omega \in \Omega$. Jede Klausel C_i aus F definiert dabei in kanonischer Weise ein Ereignis A_i, das alle Elementarereignisse ($\widehat{=}$ Belegungen) enthält, für die C_i *nicht* erfüllt ist. Mit diesen Ereignissen ist wichtig zu beachten, dass A_i genau dann stochastisch unabhängig von A_j ist, wenn $\text{vbl}(C_i) \cap \text{vbl}(C_j) = \emptyset$.

2.2 Varianten

Das Lovász Local Lemma in der sog. symmetrischen Form wird meist wie folgt formuliert[1].

Satz 1 (Lovász Local Lemma, symmetrische Form)**.** *Seien $\mathcal{A} = \{A_1, \ldots, A_M\}$ Ereignisse im W'Raum Ω, von denen jedes höchstens von d anderen Ereignissen aus $\mathcal{A}$ abhängig sei. Falls $P(A_i) \leq p \quad \forall A_i \in \mathcal{A}$ und zugleich $p \cdot e \cdot (d+1) \leq 1$, dann ist $P(\overline{A_1}, \ldots, \overline{A_M}) > 0$.*

Mit den im vorigen Abschnitt definierten A_i ist $P(A_i) = (\frac{1}{2})^k =: p$ und die Menge der von A_i abhängigen Ereignisse (ungleich A_i) ist durch die Nachbarschaft $\Gamma(C_i)$ gegeben. Wenn $|\Gamma(C_i)|+1 \leq \frac{2^k}{e}$, dann ist $(|\Gamma(C_i)|+1) \cdot e \leq 2^k$. Setzt man $d = \max_i(|\Gamma(C_i)|$, so ist also insbesondere die Ungleichung $(d+1) \cdot e \leq \frac{1}{p}$ und damit alle Voraussetzungen von Satz 1 erfüllt, woraus folgt dass $P(\overline{A_1}, \ldots, \overline{A_M}) > 0$. Dies bedeutet, dass die Wahrscheinlichkeit, alle Klauseln zu erfüllen, echt größer 0 ist, woraus wegen der Modellierung im Abschnitt 2.1 direkt die Erfüllbarkeit von F folgt.

Eine etwas allgemeinere Form des Lovász Local Lemma liefert folgende, weniger intuitive Formulierung:

[1] Die ursprüngliche Variante aus [EL75] unterscheidet sich von dieser Form nur durch Konstanten in der Ungleichung $p \cdot e \cdot (d+1) \leq 1$.

Satz 2 (Lovász Local Lemma, asymmetrische Form)**.** *Seien* $\mathcal{A} = \{A_1, \ldots, A_M\}$ *Ereignisse.* $\Gamma(A_i)$ *bezeichne die Menge*[2] *, sodass* A_i *stochastisch unabhängig von* $\mathcal{A} \setminus (\Gamma(A_i) \cup \{A_i\})$*. Gibt es* $x : \mathcal{A} \to (0,1)$*, so dass*

$$P(A_i) \leq x(A_i) \prod_{B \in \Gamma(A_i)} (1 - x(B)) \quad \forall A_i \in \mathcal{A}$$

Dann ist $P(\overline{A_1}, \ldots, \overline{A_M}) \geq \prod_{A \in \mathcal{A}} (1 - x(A)) > 0.$

Die symmetrische Form folgt aus der asymmetrischen Form, indem man $x(A) = \frac{1}{d+1}$ setzt, denn aus $p \cdot e \cdot (d+1) \leq 1$ folgt mit der Funktionsdarstellung der Eulerschen Zahl e (EZ)[3]:

$$\begin{aligned}
p &\leq \frac{1}{(d+1)} \cdot \frac{1}{e} \overset{\text{EZ}}{\leq} \frac{1}{(d+1)} \cdot \left(1 - \frac{1}{d+1}\right)^d \\
&\leq \frac{1}{(d+1)} \cdot \left(1 - \frac{1}{d+1}\right)^{|\Gamma(A_i)|} \\
&= \frac{1}{(d+1)} \cdot \prod_{A_j \in \Gamma(A_i)} \left(1 - \frac{1}{d+1}\right) \qquad \left(\text{Jetzt } x(A_j) = \frac{1}{d+1}\right) \\
&= x(A_i) \prod_{A_j \in \Gamma(A_i)} (1 - x(A_j))
\end{aligned}$$

Aus den Voraussetzungen für die *symmetrische* Form folgen also auch die Voraussetzungen der *asymmetrischen* Form. Daher folgt die Korrektheit der symmetrische Form direkt aus der asymmetrischen Form.

2.3 Beweis des Lovász Local Lemma

Wegen des Zusammenhangs der symmetrischen und der asymmetrischen Form reicht es aus, lediglich letztere zu beweisen. Der hier vorgestellte Beweis basiert auf [Cha10] und ist dem aus [EL75] vom Charakter her sehr ähnlich.

Beweis von Satz 2. Der Beweis erfolgt induktiv über $|S|$ mit $S \subset \mathcal{A}$, $A_i \notin S$ und der Hypothese $P(A_i \mid \cap_{B \in S} \overline{B}) \leq x(A_i)$. Für $S = \emptyset$ ist $P(A_i \mid \cap_{B \in S} \overline{B}) = P(A_i) \leq x(A_i)$ nach Voraussetzung der asymmetrischen Form, also ist der Induktionsanfang erfüllt.
Induktionsvoraussetzung (IV): Es gelte $P(A_i \mid \cap_{B \in S} \overline{B}) \leq x(A_i)$ für alle $S \subset \mathcal{A}$ mit $A_i \notin S$ und $|S| \leq s$ für ein $s \in \mathbb{N}$.

[2] Die hier verwendeten $\Gamma(A_i)$ sind für den Fall der Erfüllbarkeit Boolescher Formeln konsistent mit der oben definierten Nachbarschaft $\Gamma(C_i)$.

[3] Man beachte, dass sich in der hier verwendeten Form die gewohnte Monotonie der Folge umkehrt.

Zeige nun die Aussage für $|S| = s + 1$ und weiterhin $A_i \notin S$. Setze dazu $S_1 = \Gamma(A_i) \cap S$, sodass A_i stochastisch unabhänging von $S_2 := S \setminus S_1$ ist. Damit gilt dann:

$$P(A_i \mid \bigcap_{B \in S} \overline{B}) = \frac{P(A_i \cap \bigcap_{B \in S} \overline{B})}{P(\bigcap_{B \in S} \overline{B})} \tag{1}$$

$$= \frac{P(A_i \cap \bigcap_{B \in S_1} \overline{B} \mid \bigcap_{B \in S_2} \overline{B})}{P(\bigcap_{B \in S_1} \overline{B} \mid \bigcap_{B \in S_2} \overline{B})} \tag{2}$$

Der Zähler lässt sich dabei wegen der stochastischen Unabhängigkeit abschätzen mittels: $P(A_i \cap \bigcap_{B \in S_1} \overline{B} \mid \bigcap_{B \in S_2} \overline{B}) \leq P(A_i \mid \bigcap_{B \in S_2} \overline{B}) = P(A_i) \leq x(A_i) \prod_{B \in \Gamma(A_i)} (1 - x(B))$. Für $S_1 = \{B_{i1}, \ldots, B_{il}\}$ lässt sich Nenner hingegen mithilfe einer Variante der Multiplikationsformel (MPF) umformen und abschätzen, indem iterativ die $\cap$-Schreibweise mit bedingten Wahrscheinlichkeiten aufgelöst wird:

$$\frac{P(\bigcap_{B \in S_1} \overline{B} \cap \bigcap_{B \in S_2} \overline{B})}{P(\bigcap_{B \in S_2} \overline{B})} \overset{\text{MPF}}{=} \prod_{t=1}^{l} P(\overline{B_{it}} \mid \bigcap_{j=t+1}^{s} \overline{B_{ij}}) \overset{\text{IV}}{\geq} \prod_{B \in S_1} (1 - x(B))$$

Mit beiden Abschätzungen zusammen folgt der Induktionsschritt, nämlich $P(A_i \mid \cap_{B \in S} \overline{B}) \leq x(A_i)$. Das Local Lemma folgt nun direkt, da hiermit gilt:

$$\begin{aligned} P(\overline{A_i} \mid \bigcap_{B \in S} \overline{B}) &\geq 1 - x(A_i) \\ \Rightarrow P(\overline{A_i} \cap \bigcap_{B \in S} \overline{B}) &\geq (1 - x(A_i)) P(\bigcap_{B \in S} \overline{B}) \\ \overset{\text{rekursiv}}{\geq} \ldots &\geq (1 - x(A_i)) \prod_{B \in S} (1 - x(B)) \end{aligned}$$

Insbesondere ist durch die letzte Rechnung erst Gleichung 1 gerechtfertigt, da stets $P(\cap_{B \in S} \overline{B}) > 0$ bleibt, sobald die Kardinalität von S um eins erhöht wird. Erst hiermit ist gesichert, dass dort (in Gleichung 1) die bedingte Wahrscheinlichkeit überhaupt existiert. □

3 Algorithmus

3.1 Formulierung

Die Grundidee des Algorithmus ist leicht verständlich. Es wird solange – falls vorhanden – eine unerfüllte Klausel ausgewählt und deren Variablen neu „ausgewürfelt“, bis alle Klauseln erfüllt sind. Der Algorithmus in Pseudo-Code:

Algorithmus 1: sequentiellesLLL

Data : Boolesche Formel F in k-KNF
Result : Erfüllende Belegung α

$\alpha \leftarrow$ zufällige Belegung für Variablen von F
while *vlt*$(F, \alpha) \neq \emptyset$ **do**
  $C \leftarrow$ beliebige Klausel aus vlt(F, α);
  $\alpha \leftarrow \alpha$ mit zufälliger neuer Belegung von vbl(C)
end
return α

Zu beachten ist, dass der Algorithmus in dieser Formulierung keineswegs terminieren muss. Es wird jedoch in der Analyse gezeigt, dass unter der Voraussetzung des Lovász Local Lemma dieser Algorithmus *erwartet* in polynomieller Zeit bezüglich $|F|$ (Anzahl Klauseln) mit einer erfüllenden Belegung terminiert.

3.2 Analyse

Im Folgenden soll gezeigt werden, dass Algorithmus 1 erwartet polynomielle Laufzeit hat. Die Grundlagen der hiesigen Analyse stammen hauptsächlich aus [MT10].

Satz 3. *Für eine Menge $\mathcal{A} = \{A_1, \ldots, A_m\}$ von Ereignissen zu einer k-KNF-Formel F (wie oben) gebe es eine Funktion $x : \mathcal{A} \to (0, 1)$, sodass*

$$P(A_i) \leq x(A_i) \prod_{B \in \Gamma(A_i)} (1 - x(B)).$$

Dann wird (für Algorithmus 1) jede Klausel aus F erwartet $\frac{x(A)}{(1-x(A))}$ mal zur Neubelegung ausgewählt, bis der Algorithmus mit einer erfüllender Belegung für F terminiert.

Die Grundidee des Beweises ist folgende: Man zähle, wie oft jede Klausel C_i neu gewürfelt wird. Dies definiert eine Zufallsvariable, für die gezeigt werden muss, dass ihr Erwartungswert existiert und endlich ist.

Damit im Beweis erscheinenden Objekte wohldefiniert sind, nehmen wir an, dass das Auswahlverfahren, mit dem im Algorithmus die nächste nicht-erfüllte Klausel bestimmt wird, beliebig aber fest ist (egal ob es sich um ein randomisertes oder deterministisches Verfahren handelt). Darüberhinaus:

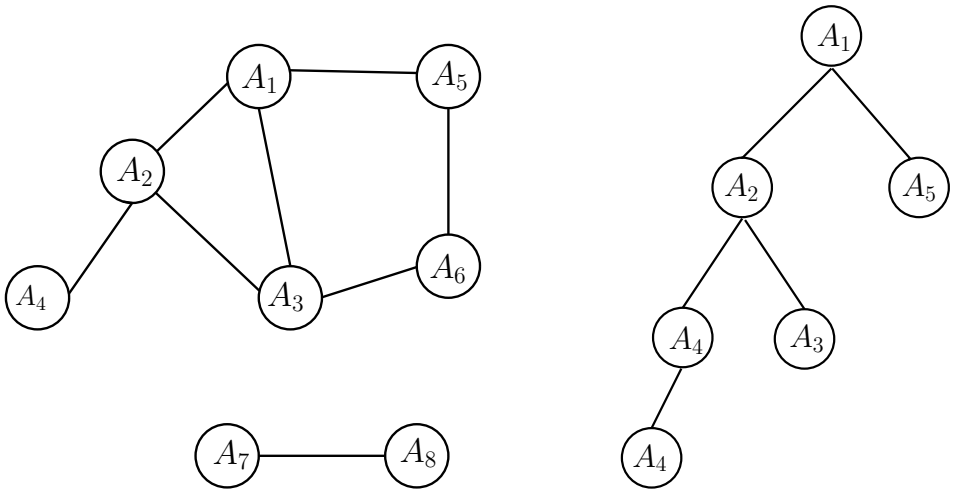
Definition 1. *Ein **Log** sei eine Abbildung $\mathcal{C} : \mathbb{N} \to \mathcal{A}$, die repräsentiert, welche Klausel (bzw. welches Ereignis) der Algorithmus im Schritt t zur Neubelegung ausgewählt hat.*

3.2.1 Konzept von witness trees

Definition 2. *Ein* ***witness tree*** $\tau = (T, \sigma_\tau)$ *sei ein endlicher Baum* T *mit Knotenbeschriftung* $\sigma_\tau : V(T) \to \mathcal{A}$*, sodass die Kinder eines jeden Knoten* $v \in V(T)$ *nur mit Elementen aus* $\Gamma^+(\sigma_\tau(v))$ *beschriftet sind.*
Abkürzend sei: $[v] := \sigma_\tau(v) \quad$ *(für* $v \in V(T)$*) und* $V(\tau) := V(T)$

Man bezeichnet einen witness tree τ als *zulässig*, wenn für $u \in V(\tau)$ alle Kinder von u *unterschiedliche* Beschriftungen haben.

Zu einem gegebenen Log $\mathcal{C}$ kann man in natürlicher Weise einen witness tree $\tau_\mathcal{C}(t)$ zum Zeitpunkt t betrachten, indem man zunächst einen Wurzelknoten erzeugt, diesen mit $\mathcal{C}(t)$ beschriftet und dann $\mathcal{C}$ iterativ rückwärts durchläuft (Verwende hierzu Laufindex i, der bei $t-1$ beginnt und mit 1 endet): Falls sich im bisher konstruierten Baum Knoten befinden, die mit einem (inklusiven) Nachbarn von $\mathcal{C}(i)$ (d.h. einem Element aus $\Gamma^+(\mathcal{C}(i))$) beschriftet sind, so füge einen Knoten mit Beschriftung $\mathcal{C}(i)$ als tiefstes Kind aller dieser „Nachbar"-Knoten in den Baum ein. Andernfalls (kein Knoten ist mit Nachbarn von $\mathcal{C}(i)$ beschriftet) ignoriere $\mathcal{C}(i)$. Die Abbildung 1 zeigt ein Beispiel zu Log und Konstruktion eines witness tree. Gegeben ist ein fester Log mit einem Graphen in Abbildung 1a, der die Abhängigkeiten (also die Nachbarschaft) der Ereignisse A_i repräsentiert.



(a) Abhängigkeitsgraph (b) witness tree $\tau_\mathcal{C}(8)$

Abbildung 1: Log $\mathcal{C} = (A_4, A_3, A_4, A_7, A_5, A_2, A_6, A_1, \dots)$

In diesem Zusammenhang sagt man τ *tritt auf* in $\mathcal{C}$, wenn sich mit der eben geschilderten Vorgehensweise ein solcher witness tree τ aus $\mathcal{C}$ konstruieren lässt. Das heißt, es gibt ein t aus $\mathbb{N}$, sodass $\tau = \tau_\mathcal{C}(t)$.

Da die erwartete Laufzeit des Algorithmus 1 elegant mithilfe von witness trees bewiesen werden kann, benötigt man für den weiteren Beweis zunächst

noch ein Lemma, das den Zusammenhang von Logs und witness trees mit der Vorgehensweise des Algorithmus erklärt.

Lemma 1. *Sei τ ein fester witness tree und $\mathcal{C}$ ein (zufälliger) Log, der durch den Algorithmus 1 generiert wurde. Dann gilt:*

(i) Wenn τ in $\mathcal{C}$ auftritt, dann ist τ zulässig.

(ii) Die Wahrscheinlichkeit dass τ in $\mathcal{C}$ auftritt ist $\leq \prod_{v \in V(\tau)} P([v])$

Beweisskizze des Lemmas. (i) Folgt direkt durch Konstruktion von $\tau = \tau_{\mathcal{C}}(t)$ (für ein t aus $\mathbb{N}$): Angenommen τ sei nicht zulässig, das heißt, es gibt einen Knoten v in τ, der mindestens 2 Kinder u_1, u_2 mit gleicher Beschriftung ($[u_1] = [u_2]$) besitzt. Wurde o.B.d.A. u_1 zuerst eingefügt, so hat $[u_2]$ beim Einfügen als tiefsten Nachbarn wenigstens $[u_1]$, weshalb u_2 mindestens als Kind von u_1 eingefügt werden muss (Widerspruch zur Annahme).

Zu (ii): Geht man τ schichtenweise aufwärts durch, so entspricht dies ungefähr der Reihenfolge der vom Algorithmus ausgewählten Ereignisse. Ereignisse gleicher Schicht sind unabhängig (nach Konstruktion, wie in Teil (i)). Daher ist die Wahrscheinlichkeit für deren gemeinsames Auftreten $\leq \prod_{v \in \text{ Schicht } i} P([v])$. Neuwürfeln von vbl($[v]$) in Schicht i „entfernt“ dabei gewissermaßen die Abhängigkeit zu Ereignissen aus Schicht $i-1$, denn Schicht $i-1$ erhält dadurch Zufallsdaten, die noch nie zuvor „betrachtet“ wurden, also vom Charakter unabhängig gleichverteilte Zufallsbits für alle Variablen. Dies erklärt sich induktiv, denn die erste Belegung α würfelt alle Zufallsvariablen (bzw. -Bits) zufällig gleichverteilt aus, wodurch zunächst für alle Ereignisse ungenutzte, „frische“ Zufallsdaten zur Verfügung stehen. Falls nun ein Ereignis A_i eintritt (Klausel unerfüllt) und dieses vom Algorithmus ausgewählt wird, so wird durch das Neuwürfeln von vbl(A_i) genau der Teil an Zufallsvariablen neu belegt, der A_i mit seinen Nachbarn $\Gamma^+(A_i)$ (bzw. $\Gamma^+(C_i)$) verbindet, wodurch diese Nachbarn nach der Neubelegung vom Charakter her ungenutzte Zufallsbits sehen. Damit ist die Auftrittswahrscheinlichkeit $\leq \left(\prod_{v \in \text{ Schicht } i} P([v]) \cdot \prod_{v \in \text{ Schicht } i-1} P([v]) \right)$. Iterativ über alle Schichten folgt damit die Behauptung. □

3.2.2 Galton Watson Prozess (zufälliger Verzweigungsprozess)

Als letztes Hilfsmittel für den Beweis der Laufzeit benötigt man das Konzept des sog. **Galton-Watson (Verzweigungs-) Prozesses**. Damit kann ein zufälliger zulässiger witness tree produziert werden, indem zunächst eine Wurzel mit Beschriftung A erzeugt wird und dann für jedes $B \in \Gamma^+(A)$ ein Kindknoten (beschriftet mit B) mit Wahrscheinlichkeit $x(B)$ erzeugt

wird. Danach werden für jedes erzeugte Kind rekursiv in der gleichen Weise Kindknoten erzeugt und für deren erzeugte Kinder ebenfalls usw. Dabei gilt folgendes Lemma.

Lemma 2. *Sei τ fester, zulässiger witness tree mit Wurzel A. Dann ist die Wahrscheinlichkeit, dass ein Galton-Watson Prozess zu genau τ führt:*

$$p_\tau = \frac{1 - x(A)}{x(A)} \prod_{v \in V(\tau)} x'([v])$$

wobei $x'(B) := x(B) \prod_{C \in \Gamma(B)} (1 - x(C))$

Beweis. Sei $W_v \subseteq \Gamma^+([v])$ Menge der Beschriftungen, die im Baum *nicht* als Kinder von v auftauchen. Dann gilt offenbar:

$$\begin{aligned}
p_\tau &= \frac{1}{x(A)} \prod_{v \in V(\tau)} \left(x([v]) \prod_{u \in W_v} (1 - x([u])) \right) \\
&= \frac{1}{x(A)} \prod_{v \in V(\tau)} \left(x([v]) \prod_{u \in W_v} (1 - x([u])) \prod_{u \in \Gamma^+([v]) \setminus W_v} \frac{(1 - x([u]))}{(1 - x([u]))} \right) \\
&= \frac{1}{x(A)} \prod_{v \in V(\tau)} \left(x([v]) \prod_{u \in \Gamma^+([v])} (1 - x([u])) \prod_{u \in \Gamma^+([v]) \setminus W_v} \frac{1}{(1 - x([u]))} \right)
\end{aligned}$$

Tatsächlich erzeugte Kinder von v (im Produkt ganz rechts) werden „später“ im äußeren Produkt nochmal betrachtet, wodurch sich die Formel vereinfacht zu:

$$\begin{aligned}
p_\tau &= \frac{1 - x(A)}{x(A)} \prod_{v \in V(\tau)} \left(\frac{x([v])}{1 - x([v])} \prod_{u \in \Gamma^+([v])} (1 - x([u])) \right) \\
&= \frac{1 - x(A)}{x(A)} \prod_{v \in V(\tau)} x'([v])
\end{aligned}$$

□

3.2.3 Beweis der Laufzeit

Man definiert nun eine Zufallsvariable N_A als die Anzahl an Vorkommen von A im Log $\mathcal{C}$. Das heißt N_A zählt, wie oft vbl(A) neu gewürfelt werden, wobei N_A auch zugleich die Anzahl der zulässigen witness trees in $\mathcal{C}$ mit Wurzel A ist. Indem nun die Endlichkeit des Erwartungswertes von N_A gezeigt wird,

kann kann Satz 3 bewiesen werden. Sei dafür $\mathcal{T}_A$ die Menge aller zulässigen witness trees mit Wurzel A. Dann gilt:

$$\begin{aligned}
\mathbb{E}(N_A) &= \sum_{\tau \in \mathcal{T}_A} \mathrm{P}(\tau \text{ tritt auf in Log } \mathcal{C}) \\
&\overset{\text{Lem 1}}{\leq} \sum_{\tau \in \mathcal{T}_A} \prod_{v \in V(\tau)} P([v]) \\
&\overset{\text{Vor.}}{\leq} \sum_{\tau \in \mathcal{T}_A} \prod_{v \in V(\tau)} x'([v]) \\
&\overset{\text{Lem 2}}{=} \frac{x(A)}{1-x(A)} \sum_{\tau \in \mathcal{T}_A} p_\tau \leq \frac{x(A)}{1-x(A)}
\end{aligned}$$

Damit ist Satz 3 bewiesen. Falls $x(A)$ konstant ist (für m: Anzahl Ereignisse bzw. Klauseln), gilt also:

$$\mathbb{E}(\sum_{A \in \mathcal{A}} N_A) \leq m \cdot \frac{x(A)}{1-x(A)}$$

Insbesondere wenn $x(A) = \frac{1}{d+1}$ (Lovász Local Lemma, symmetrische Form) vereinfacht sich die Formel zu:

$$\mathbb{E}(\sum_{A \in \mathcal{A}} N_A) \leq \frac{m}{d}$$

Daraus folgt, dass im „Mittel“ maximal $\frac{m}{d}$ Klauseln neu gewürfelt werden müssen, falls die symmetrische Form erfüllt ist, wofür freilich weiterhin die Ungleichung $p \cdot e \cdot (d+1) \leq 1$ gelten muss. Falls etwa d groß ist, müssen nur sehr wenige Klauseln erneut ausgewürfelt werden, da in diesem Fall p wegen der Ungleichung entsprechend klein sein muss. Das heißt, die Wahrscheinlichkeit, dass Klauseln in der Startbelegung unerfüllt sind, sinkt und somit müssen erwartet auch weniger Klauseln erneut ausgewürfelt werden.

3.3 Ausblick

In diesem Abschnitt soll noch kurz eine parallele Version sowie eine Verallgemeinerung des Algorithmus vorgestellt werden.

3.3.1 Parallele Version

Der Algorithmus lässt sich parallelisieren. Hierfür muss in jedem Schritt (statt nur eines Ereignisses) *parallel* eine Menge inklusionsmaximaler unabhängiger Mengen von Ereignissen ausgewürfelt werden. Es folgt der Algo-

rithmus in Pseudo-Code:

Algorithmus 2: parallelesLLL

Data : Boolesche Formel F in k-KNF
Result : Erfüllende Belegung α

$\alpha \leftarrow$ zufällige Belegung für Variablen von F
while $vlt(F, \alpha) \neq \emptyset$ **do**
  $S \leftarrow$ inklusionsmaximale unabhängige Menge im Abhängigkeitsgraph von $\text{vlt}(F, \alpha)$
  $\alpha \leftarrow \alpha$ mit parallel berechneter zufälliger neuer Belegung von $\text{vbl}(S)$
end
return α

Der schwierigste Teil bezüglich der Effizienz dieses Algorithmus ist eine Implementierung zur geschickten Auswahl einer inklusionsmaximalen unabhängigen Menge. Darüberhinaus kann die Analyse auf den sequentiellen Fall zurückgeführt werden, siehe hierfür [MT10].

3.3.2 Verallgemeinerung

Das vorgestellte Konzept lässt sich auf allgemeinere Probleme übertragen. Dazu müssen $\mathcal{X} = \{X_1, \ldots, X_n\}$ Zufallsvariablen mit *endlichem* Wertebereich sein und $\mathcal{A} = \{A_1, \ldots, A_m\}$ Ereignisse, die eindeutig durch Zufallsvariablen aus $\mathcal{X}$ bestimmt sind. Ist für diese Ereignisse das Lovász Local Lemma erfüllt, so ist auch hier der Algorithmus anwendbar, indem man die Zufallsvariablen Werte gemäß ihrer zugrundeliegenden Wahrscheinlichkeitsverteilung annehmen läßt. Der Beweis der Laufzeit bleibt dabei im Großen und Ganzen erhalten. Wegen der besseren Anschaulichkeit und der zentralen Rolle des SAT-Problems in der Informatik wurde jedoch in dieser Ausarbeitung das Hauptaugenmerk auf den Spezialfall der Erfüllbarkeit Boolescher Formeln in k-KNF gelegt.

Literatur

[Cha10] Charika, Moses: *(Vorlesungsskript) (COS 598B) Algorithms & Complexity, Lecture 1*. www.cs.princeton.edu/courses/archive/spring10/cos598B/lec1.pdf, 2010. 3

[EL75] Erdős, Paul und László Lovász: *Infinite and Finite Sets*, Band 2, Kapitel Problems and results on 3-chromatic hypergraphs and some related questions, Seiten 609–627. North-Holland, 1975. 1, 2, 3

[Mos09] Moser, Robin A.: *A constructive proof of the Lovász Local Lemma.* Proc. 41st Ann. ACM Symp. on Theory of Computing, Seiten 343–350, 2009. 1

[MT10] Moser, Robin A. und Gábor Tardos: *A constructive proof of the general Lovász Local Lemma.* Journal of the ACM, 57(2):1–15, 2010. 1, 5, 10