Random Walks

Dumbiri Gbenoba
234 342

Zusammenfassung

In einem Fluss sind n Steine in einem Kreis angeordnet. Auf einem der Steine sitz ein Frosch. Dieser Frosch springt mit der Wahrscheinlichkeit von $\frac{1}{3}$ einen Stein in Richtung Uhrzeigersinn, mit Wahrscheinlichkeit von $\frac{1}{3}$ einen Stein in Richtung gegen den Uhrzeigersinn und springt mit einer Wahrscheinlichkeit von $\frac{1}{3}$ auf der Stelle. Das Problem besteht nun darin, herauszufinden, wo sich der Frosch wohl nach t Sprüngen befinden wird. Dies ist eines der populärsten Beispiele, mit denen das Konzept der *Random Walks* verdeutlicht werden kann. Diese Seminararbeit stellt ein paar Wahrscheinlichkeitsprobleme vor und zeigt, wie diese mit Hilfe der Modellierung von Random Walks gelöst werden können.

Inhaltsverzeichnis

1 Einführung

1.1 Random Walk

Random Walks (deutsche Bezeichnungen wie *Zufallsbewegung* oder *Irrfahrt* haben sich nicht durchsetzen können) bilden eine wichtige Klasse von stochastischen Prozessen. Der Random Walk kann als ein Bernoulli-Prozess interpretiert werden, das heißt eine Folge von unabhängigen Bernoulli-Versuchen. Dies führt zu einer Binomialverteilung.

Beispiel Eine beliebte Veranschaulichung lautet ungefähr wie folgt: Ein desorientierter Fußgänger läuft in einer Gasse mit einer Wahrscheinlichkeit p einen Schritt nach vorne, mit einer Wahrscheinlichkeit $q = 1 - p$ einen Schritt zurück. Die Wahrscheinlichkeit, dass er nach n Schritten eine Strecke X zurückgelegt hat, beträgt dann

$$Prob(X = -n + 2k) = \binom{n}{k} p^k q^{n-k}$$

1.2 Inhalt dieser Arbeit

In dieser Seminararbeit werden verschiedene Möglichkeiten der Modellierung mit Random Walks zur Analyse von Wahrscheinlichkeitsproblemen vorgestellt. Im Abschnitt 2 wird ein Satz von Papadimitriou vorgestellt und mit Hilfe der Modellierung eines Random Walks bewiesen. In dem hier vorgestellten Beweis des Satzes von Conway (Abschnitt 3) wird ein nicht-transitives Spiel über einem Random Walk näher betrachtet und analysiert. Im Abschnitt 4 werden zwei Suchprobleme vorgestellt und mit Hilfe einer interessanten Anwendung (Razborov, Widgerson und Yao, 1997) von Random Walks analysiert.

2 Erfüllbarkeit von 2-KNF

2.1 Einführung

Eine k-KNF (konjunktive Normalform) ist eine „Und"-Verknüpfung beliebig vieler *Klauseln*, die jeweils eine „Oder"-Verknüpfung von k *Literalen* sind. Dabei ist ein Literal eine boolsche Variable x_i oder deren Negation $\bar{x_i}$. Für eine gegebene 2-KNF F suchen wir nun eine erfüllende Belegung, so dass alle Klauseln von F erfüllt sind.

Beispiel Sei $F = (x_1 \vee \bar{x_2}) \wedge (x_2 \vee x_3) \wedge (\bar{x_1} \vee \bar{x_3})$, dann ist $a = (1, 1, 0)$ eine erfüllende Belegung für F.

2.2 Satz von Papadimitriou

Die Frage, die sich jetzt für eine beliebige, erfüllbare 2-KNF F stellt, ist: „Wie schnell können wir eine erfüllende Belegung finden?". Ein Algorithmus, der dieser Frage auf den Grund geht, stammt von Papadimitriou (1991) und wird nun folgend vorgestellt.

Algorithmus Man starte mit einer beliebigen Belegung $a \in \{0, 1\}^n$ der Literale der 2-KNF F.

1. Solange eine Klausel von F nicht erfüllt ist, verändern wir a wie folgt: Wir wählen aus einer beliebigen unerfüllten Klausel eines der zwei Literale aus und komplementieren in der Belegung a dessen Wert.
2. Falls nun alle Klauseln erfüllt sind, sind wir fertig und haben eine erfüllbare Belegung gefunden, nämlich a. Andernfalls gehen wir wieder zurück zu Schritt 1.

Satz, Papadimitriou (1991) Sei F eine erfüllbare 2-KNF mit n Variablen. Dann findet der soeben beschriebene Algorithmus mit einer Wahrscheinlichkeit von mindestens $\frac{1}{2}$ eine erfüllende Belegung der Variablen in $2n^2$ Schritten.

Beweis Sei $a \in \{0, 1\}^n$ eine erfüllende Belegung für F und bezeichnen wir die Belegung der Werte in a als die „korrekten Werte". Sei nun $i \in \{0, 1, .., n\}$ die

Anzahl der nicht korrekten Werte für eine beliebige Belegung. Bei jedem Schritt des Algorithmus ändert sich die Anzahl der korrekten Werte um 1. Das gleiche gilt natürlich auch für die Anzahl der nicht korrekten Werte. Das bedeutet, dass ausgehend von $0 < i < n$ nach einem Schritt des Algorithmus die Anzahl der nicht korrekten Werte entweder $i - 1$ oder $i + 1$ ist (Abbildung 1). Es führt i also einen Random Walk zwischen 0 und n aus.

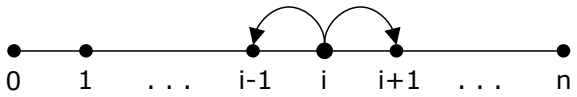

Abbildung 1: Veränderung der inkorrekten Werte in a bei einem Schritt des Algorithmus

Definieren wir nun eine Funktion t so, dass $t(i)$ die erwartete Anzahl von Schritten angibt um ausgehend von einer Belegung mit i inkorrekten Werten zu einer korrekten Belegung (0 inkorrekte Werte) zu kommen. Unser Ziel ist nun zu zeigen, dass

$$t(i) \leq n^2, \forall i \in \{0, 1, .., n\}\,.$$

Da bei n inkorrekten Werten bei einem Schritt von t kein inkorrekter Wert mehr hinzukommen kann, also auf jeden Fall danach ein inkorrekter Wert weniger existiert, und da der Prozess beendet ist, sobald es 0 inkorrekte Werte gibt (der Prozess kann auch eher terminieren, falls eine andere erfüllende Belegung als a erreicht wird), gilt $t(n) \leq t(n-1) + 1$ und $t(0) = 0$. Im allgemeinen gilt

$$t(i) = p_{i,i-1}(1 + t(i-1)) + p_{i,i+1}(1 + t(i+1)),$$

wobei $p_{i,j}$ die Wahrscheinlichkeit angibt, mit welcher der Algorithmus ausgehend von i inkorrekten Werten in einem Schritt auf $j \in \{i-1, i+1\}$ inkorrekte Werte kommt.

In einer unerfüllten Klausel einer 2-KNF hat mindestens eines der zwei Literale einen inkorrekten Wert. Da wir zufällig einen der Werte geändert haben, haben wir mit einer Wahrscheinlichkeit von mindestens $\frac{1}{2}$ die Anzahl der Variablen mit inkorrekten Werten verringert. Die Veränderung der Anzahl i der Variablen mit inkorrekten Werten stellt einen Random Walk dar, wobei $p_{i,i-1} \geq \frac{1}{2}$. Daraus folgt, dass

$$t(i) \leq \frac{t(i-1) + t(i+1)}{2} + 1.$$

Ersetzen wir nun die obigen Ungleichungen durch Gleichungen, so erhalten wir als obere Grenze für t für $0 \leq i \leq n$ die Funktion x

$$x(0) = 0, \quad x(i) = \frac{x(i-1) + x(i+1)}{2} + 1, \quad x(n) = x(n-1) + 1.$$

Durch Einsetzen erhält man $x(1) = 2n - 1$, $x(2) = 4n - 4$ und im Allgemeinen $x(i) = 2in - i^2$. Und damit gilt wie gewünscht

$$t(i) \leq x(i) \leq x(n) = n^2.$$

Nach der Markov-Ungleichung kann eine Zufallsvariable einen Wert größer als das Doppelte ihres Erwartungswerts nur mit einer Wahrscheinlichkeit $< \frac{1}{2}$ annehmen. Daher ist die Wahrscheinlichkeit, dass der Algorithmus bei einer Belegung mit i inkorrekten Werten mehr als $2t(i)$ Schritte braucht, bis er 0 inkorrekte Werte erreicht, kleiner als $\frac{1}{2}$. Daraus folgt, dass der Algorithmus mit einer Wahrscheinlichkeit von mindestens $\frac{1}{2}$ in höchstens $2n^2$ Schritten terminiert.

3 Satz von Conway

3.1 Einführung

Nehmen wir einen Random Walk $w \in \{0,1\}^*$ über einen unendlichen binären Baum an, wobei in jedem Knoten jeweils die 0-Kante oder die 1-Kante mit einer Wahrscheinlichkeit von $\frac{1}{2}$ gewählt wird. Die Spieler A und B wählen nun beide nacheinander je ein Wort a bzw. b der Länge k, natürlich ohne w zu kennen. Gewonnen hat der Spieler, dessen Wort als erstes in w vorkommt. Auf den ersten Blick sieht es so aus, als wenn das Spiel unabhängig von der Wahl der Wörter fair sei. Das folgende einfache Beispiel zeigt jedoch, dass einige Wörter mit höherer Wahrscheinlichkeit eher in w vorkommen als andere. Bezeichnen wir das Chancenverhältnis zwischen den Wörtern b und a mit $W(a,b)$.

Beispiel

- Sei $k := 2$
- Spieler A wählt Wort $a = 00$
- B wählt Wort $b = 10$

Durch Werfen einer fairen Münze simulieren wir nun den zufälligen Pfad (Random Walk) w über einen unendlichen binären Baum. Nach zweimaligem Werfen der Münze erhalten wir $w = 00 = a$ (A gewinnt), $w = 10 = b$ (B gewinnt), $w = 01$ oder $w = 11$. Betrachtet man nun die beiden letzten Fälle, so sieht man, dass A keine Möglichkeit mehr hat, das Spiel zu gewinnen, da beim nächsten Auftreten einer 0 Spieler B gewonnen hat. Bei der Wahl dieser Wörter a und b stehen die Chancen $3 : 1$, dass Spieler A gewinnt. Wählt jedoch Spieler B das Wort $b = 01$, so haben beide Spieler die gleichen Chancen zu gewinnen, das Spiel wäre fair.

Wie gerade gezeigt wurde sind einige Wörter „stärker“ als andere. Man könnte annehmen, dass Spieler A einfach das „stärkste“ Wort a wählen könnte und somit seine Chancen das Spiel zu gewinnen höher sind als die von Spieler B, der als zweites ein Wort b wählt. Jedoch handelt es sich für Wörter der Länge $k \geq 3$ um ein nicht-transitives Spiel: Egal welches Wort Spieler A wählt, es existiert ein Wort b, welches stärker ist.

3.2 Definition

Für zwei Wörter a und b ($|a| = |b|$) sei

$$H_{ab} := \{\, x \mid a = x \cdot y, b = y \cdot z, |y| > 0, z \in \{0,1\}^*\} \,.$$

H_{ab} ist also die Menge aller Wörter x, so dass $a = xy$ für ein nichtleeres Prefix y von b. Sei nun X eine Menge von Wörtern über dem Alphabet $\{0,1\}$, so ist $P(X) = \sum_{x \in X} 2^{-|x|}$ die Wahrscheinlichkeit, dass durch den fairen Münzwurf ein Wort aus X erzeugt wird. Definiere

$$P_{ab} := P(H_{ab})$$

3.3 Satz, Conway

$\forall$ Wörter a, b über dem Alphabet $\{0,1\}$ mit Länge $k \geq 1$ gilt

$$W(a,b) = \frac{P_{aa} - P_{ab}}{P_{bb} - P_{ba}}$$

Beweis, Prevzner (1993) Ein Wort s bezeichnen wir als A-Sieg, wenn es das Wort b nicht enthält und das Wort a Suffix von s ist (s endet mit a). Wir definieren die Menge $S_a := \{\, s \mid s \cdot a$ ist A-Sieg$\}$. B-Sieg und S_b definieren wir analog. Wir können nun die Wahrscheinlichkeit, dass Spieler A gewinnt durch, $2^{-k}P(S_a)$ ausdrücken. Seien X und Y zwei Mengen über Wörtern, so definieren wir die Menge aller Konkatenationen der Elemente aus X mit denen aus Y als $X \cdot Y := \{\, x \cdot y \mid x \in X, y \in Y\}$.

Die Idee ist nun zu zeigen, dass $S = \{\, s \mid s$ enthält weder a noch $b\}$ dargestellt werden kann durch

$$S = (S_a \cdot H_{aa}) \cup (S_b \cdot H_{ba}) \quad {}^{(*)}$$

und durch

$$S = (S_b \cdot H_{bb}) \cup (S_a \cdot H_{ab}) \quad {}^{(**)}$$

$^{(*)}$ Jedes Wort $w = s \cdot a$, mit $s \in S$ ist entweder ein A-Sieg, falls a vor b oder b nicht in w vorkommt (Abbildung 2). Andernfalls, also wenn b vor a auftritt (Abbildung 3), ist w ein B-Sieg.

$^{(**)}$ kann analog zu $^{(*)}$ gezeigt werden durch vertauschen von a und b: Jedes Wort $w = s \cdot b,\ s \in S$ ist entweder ein B-Sieg, falls b vor a oder a nicht in w vorkommt. Andernfalls, also wenn a vor b auftritt, ist w ein A-Sieg.

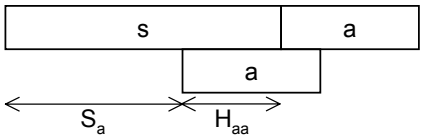

Abbildung 2: Darstellung der Wörter aus $S \cdot \{a\}$, wobei $s \in S_a \cdot H_{aa}$

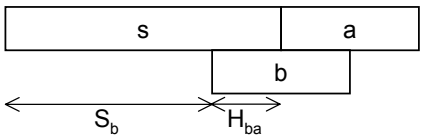

Abbildung 3: Darstellung der Wörter aus $S \cdot \{a\}$, wobei $s \in S_b \cdot H_{ba}$

Aus $^{(*)}$ folgt

$$P(S) = P(S_a \cdot H_{aa}) + P(S_b \cdot H_{ba}) = P(S_a) \cdot P(H_{aa}) + P(S_b) \cdot P(H_{ba}),$$

und aus $^{(**)}$ folgt

$$P(S) = P(S_b \cdot H_{bb}) + P(S_a \cdot H_{ab}) = P(S_b) \cdot P(H_{bb}) + P(S_a) \cdot P(H_{ab}).$$

$$\Rightarrow W(a,b) = \frac{P(S_b)}{P(S_a)} = \frac{P_{aa} - P_{ab}}{P_{bb} - P_{ba}}$$

4 Random Walks und Suchprobleme

4.1 Einführung

In diesem Kapitel wird eine interessante Anwendung von Random Walks für die Analyse von Suchproblemen vorgestellt. Sie stammt von Razborov, Widgerson und Yao (1997).

Definition Sei $G = (V, E)$ ein gerichteter zykelfreier Graph. Der Grad $d(v)$ eines Knotens v ist definiert als die Anzahl der ausgehenden Kanten von v. Ein in Knoten v_0 beginnender Random Walk der Länge k in G ist eine zufällige Folge von Knoten $p = (v_0, v_1, \ldots, v_k)$, wobei jeder Knoten v_{i+1} zufällig aus den $d(v_i)$ Nachbarn von v_i mit einer Wahrscheinlichkeit von $\frac{1}{d(v_i)}$ gewählt wird. Wir sagen **p** *besucht* eine Kante v, falls ein i existiert mit $v = v_i$.

Nehmen wir nun einen solchen gerichteten zykelfreien Graphen $G = (V, E)$ mit einer Wurzel w. Zu jedem Knoten v ist uns der jeweilige Grad $d(v)$ bekannt. Wollen wir nun eine untere Grenze für die Anzahl der Knoten $|V|$ in dem Graphen angeben, so können wir nach Razborov, Widgerson und Yao wie folgt vorgehen: Wir wählen eine geeignete Untermenge $U \subseteq V$ der Knoten, so dass jeder Pfad ausgehend von der Wurzel w ...

$^{(*)}$ mindestens einen Knoten aus U besucht und
$^{(**)}$ k Knoten mit einem Grad von mindestens d durchläuft, bevor er U besucht.

Aus $^{(*)}$ folgt, dass ein Random Walk mindestens einen Knoten aus U mit Wahrscheinlichkeit 1 besuchen wird. Und aus $^{(**)}$ folgt, dass die Wahrscheinlichkeit, dass der Random Walk eine ausgewählte Kante aus U besucht höchstens d^{-k} beträgt. Damit gilt $|U| \geq d^k$. Diese Idee wird in den folgenden Kapiteln anhand konkreter Situationen (4.2, 4.3) genauer betrachtet.

4.2 Lange Wörter über einem kleinen Alphabet

Betrachten wir das folgende Problem der Wortsuche $W(n, m)$ über einem Alphabet A (Menge von Buchstaben) mit $|A| = n$. Ein Wort x wird definiert als eine Folge der Länge m von Buchstaben aus A, also $x = (x_1, \ldots, x_m) \in A^m$. Wir nehmen an, dass $m \geq n + 1$. Nach dem Schubfachprinzip folgt, dass jedes solche Wort zwei gleiche Buchstaben enthalten muss. Das Problem besteht nun darin, in

einem gegebenen Wort zwei gleiche Buchstaben zu finden. Dabei sind lediglich Tests der Form $x_i = a$ ($1 \geq i \geq n$, $a \in A$) zulässig. Ohne weitere Beschränkungen wäre die Aufgabe trivial: Man könnte einfach alle $n\binom{m}{2}$ Vergleiche der Form $x_i = a, x_j = a (\forall i, j : i \neq j$ und $a \in A)$ durchführen. In diesem Fall haben wir jedoch die Beschränkung, dass die Buchstaben x_i nach einem Test verschwinden bzw. durch ein Symbol $* \notin A$ ersetzt werden.

Übertragen wir das Problem $W(n, m)$ auf einen gerichteten zykelfreien Graphen $G = (V, E)$ mit Wurzel w. Jeder Knoten (bis auf Senken) $v \in V$ wird mit einer Zahl $i \in \{0, \ldots, n\}$ beschriftet und hat n ausgehende Kanten. Jede dieser ausgehenden Kanten ist jeweils mit einem unterschiedlichen Bezeichner $a \in A$ beschriftet. Senken ($d(v) = 0$) sind mit möglichen Lösungen der Form (i, j, k) beschriftet, wobei $1 \geq i \neq j \geq m$ und $a \in A$. Eine solche Lösung gilt für ein Wort $x = (x_1, \ldots, x_m)$, wenn $x_i = x_j = a$.

Jedes Wort beschreibt einen bestimmten Pfad im Graphen ausgehend von der Wurzel w zu einer Senke: An einem Knoten v, der mit i bezeichnet ist, läuft der Pfad entlang der Kante, die mit $a \in A$ bezeichnet ist, falls gilt $x_i = a$. Der Graph löst das Problem, falls jedes Wort zu einer Senke mit einer gültigen Lösung führt. Da wir das Problem mit der Beschränkung betrachten, dass jedes x_i nur einmal getestet werden darf, darf keiner der Pfade von der Wurzel bis zu einer Senke zwei Knoten mit dergleichen Beschriftung enthalten.[1]

Satz Ein solcher Graph, der das Suchproblem $W(n, m)$ löst, hat mindestens $n^{\Omega(n)}$ Knoten.

Beweis Sei $n \geq 3$ und $G = (V, E)$ ein Graph mit den soeben beschriebenen Eigenschaften, der das Problem $W(n, m)$ löst. Wir definieren $J(v) := \{ a \in A \,|x_i = a$ wird auf jedem Pfad von w nach v zugewiesen für ein $i \in [m]\}$. Ist $e = (v, u)$ eine Kante von v nach u, so gilt, wie man leicht sieht, $|J(u)| \leq |J(v)| + 1$. Wir bezeichnen eine von einem Knoten v ausgehende Kante e, beschriftet mit dem Buchstaben a, als legal, falls $a \notin J(v)$, und andernfalls als illegal. Es ist leicht einzusehen, dass kein Pfad von der Wurzel zu einer Senke existiert, der ausschließlich legale Kanten enthält.

Wir definieren nun einen Random Walk **p** in G, welcher ausgehend von der Wurzel an jedem Knoten eine der dort ausgehenden legalen Kanten zufällig (mit glei-

[1] Das bedeuted nicht, dass in dem Graphen nicht mehrere Knoten mit der gleichen Beschriftung existieren dürfen. Wichtig ist nur, dass auf einzelnen Pfaden von der Wurzel zu einer Senke nicht zwei Knoten die gleiche Beschriftung haben dürfen, da dies dem mehrmaligen Vergleich eines Buchstabens x_i des zu untersuchenden Wortes x entsprechen würde.

chen Wahrscheinlichkeiten) wählt und so zum nächsten Knoten gelangt. Da kein Pfad von der Wurzel bis zur Senke existiert, der ausschließlich legale Kanten enthält, erreicht **p** mit einer Wahrscheinlichkeit von 0 eine Senke und erreicht mit Wahrscheinlichkeit von 1 einen Knoten v an dem $J(v) = A$ gilt. Der Wert von $|J(v)|$ kann sich nach jeder durchlaufenen Kante entweder um 1 erhöhen oder seinen Wert beibehalten. Daraus folgt, dass **p** mit Wahrscheinlichkeit 1 einen Knoten v besuchen wird, an dem gilt $|J(v)| = k$ mit $k := \lceil \frac{|A|}{2} \rceil = \lceil \frac{n}{2} \rceil$. Bezeichnen wir den ersten solchen Knoten in **p** mit **v**. Nun bleibt noch zu zeigen, dass für jeden Knoten v_0 mit $J(v_0) = k$ gilt $Prob(\mathbf{v} = v_0) \leq n^{-\Omega(n)}$.

Sei $J(v_0) = \{a_1, \ldots, a_k\}$, und seien die $i_1, \ldots, i_k$ so gewählt, dass alle Pfade von der Wurzel nach v_0 die Zuweisungen $x_{i_1} = a_1, \ldots, x_{i_k} = a_k$ ($i_1, \ldots, i_k$ paarweise verschieden) vornehmen. Gilt nun $\mathbf{v} = v_0$, so folgt daraus, dass **p** alle Buchstaben $x_{i_1}, \ldots, x_{i_k}$ getestet haben muß, bevor der Knoten v_0 erreicht wird. Dabei muß **p** aus einer Auswahl von mindestens k ausgehenden legalen Kanten immer den „richtigen Test“ $x_{i_\nu} = a_\nu$ ausgewählt haben. $\Rightarrow$ Die Wahrscheinlichkeit an jedem der k Knoten die richtige Kante auszuwählen beträgt $\frac{1}{k}$.

$$\Rightarrow Prob(\mathbf{v} = v_0) \leq \frac{1}{k}^k \leq n^{-\Omega(n)}$$

4.3 Kurze Wörter über einem großen Alphabet

In Abschnitt 4.2 haben wir den Fall betrachtet, dass $m > n$ ist, nun wollen wir den Fall $m < n$ betrachten. Das Alphabet ist in diesem Fall also größer als das betrachtete Wort. Nimmt man diese Situation an, so ist klar, dass in jedem solchen Wort mindestens ein Buchstabe aus dem Alphabet fehlt. Das Suchproblem $W^*(n, m)$ ist in diesem Fall, bei einem gegebenen Wort $x = (x_1, \ldots, x_m)$ herauszufinden, welcher Buchstabe fehlt. Das zugehörige Modell eines Suchgraphen ist genauso definiert wie in Abschnitt 4.2 mit dem Unterschied, dass die Senken nur mit Buchstaben beschriftet sind, da mögliche Lösungen die Form $a \notin x$ haben.

Auch in diesem Abschnitt setzen wir die Einschränkung voraus, dass jeder Buchstabe des Wortes nur einmal gelesen werden kann. Daher darf auch hier auf jedem Pfad des Baumes kein Knoten die gleiche Beschriftung wie ein anderer Knoten des gleichen Pfades haben. Sei $L(p) \subseteq \{1, \ldots, m\}$ die Menge der Beschriftungen der von p besuchten Knoten. Ein solcher Graph nennen wir *einheitlich*, falls ...

(*) $L(p)$ für einen an der Wurzel beginnenden Pfad p nur von dem letzten Knoten v von p abhängt (wir können dann auch $L(v)$ schreiben) und

(**) für jede Senke t gilt, dass $L(t) = \{1, \ldots, m\}$.

Satz Jeder Graph, der das Wortproblem $W^*(n,m)$ mit den gerade beschriebenen Eigenschaften löst, hat mindestens $2^{\Omega(m/\log n)}$ Knoten.

Beweis Sei $G = (V,E)$ ein Graph mit Wurzel w, der das Problem $W^*(n,m)$ löst. Wir können annehmen, dass G einheitlich ist, da wie man leicht einsieht jeder solche Graph durch Vergrößern um einen Faktor von höchstens m in einen einheitlichen Graphen erweitert werden kann. Für einen Knoten v von G definieren wir $I(v) := \{\, a \in A \mid$ keine der Zuweisungen $x_i = a (i \in L(v))$ wird entlang irgendeines Pfades von r nach v festgelegt $\}$. Beachte, dass $I(w) = A$, $I(v)$ sich bei einer Zuweisung, also bei einem Lauf entlang einer Spalte, seinen Wert nur verringern kann. Es gilt $a \in I(t)$ für eine mit a beschriftete Senke t. Außerdem gilt $I(v) \neq \emptyset, \forall v \in V$.

Wir bezeichnen eine von v ausgehende, mit a beschriftete Kante als legal, falls $a \in I(v)$, und sonst als illegal. Wir definieren nun wie in Abschnitt 4.2 einen Random Walk **p** in G, welcher ausgehend von der Wurzel an jedem Knoten eine der dort ausgehenden legalen Kanten zufällig (mit gleichen Wahrscheinlichkeiten) wählt und so zum nächsten Knoten gelangt. Es gilt, dass **p** mit Wahrscheinlichkeit 1 an einer Senke t ankommt. Da G einheitlich ist hat **p** die Länge m. Sei $k := \lceil \log n \rceil$ und seien $w = v_0, \mathbf{v}_1, \ldots, \mathbf{v}_k = \mathbf{t}$ die Knoten die den Pfad in Teile der Länge von mindestens jeweils $\lfloor \frac{m}{k} \rfloor$ unterteilen. Da $|I(v_0| = |A| = n, |I(\mathbf{v}_k)| \leq 1$ und da $I(\mathbf{v}_\nu)$ bei jedem Schritt im Pfad kleiner wird, gilt für $0 \leq \nu \leq k-1$:

$$|I(\mathbf{v}_{\nu+1})| \geq \frac{1}{2}|I(\mathbf{v}_\nu)|$$

Wir bezeichnen ein Paar (u_0, u_1) von Knoten als *gut*, wenn gilt

$$|L(u_1) - L(u_0)| \geq \left\lfloor \frac{m}{k} \right\rfloor, I(u_1) \subseteq I(u_0), |I(u_1)| \geq \frac{1}{2}|I(u_0)|.$$

Es gilt nun zu zeigen, dass für ein beliebiges *gutes* Paar gilt, dass

$$Prob(u_0 \text{ und } u_1 \text{ kommen in } \mathbf{p} \text{ in dieser Reihenfolge vor}) \leq 2^{-\lfloor \frac{m}{k} \rfloor}.$$

Um dies zu zeigen beachte, dass jeder erfolgreiche Pfad **p** zwischen den Knoten u_0 und u_1 nur die Knoten v besuchen kann für die gilt

$$I(u_0) \supseteq I(v) \supseteq I(u_1).$$

An jedem solchen Knoten v gibt es $|I(v)| \geq |I(u_1)|$ ausgehende legale Kanten, aber höchstens $|I(v) - I(u_1)|$ dieser Kanten führen **p** zu dem Knoten u_1. Daher be-

trägt die Wahrscheinlichkeit, dass ein Random Walk an einem beliebigen Knoten v zwischen den Knoten u_0 und u_1 die richtige Kante wählt, höchstens

$$\frac{|I(v) - I(u_1)|}{|I(v)|} \leq 1 - \frac{|I(u_1)|}{|I(u_0)|} \leq \frac{1}{2}.$$

Da auf dem Weg von u_0 nach u_1 mindestens $\lfloor \frac{m}{k} \rfloor$-mal diese richtige Wahl getroffen werden muß, gilt

$$Prob(u_0 \text{ und } u_1 \text{ kommen in } \mathbf{p} \text{ in dieser Reihenfolge vor}) \leq 2^{-\lfloor \frac{m}{k} \rfloor}.$$