Mitschrift zur Vorlesung
Randomisierte Algorithmen

Dozent: Prof. Dr. Rolf Wanka

Sommersemester 2016

Disclaimer

Diese Mitschrift ist nicht offiziell. Insbesondere erhebe ich keinen Anspruch auf Vollständigkeit oder Korrektheit.

Kapitel 1. Zwei Beispiele zum Warmwerden

1.1. Randomized Quicksort

Das Pivot-Element wird zufällig ausgewählt.

Algorithmus RandQS
input: S : array aus n (verschiedenen) Schlüsseln
output: Elemente von S in aufsteigend sortierter Reihenfolge

- wähle ein Pivot-Element y aus S u.a.r. (uniformly at random, d. h. gleichverteilt, unabhängig)
- $S_1 := \{x \in S |\ x < y\}$; $S_2 := \{x \in S |\ x > y\}$
- if $S_1 \neq \varnothing$ then RandQS(S_1) fi
- gib y aus
- if $S_2 \neq \varnothing$ then RandQS(S_2) fi

1.1.1. Laufzeit im Worst Case

Schlüssel	Anzahl der Vergleiche	Wahrscheinlichkeit
① 2 3 4 5 6 7	6	$\frac{1}{7}$
② 3 4 5 6 7	5	$\frac{1}{6}$
③ 4 5 6 7	4	$\frac{1}{5}$
④ 5 6 7	3	$\frac{1}{4}$
⑤ 6 7	2	$\frac{1}{3}$
⑥ 7	1	$\frac{1}{2}$
⑦	0	1

Gesamtzahl der Vergleiche: $\sum_{i=1}^{n} (i-1) = \sum_{i=0}^{n-1} i = \frac{n(n-1)}{2} = \Theta\left(n^2\right)$

Wahrscheinlichkeit, dass dieser Fall eintritt: $\prod_{i=1}^{n} \frac{1}{i} = \frac{1}{n!}$

1.1.2. Probabilistische Laufzeitanalyse

Wir bestimmen die erwartete Laufzeit von RandQS. s_i sei der Schlüssel mit Rang i, d. h. derjenige Schlüssel, der am Ende der Sortierung an Stelle i steht. (s_i, s_j) stoßen höchstens einmal aufeinander, denn bei einem Vergleich muss einer der beteiligten Schlüssel das Pivot-Element sein.

Best case: $T(n) = 2T\left(\frac{n}{2}\right) + n$; worst case: $T(n) = T(n-1) + n$

Zur Analyse definieren wir eine Zufallsvariable (ZV) (auch Indikatorvariable) für $1 \leq i < j \leq n$:

$$x_{ij} = \begin{cases} 1 & \text{falls RandQS } s_i \text{ und } s_j \text{ vergleicht} \\ 0 & \text{sonst} \end{cases}$$

Wir definieren eine weitere ZV Z, die Gesamtzahl der Vergleiche:

$$Z = \sum_{i=1}^{n} \sum_{j=i+1}^{n} x_{ij}$$

Erwartete Anzahl an Vergleichen:

$$E[Z] = E\left[\sum_{i=1}^{n} \sum_{j=i+1}^{n} x_{ij}\right] = \sum_{i=1}^{n} \sum_{j=i+1}^{n} E[x_{ij}]$$

Dabei ist $E[x_{ij}] = 0 \cdot \Pr[x_{ij} = 0] + 1 \cdot \Pr[x_{ij} = 1] = \Pr[x_{ij} = 1] =: p_{ij}$.

s_i und s_j werden nur dann miteinander verglichen, wenn kein $s_k \in \{s_{i+1}, ..., s_{j-1}\}$ vorher als Pivot-Element gewählt wurde. Damit ist

$$x_{ij} = \begin{cases} 1 & \text{falls der } s_j\text{-Knoten Nachfolger des } s_i\text{-Knoten ist oder umgekehrt} \\ 0 & \text{sonst} \end{cases}$$

$$p_{i,j} = \Pr[s_i \text{ wird als erstes Pivot-Element aus } \{s_i, ..., s_j\} \text{ ausgewählt}]$$
$$+ \Pr[s_j \text{ wird als erstes Pivot-Element aus } \{s_i, ..., s_j\} \text{ ausgewählt}] = \frac{2}{j-i+1}$$

$$\Rightarrow E[Z] = \sum_{i=1}^{n} \sum_{j=i+1}^{n} E[x_{ij}] = \sum_{i<j} \frac{2}{j-i+1} = \sum_{i=1}^{n-1} \sum_{j=2}^{n-i+1} \frac{2}{j} = ... = (2n+2) \sum_{i=1}^{n} \frac{1}{i} - 4n \leq 2n \ln(n) = \frac{2n \log_2(n)}{\log(e)}$$

$\left(\text{harmonische Reihe: } \ln(i+1) \leq \sum_{k=1}^{i} \frac{1}{k} \leq \ln(i) + 1\right)$

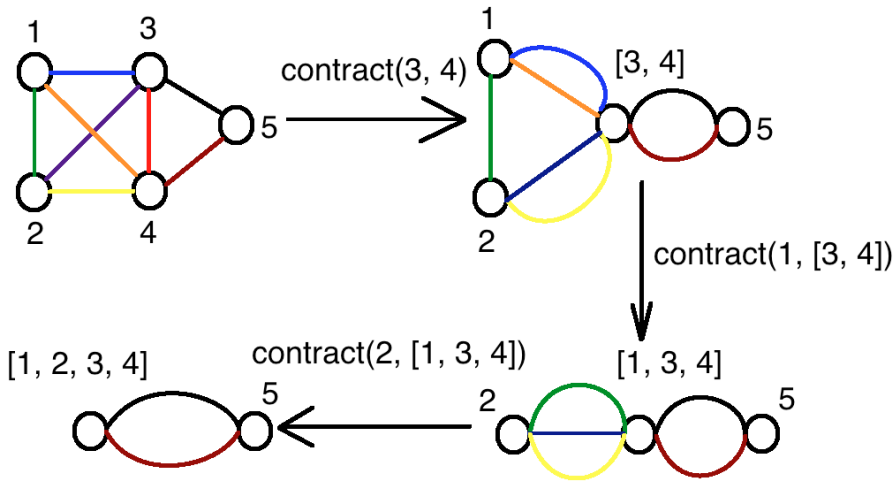
1.2. Ein Min-Cut-Algorithmus

Def. 1.2: In einem Multigraphen kann es mehr als eine Kante zwischen zwei Knoten geben. Ein Schnitt (cut) durch einen Multigraphen G ist eine Kantenmenge, deren Entfernung aus G den Graphen G in zwei oder mehr Zusammenhangskomponenten zerlegt. Ein minimaler Schnitt ist ein Schnitt kleinstmöglicher Kardinalität.

Bem. 1.3: Das Min-Cut-Problem ist in Polynomzeit deterministisch lösbar, aber mit "größerem" Polynom. (Max-Cut-Problem ist NP-vollständig.)

Die Operation CONTRACT arbeitet wie folgt:
input Multigraph G, Kante $\{x, y\}$ zwischen Knoten x, y
output Multigraph, in dem x und y "fusioniert" wurden

- ersetze x und y durch einen neuen Knoten z
- lösche alle Kanten zwischen x und y
- ersetze alle übrigen Kanten $\{x, v\}$ bzw. $\{y, v\}$ durch die neue Kante $\{z, v\}$ (Multikanten)



Algorithmus CONTRACT_GRAPH
input Graph G
output Schnitt C durch G
$H = G$
while H enthält mehr als zwei Knoten do

- wähle eine Kante $\{x, y\}$ aus H u.a.r.
- CONTRACT(x, y) in H

end
C = Menge der Kanten in H, zurückgerechnet auf G.

1.2.1. Analyse der Erfolgswahrscheinlichkeit von CONTRACT_GRAPH(G)

Wir interessieren uns für die Wahrscheinlichkeit, dass CONTRACT_GRAPH eine vorgegebene fest gewählte optimale Lösung ("Referenzlösung") "überleben" lässt. Sei e_i die in der i-ten Iteration der while-Schleife kontrahierte Kante $(1 \le i \le n-2)$ und sei H_i der zugehörige Graph ($H_0 = G$, H_{n-2} der letzte Graph).

- Beobachtung 1: C ist ein Schnitt durch H_0.
- Beobachtung 2: In jedem H_i ist die Größe eines minimalen Schnitts nicht kleiner als in G.

Sei K ein beliebiger minimaler Schnitt durch G, der Referenzschnitt. K überlebt CONTRACT_GRAPH(G) genau dann, wenn in jeder Iteration keine Kante aus K zur Kontraktion ausgewählt wurde.

$$\Pr[C \text{ ist ein minimaler Schnitt}]_{n-2} \ge \Pr[C = K] = \Pr\left[\bigwedge_{1 \le i \le n-2} (e_i \notin K)\right] = \prod_{i=1}^{n=2} \Pr\left[e_i \notin K \,\middle|\, \bigwedge_{1 \le j < i} e_j \notin K\right]$$

Lemma 1.4: Sei $n := |V|$. Für i, $1 \le i \le n-2$, gilt:

$$\Pr\left[e_i \in K \,\middle|\, \bigwedge_{1 \le j < i} (e_j \notin K)\right] \le \frac{2}{n-i+1}$$

Beweis: Induktion nach i.
$i = 1$: Jeder Knoten von G hat mindestens den Grad $|K|$, denn sonst könnte man einen Knoten, der kleineren Grad hat, als Seite eines noch besseren Schnitts nehmen. $n\frac{|K|}{2} \le |E|$. Die Wahrscheinlichkeit, dass eine Kante aus K gewählt wird, ist $\frac{|K|}{|E|} \le \frac{|K|}{n\frac{|K|}{2}} = \frac{2}{n}$.

$i = 2$: H_1 hat $n-1$ Knoten. Auch hier hat jeder Knoten Grad mindestens $|K|$. Also $\Pr[e_i \in K \,|\, e_1 \notin K_1] \le \frac{|K|}{\frac{n-1}{2}|K|} = \frac{2}{n-1}$.

allg. für i: $\Pr\left[e_i \in K \,\middle|\, \bigwedge_{1 \le j < i} (e_j \notin K)\right] \le \frac{|K|}{\frac{n-i+1}{2}|K|} = \frac{2}{n-i+1}$. □

$$\Pr[C \text{ ist minimaler Schnitt}] \ge \prod_{i=1}^{n-2} \left(1 - \frac{2}{n-i+1}\right) = \prod_{i=1}^{n-2} \frac{n-i-1}{n-i+1} = \frac{2}{n(n-1)} \approx \frac{1}{n^2}$$

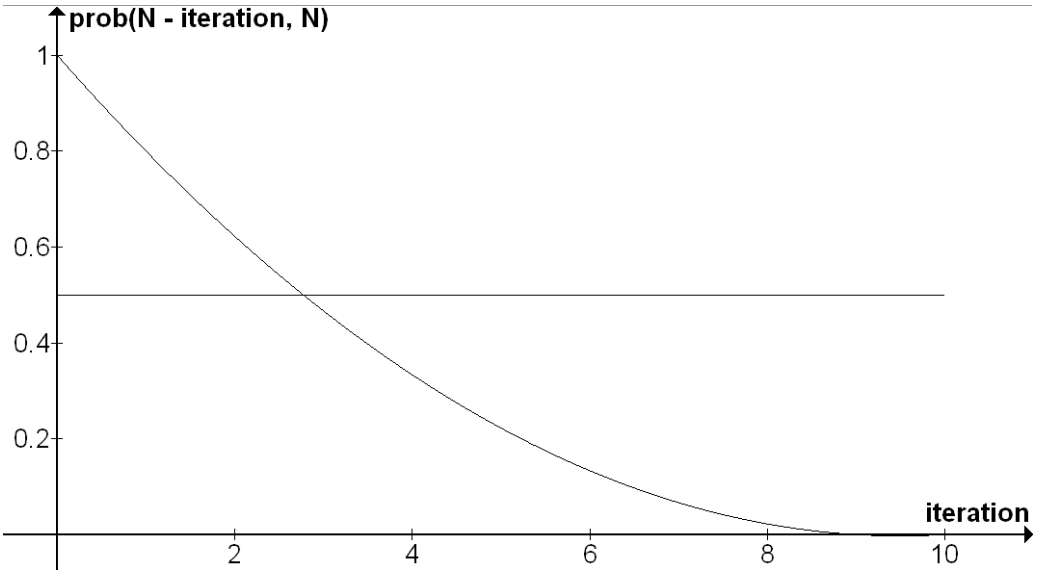
Satz 1.5:

a) CONTRACT_GRAPH(G) gibt mit der Wahrscheinlichkeit von mindestens $\frac{2}{n(n-1)} \left(\approx \frac{1}{n^2}\right)$ einen minimalen Schnitt durch G aus.
b) Die erwartete Anzahl an Wiederholungen, bis CONTRACT_GRAPH(G) einen minimalen Schnitt durch G findet, ist höchstens $\frac{n(n-1)}{2}$.

Fakt: CONTRACT_GRAPH kann so implementiert werden, dass er in Zeit $\mathcal{O}\left(n^2\right)$ läuft. Inklusive der Wiederholungen ergibt dies aus 1.5 b) $\mathcal{O}\left(n^4\right)$.

Korollar 1.6 (zu Lemma 1.4): Die Wahrscheinlichkeit, dass nach $n-l$ CONTRACTs der Referenzschnitt K noch lebt, ist mindestens $\frac{l(l-1)}{n(n-1)}$ für $l \in \{2, ..., n\}$. l ist also die Anzahl der Knoten im kontrahierten Graphen.

$$\Pr\left[K \text{ überlebt } n-l \text{ CONTRACTs}\right] = \frac{l\,(l-1)}{n\,(n-1)}.$$



$$\text{prob}\,(x, n) = \frac{x(x-1)}{n(n-1)}$$

1.2.2. Ein verbesserter Min-Cut-Algorithmus

Wie groß ist die Überlebenswahrscheinlichkeit des Referenzschnitts, wenn der kontrahierte Graph noch $\left\lceil \frac{n}{\sqrt{2}} + 1 \right\rceil$ Knoten enthält? Einsetzen ergibt: $\geq \frac{1}{2}$.

Algorithmus RECURSIVECONTRACT(G)
    if G hat höchstens 6 Knoten
    then gib optimale Lösung aus
    else
        G_1' := kontrahiere G herunter auf $\left\lceil \frac{n}{\sqrt{2}} + 1 \right\rceil$ Knoten
        c_1 = RECURSIVECONTRACT(G_1')
        G_2' = kontrahiere G herunter auf $\left\lceil \frac{n}{\sqrt{2}} + 1 \right\rceil$ Knoten
        c_2 = RECURSIVECONTRACT(G_2')
        gib den besseren Schnitt unter C_1 und C_2 aus
    fi
end

Laufzeit: Sei $T(n)$ die Laufzeit. Es gilt:

$$\begin{aligned} T(n) &= \text{const. für } n \leq 6 \\ T(n) &\leq 2\left(n^2 + T\left(\left\lceil \frac{n}{\sqrt{2}} + 1 \right\rceil\right)\right) \\ \Rightarrow T(n) &= \mathcal{O}\left(n^2 \log n\right) \end{aligned}$$

Insbesondere ist die Tiefe der Rekursion $\mathcal{O}(\log n)$.

Die Erfolgswahrscheinlichkeit, dass RECURSIVECONTRACT den Referenzschnitt K ausgibt, sei $P(n)$. Die Wahrscheinlichkeit, dass K bei einem Durchlauf überlebt, ist für $n \geq 7$

$$P(n) \geq 1 - \left(1 - \frac{1}{2} \cdot P\left(\left\lceil \frac{n}{\sqrt{2}} + 1 \right\rceil\right)\right)^2$$

und für $n \leq 6$ ist $P(n) \geq \frac{1}{15}$.

Variablentransformation: $p_{k+1} \geq 1 - \left(1 - \frac{1}{2} \cdot p_k\right)^2 = p_k - \frac{1}{4}p_k^2$; $p_0 \geq \frac{1}{15}$ und $n = \Theta\left(\sqrt{2}^k\right)$.
Setze $z_k = \frac{4}{p_k} - 1 \Rightarrow p_k = \frac{4}{z_k + 1}$. Damit: $z_0 = 59$, $z_{k+1} = z_k + 1 + \frac{1}{z_k}$. Da $z_k \geq 1$, gilt $z_{k+1} \leq z_k + 2$.
Also ist $k < z_k < 2k + 59 \Rightarrow z_k = \Theta(k) \Rightarrow p_k = \frac{4}{z_k + 1} = \Theta\left(\frac{1}{k}\right) \Rightarrow P(n) = \Theta\left(\frac{1}{\log n}\right)$.

Satz 1.7:

a) RECURSIVECONTRACT(G) gibt mit Warscheinlichkeit $\Omega\left(\frac{1}{\log n}\right)$ einen minimalen Schnitt durch G aus.
b) Anzahl der Wiederholungen, bis RECURSIVECONTRACT einen minimalen Schnitt findet, ist $\mathcal{O}(\log n)$, d. h. Gesamtlaufzeit $\mathcal{O}\left(n^2 (\log n)^2\right)$.

1.2.3. Wahrscheinlichkeitsverstärkung

Wiederhole CONTRACT_GRAPH $\frac{n(n-1)}{2} \cdot c \cdot \ln n$ Mal (c: beliebige, frei wählbare Konstante) und gib den besten gefundenen Schnitt aus.

$$\Pr[\text{Kein minimaler Schnitt wird ausgegeben}] \leq \left(1 - \frac{2}{n(n-1)}\right)^{\frac{n(n-1)}{2} \cdot c \cdot \ln n} \leq e^{-c \cdot \ln n} = n^{-c} = \frac{1}{n^c}.$$

"Dieser Algorithmus findet mit hoher Wahrscheinlichkeit einen minimalen Schnitt".

Kapitel 2. "Balls & Bins" und Chernoff-Schranken

2.1. Lastverteilung

($m =$) n Bälle, n Kisten, die Bälle werden auf die Kisten geworfen, die Wahrscheinlichkeit, eine bestimmte Kiste zu treffen, ist $\frac{1}{n}$. Wir interessieren uns hier für die Anzahl der Bälle in der vollsten Kiste. Die Anzahl sei L. Die Kisten seien von 1 bis n durchnumeriert. L_i sei die Anzahl der Bälle in Kiste i. $E[L_i] = 1$ für alle i.

$$L = \max\{L_i \,|\, i \in \{1, ..., n\}\}, E[L] > 1 \text{ für } n \geq 2.$$

Es gibt einen Unterschied zwischen der erwarteten maximalen Last und der maximal erwarteten Last.

Def. 2.1: Sei X eine ZV für ein Zufallsexperiment mit Parameter n. Sei $f : \mathbb{N} \to \mathbb{R}^{\geq 0}$ eine Funktion. Eine Schranke der Form $X = \mathcal{O}(f(n))$ gilt mit hoher Wahrscheinlichkeit, wenn gilt:

$$\forall \alpha > 0 \; \exists c > 0 \; \exists n_0 \geq 1 \; \forall n \geq n_0 : \Pr[x \geq c \cdot f(n)] \leq \frac{1}{n^\alpha}$$

Satz 2.2: Bei n Bällen und n Kisten gilt:

$$L = \mathcal{O}\left(\frac{\log n}{\log\log n}\right) \begin{array}{c}\text{mit hoher Wahrscheinlichkeit (m. h. W)}\\ \text{with high probability (w. h. p.)}\end{array}$$

Beweis: Für Ball j und Kiste i sei X_j^i die ZV mit

$$X_j^i = \begin{cases}1 & \text{falls Ball } j \text{ in Kiste } i \text{ landet}\\ 0 & \text{sonst.}\end{cases}$$

$L_i = \sum_{j=1}^{n} X_j^i$. Wir betrachten feste Kiste i und interessieren uns für die Wahrscheinlichkeit, dass in Kiste i mindestens k Bälle landen, d. h. für das Ereignis $L_i \geq k$. Gesucht ist also $\Pr[L_i \geq k]$. Dies gilt genau dann, wenn es eine Teilmenge $I \subseteq \{1,...,n\}$ der Bälle gibt mit $|I| = k$, sodass alle Bälle aus I in die Kiste i fallen.

$$\Pr[L_i \geq k] = \Pr\left[\exists I \subseteq \{1,...,n\}, |I| = k, \forall j \in I : X_j^i = 1\right] \leq \sum_{I\subseteq\{1,...,n\};|I|=k} \Pr\left[\forall j \in I : X_j^i = 1\right]$$

$$= \sum_{I\subseteq\{1,...,n\};|I|=k} \prod_{j\in I} \Pr\left[X_j^i = 1\right] = \sum_{I\subseteq\{1,...,n\};|I|=k} \left(\frac{1}{n}\right)^k = \binom{n}{k}\cdot\left(\frac{1}{n}\right)^k$$

Mit der Nebenrechnung

$$\binom{n}{k} = \frac{n!}{(n-k)!k!} \leq \frac{n^k}{k!} = \frac{n^k}{k^k}\cdot\frac{k^k}{k!} < \left(\frac{n}{k}\right)^k \cdot \sum_{i=0}^{\infty}\frac{k^i}{i!} = \left(\frac{n\cdot e}{k}\right)^k$$

erhalten wir

$$\Pr[L_i \geq k] \leq \left(\frac{n\cdot e}{k}\right)^k \cdot \left(\frac{1}{n}\right)^k = \left(\frac{e}{k}\right)^k.$$

Damit gilt $k = \max\left\{2\alpha\frac{\log n}{\log\log n}, e\sqrt{\log n}\right\} = \mathcal{O}\left(\frac{\log n}{\log\log n}\right)$; $\Pr[L_i \geq k] \leq \left(\frac{e}{k}\right)^k \leq \left(\frac{1}{\sqrt{\log n}}\right)^k$. Für $n > 2$ ist $\frac{1}{\sqrt{\log n}} < 1$, also ist $\Pr[L_i \leq k] \leq \left(\frac{1}{\sqrt{\log n}}\right)^{2\alpha\cdot\frac{\log n}{\log\log n}} = \left(\frac{1}{\log n}\right)^{\alpha\cdot\frac{\log n}{\log\log n}} = \left(\frac{1}{2}\right)^{\alpha\cdot\log n} = \frac{1}{n^\alpha}$.
(Der Faktor $\frac{1}{2}$ entsteht dadurch, dass mit log der Zweierlogarithmus gemeint ist.)

Nun wollen wir wissen, wie groß die Wahrscheinlichkeit ist, dass eine beliebige Kiste mindestens k Bälle enthält, höchstens ist.

$$\Pr[L \geq k] = \Pr[\exists i \in \{1,...,n\} : L_i \geq k] \leq \sum_{i=1}^{n}\Pr[L_i \geq k] \leq n\cdot\frac{1}{n^\alpha} = \frac{1}{n^{\alpha-1}}.$$

□

2.2. Chernoff-Schranken

Satz (Markov-Ungleichung): Sei X eine nicht negative ZV. Dann gilt für alle $t > 0$: $\Pr[X \geq t] \leq \frac{E[X]}{t}$. Mit $t = c\cdot E[X]$ für $c \geq 1$: $\Pr[X \geq c\cdot E[X]] \leq \frac{1}{c}$.

Beweis: Sei $t > 0$ gegeben. Sei $\tau(X)$ wie folgt definiert:

$$\tau(X) = \begin{cases}1 & \text{falls } \frac{X}{t} \geq 1\\ 0 & \text{sonst}\end{cases}.$$

Es gilt: $\tau(X) \leq \frac{X}{t}$ für alle $X \geq t$.

$$\frac{E[X]}{t} = E\left[\frac{X}{t}\right] \geq E[\tau(X)] = \Pr[\tau(X) = 1] = \Pr\left[\frac{X}{t} \geq 1\right] = \Pr[X \geq t]$$

□

Def. (Varianz, Standardabweichung): Sei $X : \Omega \to \mathbb{R}$ eine ZV,

$$\mathrm{Var}[X] = E\left[(X - E[X])^2\right] = E\left[X^2 - 2XE[X] + E[X]^2\right] = E\left[X^2\right] - 2E[X]^2 + E[X]^2 = E\left[X^2\right] - E[X]^2$$
$$\sigma[X] = \sqrt{\mathrm{Var}[X]}$$

Satz (Rechenregeln für die Varianz): Seien X und Y unabhängige ZV und sei c eine beliebige Konstante. Dann gelten folgende Rechenregeln:

a) $\mathrm{Var}[X + Y] = \mathrm{Var}[X] + \mathrm{Var}[Y]$
b) $\mathrm{Var}[c \cdot X] = c^2 \cdot \mathrm{Var}[X]$
c) Sei X eine $0-1$–ZV: $\mathrm{Var}[X] = E[X] \cdot (1 - E[X])$

Satz (Tschebyscheffsche Ungleichung): Sei X eine ZV. Für jedes $\epsilon > 0$: $\Pr[|X - E[X]| \geq \epsilon] \leq \frac{\mathrm{Var}[X]}{\epsilon^2}$.

Beweis:

$$\mathrm{Var}[X] = \sum_{\omega\in\Omega} (X(\omega) - E[X])^2 \cdot \Pr[\omega] \geq \sum_{\omega\in\Omega;(X(\omega)-E[X])^2\geq\epsilon^2} (X(\omega) - E[X])^2 \cdot \Pr[\omega]$$
$$\geq \sum_{\omega\in\Omega;|X(\omega)-E[X]|\geq\epsilon} \epsilon^2 \cdot \Pr[\omega] = \epsilon^2 \sum_{\omega\in\Omega;|X(\omega)-E[X]|\geq\epsilon} \Pr[\omega] = \epsilon^2 \cdot \Pr[|X \cdot E[X]|] \geq \epsilon$$

□

Für $\epsilon = k \cdot \sigma[X]$: $\Pr[|X - E[X]| \leq k \cdot \sigma[X]] \leq \frac{1}{k^2}$.
Für $\epsilon = \epsilon' \cdot E[X]$: $\Pr[|X - E[X]| \geq \epsilon' \cdot E[X]] \leq \frac{1}{\epsilon'^2} \cdot \frac{\mathrm{Var}[X]}{E[X]^2}$.

Münzwurf, fair: n Würfe. Wir interessieren uns für die Wahrscheinlichkeit, dass wenigstens $\frac{3}{4} \cdot n$ Mal "Zahl" kommt.
ZV: $X_i = \begin{cases} 1 & i-\text{ter Wurf ist "Zahl"} \\ 0 & \text{sonst} \end{cases}$; $E[X_i] = \frac{1}{2}; \mathrm{Var}[X_i] = \frac{1}{4}; X = \sum_{i=1}^{n} X_i$.
$\Pr\left[|X - E[X]| \geq \frac{n}{4}\right] \leq \frac{\mathrm{Var}[X]}{\left(\frac{n}{4}\right)^2} = \frac{\frac{n}{4}}{\left(\frac{n}{4}\right)^2} = \frac{4}{n}$.

Satz (Chernoff-Schranken): Seien $X_1, ..., X_n$ unabhängige $0-1$–ZV, $X = \sum_{i=1}^{n} X_i$. $E[X] = \sum_{i=1}^{n} \Pr[X_i = 1]$. Dann gilt für jedes ϵ, $0 < \epsilon < 1$:

a) $\Pr[X \geq (1 + \epsilon) \cdot E[X]] < \left(\frac{e^\epsilon}{(1+\epsilon)^{1+\epsilon}}\right)^{E[X]}$
b) $\Pr[X \geq (1 + \epsilon) \cdot E[X]] \leq e^{-\frac{\epsilon^2}{3} \cdot E[X]}$

Beweis (von a)): Für $t > 0$: $\Pr[X \geq (1 + \epsilon) \cdot E[X]] = \Pr\left[e^{t \cdot X} \geq e^{t \cdot (1+\epsilon) \cdot E(X)}\right]$. Mit Markov folgt:

$$\Pr\left[e^{t \cdot X} \geq e^{t \cdot (1+\epsilon) \cdot E(X)}\right] \leq \frac{E\left[e^{t \cdot X}\right]}{e^{t \cdot (1+\epsilon) \cdot E[X]}} = e^{-t \cdot (1+\epsilon) \cdot E[X]} \cdot E\left[e^{t \cdot \sum_{i=1}^{n} X_i}\right] = e^{-t \cdot (1+\epsilon) \cdot E[X]} \cdot E\left[\prod_{i=1}^{n} e^{t \cdot X_i}\right]$$
$$= e^{-t \cdot (1+\epsilon) \cdot E[X]} \cdot \prod_{i=1}^{n} E\left[e^{t \cdot X_i}\right] = e^{-t \cdot (1+\epsilon) \cdot E[X]} \cdot \prod_{i=1}^{n} \left((1 - \Pr[X_i = 1]) + e^t \cdot \Pr[X_i = 1]\right)$$
$$= e^{-t \cdot (1+\epsilon) \cdot E[X]} \cdot \prod_{i=1}^{n} \left(1 + \Pr[X_i = 1] \cdot \left(e^t - 1\right)\right) < e^{-t \cdot (1+\epsilon) \cdot E[X]} \cdot \prod_{i=1}^{n} e^{\Pr[X_i=1] \cdot \left(e^t - 1\right)}$$
$$= e^{-t \cdot (1+\epsilon) \cdot E[X]} \cdot e^{\sum_{i=1}^{n} \Pr[X_i=1] \cdot \left(e^t - 1\right)} = e^{-t \cdot (1+\epsilon) \cdot E[X]} \cdot e^{E[X] \cdot \left(e^t - 1\right)}$$
$$= \left(e^{-t \cdot (1+\epsilon) + e^t - 1}\right)^{E[X]} = \left(\frac{e^{e^t - 1}}{e^{t \cdot (1+\epsilon)}}\right)^{E[X]}.$$

Mit $t := \ln(1 + \epsilon)$ gilt $\left(\frac{e^{e^t - 1}}{e^{t \cdot (1+\epsilon)}}\right)^{E[X]} = \left(\frac{e^\epsilon}{(1+\epsilon)^{1+\epsilon}}\right)^{E[X]}$. □

3. Die probabilistische Methode

Gesucht: ein Objekt A mit der Eigenschaft E. Wir stellen die Frage, ob es ein solches Objekt überhaupt gibt. Wir würfeln ein Objekt A und zeigen:

$$\Pr[A \text{ hat die Eigenschaft } E \text{ nicht}] < 1$$

$\Rightarrow$ es gibt ein A mit E.

3.1. Das Max-Cut-Problem

Gegeben: Zusammenhängender Graph $G = (V, E), n = |V|, m = |E|$.
Gesucht: Partition von V in A und B, sodass möglichst viele Kanten zwischen A und B verlaufen. Diese Zahl der Kanten ist der Wert $c(A, B)$ des Schnitts $[A, B]$.
Allgemein: das Entscheidungsproblem ist NP-vollständig.

Satz: Es gibt immer einen Schnitt $[A, B]$ mit $c(A, B) \geq \frac{m}{2}$.

Beweis: Wir verwenden den folgenden randomisierten Algorithmus

CUT
input: $G = (V, E)$
output: Schnitt $[A, B]$
$A := \varnothing$; $B := \varnothing$; $n := |V|$;
for i:=1 to n do
$\quad$ mit Wahrscheinlichkeit $\frac{1}{2}$: lege Knoten v_i nach A
$\quad$ mit Wahrscheinlichkeit $\frac{1}{2}$: lege Knoten v_i nach B
end
gib $[A, B]$ aus

Wir zeigen: $E[c(A, B)] = \frac{m}{2}$ ($\Rightarrow$ es muss einen Schnitt mit $c(A, B) \geq \frac{m}{2}$ geben). Wir haben: $\Pr[v_i \in A] = \Pr[v_i \in B] = \frac{1}{2}$. Wir ordnen die Kanten beliebig an: $e_1, e_2, ..., e_n$. Indikator:

$$X_j = \begin{cases} 1 & \text{falls } e_j \text{ über den Schnitt geht} \\ 0 & \text{sonst} \end{cases}.$$

$$c(A, B) = \sum_{j=1}^{m} X_j \Rightarrow E[c(A, B)] = E\left[\sum_{j=1}^{m} X_j\right] = \sum_{j=1}^{m} E[X_j] = \sum_{j=1}^{m} \Pr[e_j \text{ geht über den Schnitt}]$$

$e_j = \{v_x, v_y\}$:

$$\Pr[\{v_x, v_y\} \text{ geht über den Schnitt}] = \Pr[(v_x \in A \wedge v_y \in B) \vee (v_y \in A \wedge v_x \in B)]$$

$$= \Pr[v_x \in A \wedge v_y \in B] + \Pr[v_y \in A \wedge v_x \in B] = \Pr[v_x \in A] \cdot \Pr[v_y \in B] + \Pr[v_x \in B] \cdot \Pr[v_y \in A] = \frac{1}{2} \cdot \frac{1}{2} + \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{2}.$$

$$\Rightarrow E[c(A, B)] = \sum_{j=1}^{m} \frac{1}{2} = \frac{m}{2}.$$

Wie oft muss man CUT aufrufen, bis ein Schnitt der Größe $\geq \frac{m}{2}$ gefunden wurde?
CUT ist ein Monte-Carlo-Algorithmus dafür, da er auch kleinere Schnitte ausgeben kann.

Algorithmus LV_CUT
input: Graph $G(V, E)$ zusammenhängend
output: Schnitt $[A, B]$ mit $c(A, B) \geq \frac{|E|}{2}$
$m := |E|$; $t := 0$; repeat $t := t + 1$, $c(A, B) := \text{CUT}(G)$
until $c(A, B) \geq \frac{m}{2}$
gib $[A, B]$ aus.

(Las-Vegas-Algorithmus; die ZV ist die Laufzeit.)

Gesucht: $E[t]$. Sei $p := \Pr\left[c(A,B) \geq \frac{m}{2}\right]$.

$$\frac{m}{2} = E[c(A,B)] = \sum_{i=1}^{m} i \cdot \Pr[c(A,B) = i] = \sum_{i \leq \frac{m}{2}-1} i \cdot \Pr[c(A,B) = i] + \sum_{i \geq \frac{m}{2}} i \cdot \Pr[c(A,B) = i]$$
$$\leq \left(\frac{m}{2} - 1\right) \cdot (1-p) + m \cdot p \Rightarrow p \geq \frac{1}{\frac{m}{2}+1}$$

$\Rightarrow E[t] = \frac{1}{p} \leq \frac{m}{2} + 1$.

3.2. Independent Sets und Sample & Modify

Gegeben: Graph $G = (V,E)$. Eine Knotenmenge U, $U \subseteq V$, heißt unabhängig (independent), falls es keine Kanten in E gibt, die Knoten aus U verbinden. Gesucht ist eine möglichst große unabhängige Knotenmenge U. Das zugehörige Entscheidungsproblem ist NP-vollständig.

Satz: G hat eine unabhängige Menge U mit $|U| \geq \frac{n^2}{4 \cdot m}$.

Beweis: Wir erzeugen zufällig gewählte Teilmengen von V. (Sample) Die ist nicht unbedingt eine unabhängige Menge und und wird abschließend "repariert" (Modify).
Sei $d = \frac{2m}{n}$ der durchschnittliche Knotengrad. ($\Rightarrow m = \frac{n \cdot d}{2}$)

Algorithmus INDEPENDENT_SET
input: $G = (V,E)$ Graph
output: U, $U \subseteq V$, unabhängige Knotenmenge
$n := |V|$; $m := |E|$; $d := \frac{2m}{n}$; $U := V$;
$H :=$ Kopie von G
{Sample}
for $i := 1$ to n do
    mit Wahrscheinlichkeit $1 - \frac{1}{d}$ lösche v_i und seine Kanten aus H bzw. U.
    $\left\{\Pr[v_i \text{ überlebt}] = \frac{1}{d}\right\}$
end
{Modify}
for alle übriggebliebenen Kanten do
    lösche die Kante und einen ihrer Knoten aus H bzw. U.
end

Sei X die Anzahl der Knoten, die die Sample-Phase überleben. $E[X] = \frac{n}{d}$.
Sei Y die Anzahl der Kanten, die die Sample-Phase überleben. Eine Kante überlebt, wenn ihre beiden Endknoten überleben. Damit:

$$E[Y] = m \cdot \left(\frac{1}{d}\right)^2 = \frac{n \cdot d}{2} \cdot \left(\frac{1}{d}\right)^2 = \frac{n}{2d}.$$

Sei $Z = |U|$ die Anzahl der Knoten, die die Modify-Phase überleben:

$$E[Z] \geq E[X - Y] = E[X] - E[Y] = \frac{n}{d} - \frac{n}{2d} = \frac{n}{2d} = \frac{n}{2 \cdot \frac{2m}{n}} = \frac{n^2}{4m}.$$

□

SAT und das Lovasz-Local-Lemma (LLL)

Sei Φ eine k–KNF (k Literale je Klausel). Zwei Klauseln C_1 und C_2 sind benachbart, falls sie gemeinsame Variablen enthalten. Andernfalls sind sie unabhängig.
$\Gamma_\Phi(C) = \{C' \,|\, C \text{ und } C' \text{ sind benachbart in } \Phi\}$

Satz: Sei Φ eine k–KNF. Falls $|\Gamma(C)| \leq \frac{2^k}{e} - 1$ für alle Klauseln C in Φ, dann ist Φ erfüllbar.

Beweis: Jede Klausel habe eine Nachbarschaft von höchstens $\frac{2^k}{e} - 1 =: d$. Die Variablen von Φ werden u. a. r. mit TRUE oder FALSE belegt. Diese Belegung heiße α.
Wir bestimmen $\Pr[\alpha \text{ erfüllt } \Phi]$ $(=: \Pr[\Phi])$. Wenn $\Pr[\Phi] > 0$, dann ist Φ erfüllbar.
Sei Φ' eine (echte) Subformel von Φ, die durch Entfernung von mindestens einer Klausel aus Φ entsteht. Sei $C \in \Phi \backslash \Phi'$. $\Pr[\Phi']$ sei bekannt. Wie verringert sich die Wahrscheinlichkeit, wenn C zu Φ' zurückgelegt wird?

Behauptung: $\Pr[\Phi' \wedge C] \geq \left(1 - \frac{e}{2^k}\right) \cdot \Pr[\Phi']$ (äquivalent: $\Pr[\Phi' \wedge \neg C] \leq \frac{e}{2^k} \cdot \Pr[\Phi']$).
Mit der Behauptung ist der Satz bewiesen, da die leere Formel erfüllt wird, also $\Pr[\varnothing] = 1$, und jedes Zurücklegen von Klauseln eine Wahrscheinlichkeit $\neq 0$ erzeugt.

Beweis der Behauptung: Annahme: Die Behauptung gilt für alle echten Subformeln von Φ'.

- Ist C unabhängig von Φ', so ist $\Pr[\Phi' \wedge C] = \left(1 - \frac{1}{2^k}\right) \cdot \Pr[\Phi']$.
- C habe nun mit Φ' einige Variablen gemeinsam. Entferne aus Φ alle (das sind höchstens d viele) Klauseln, die Variablen mit C gemeinsam haben. $\Phi'' := \Phi' \backslash \Gamma(C)$. Φ'' und C sind unabhängig, also: $\Pr[\Phi'' \wedge \neg C] = \Pr[\Phi''] \cdot \frac{1}{2^k}$. Aus Φ' wurden höchstens d Klauseln entfernt. Für Φ'' gilt die Induktionsannahme. Wenn nun alle entfernten Klauseln wieder zurückgelegt werden, erhalten wir erneut Φ' und

$$\Pr[\Phi'] \geq \left(1 - \frac{e}{2^k}\right)^d \cdot \Pr[\Phi''] \geq \frac{1}{e} \cdot \Pr[\Phi''].$$

  Da jede Belegung, die Φ' erfüllt, auch Φ'' erfüllt, gilt auch:

$$\Pr[\Phi' \wedge \neg C] \leq \Pr[\Phi'' \wedge \neg C] = \frac{1}{2^k} \cdot \Pr[\Phi'']$$

  Zusammensetzen:

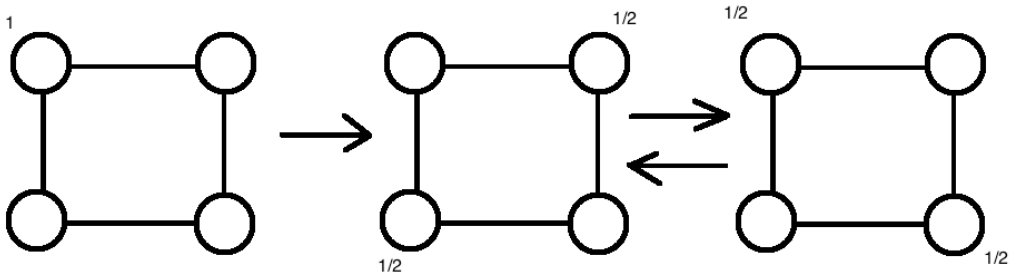
$$\frac{\Pr[\Phi' \wedge \neg C]}{\Pr[\Phi'']} \leq \frac{\frac{1}{2^k} \cdot \Pr[\Phi'']}{\frac{1}{e} \cdot \Pr[\Phi'']} = \frac{e}{2^k}.$$

□ (Behauptung)
□ (Satz)

4. Random Walks auf ungerichteten Graphen und Erreichbarkeit

Def.: Sei $G = (V, E)$ ein ungerichteter, zusammenhängender Graph. Ein Random Walk (deutsch: Irrlauf) auf G ist eine durch eine Markov-Kette beschreibbare zufällige Reise eines Objektes auf den Knoten von G. Ist das Objekt zu einem Zeitpunkt in Knoten i mit Grad $d(i)$, dann wechselt es mit Wahrscheinlichkeit $\frac{1}{d(i)}$ zum Nachbarn j.

Unerwünscht: Periodische Random Walks



Es gilt: ein Random Walk ist aperiodisch $\Leftrightarrow$ G ist nicht bipartit.
Random Walks kann man aperiodisch machen, indem man self-loops hinzufügt.

Satz: Ein aperiodischer Random Walk auf einem zusammenhängenden Graphen G konvergiert gegen die stationäre Verteilung π mit $\pi_i = \frac{d(i)}{2 \cdot |E|}$ (Self-loops zählen doppelt!)
(stationär: $\pi \cdot M = \pi$, M Markov-Kette)

Beweis:

i) π ist eine Verteilung:

$$\sum_{i=1}^{n} \pi_i = \sum_{i=1}^{n} \frac{d(i)}{2 \cdot |E|} = \frac{1}{2 \cdot |E|} \cdot \sum_{i=1}^{n} d(i) = \frac{1}{2 \cdot |E|} \cdot 2 \cdot |E| = 1.$$

ii) π ist stationär:

$$\pi_i = \sum_{j \in \Gamma(i)} \frac{d(j)}{2 \cdot |E|} \cdot \frac{1}{d(j))} = \frac{1}{2 \cdot |E|} \cdot \sum_{j \in \Gamma(i)} 1 = \frac{d(i)}{2 \cdot |E|}.$$

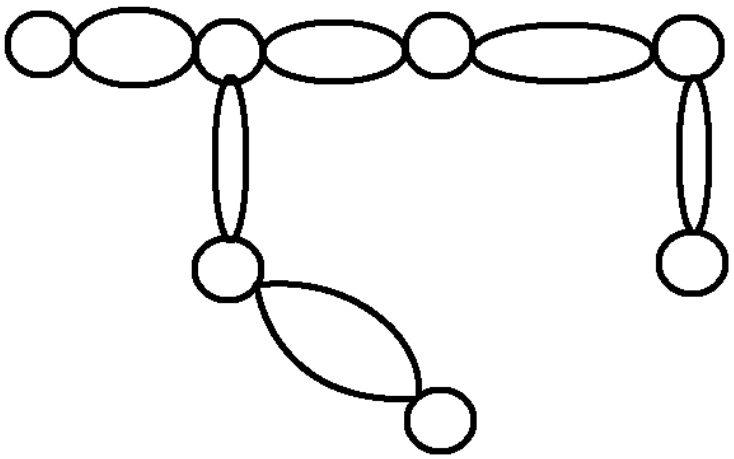
Da der Walk aperiodisch ist, konvergiert er auch. □

Korollar: Sei $h_{i,i}$ die erwartete Zeit, die vergeht, um von i wieder nach i zurückzukehren. $h_{i,i} = \frac{1}{\pi_i} = \frac{2 \cdot |E|}{d(i)}$.

Def.: Die Cover Time eines Graphen ist das Maximum über alle Knoten i der erwarteten Zeit, um alle Knoten von G, gestartet bei i, zu besuchen.

Satz: Die Cover Time von G ist höchstens $4 \cdot |V| \cdot |E|$.

Beweis: Nimm einen beliebigen Spannbaum von G und verdopple die Kanten. An jedem Knoten ist dadurch der Knotengrad gerade.



⇒ Es gibt eine Eulertour auf dem "verdoppelten" Baum.
⇒ Cover-Time $< 2 \cdot (|V| - 1) \cdot 2 \cdot |E| < 4 \cdot |V| \cdot |E|$. □

Anwendung: $s-t$–Connectivity:
Gegeben $G = (V, E)$ Graph, nicht notwendigerweise zusammenhängend, und zwei ausgezeichnete Knoten $s, t \in V$.
Frage: Gibt es einen Weg zwischen s und t?
Tiefensuche benötigt $\Theta(|V|)$ Platz und $\Theta\left(|V|^2\right)$ Zeit.
Cover Time: $< 4 \cdot |V| \cdot |E| \leq 4 \cdot |V| \cdot \frac{|V| \cdot (|V|-1)}{2} \leq 2 \cdot |V|^3$.

Algorithmus $s-t$–Connectivity
starte einen Random Walk bis
for $i := 1$ to $4 \cdot |V|^3$ do
    wechsle den Knoten über eine Kante
    if t erreicht then return JA endif
end
return NEIN

Satz: $s-t$–Connectivity gibt mit Wahrscheinlichkeit von mindestens $\frac{1}{2}$ die korrekte Antwort und kann nur im NEIN-Fall irren.

Beweis: Die Anzahl der Schritte ist klar wegen der Cover Time (doppelte Covertime!).

$$\begin{aligned}
&\Pr\left[\text{Algorithmus findet in } 4\cdot|V|\cdot|E| \text{ Schritten einen Weg von } s \text{ nach } t\right] \\
=\ &1-\Pr\left[\text{Algorithmus findet in } 4\cdot|V|\cdot|E| \text{ Schritten keinen Weg von } s \text{ nach } t\right] \\
=\ &1-\Pr\left[\text{Algorithmus braucht } > 4\cdot|V|\cdot|E| \text{ Schritte}\right] \\
=\ &1-\Pr\left[\text{Algorithmus braucht } > 2\cdot 2\cdot|V|^3 \text{ Schritte}\right] \geq^{(\text{Markov-Ungleichung})} \geq 1-\frac{1}{2}=\frac{1}{2}
\end{aligned}$$

□

Platz: $\Theta(\log|V|)$. Zeit: $\Theta(|V|)^3$.

5. Counting und die Monte-Carlo-Methode

5.1. Kombinatorische Zählprobleme und $\#P$–Vollständigkeit

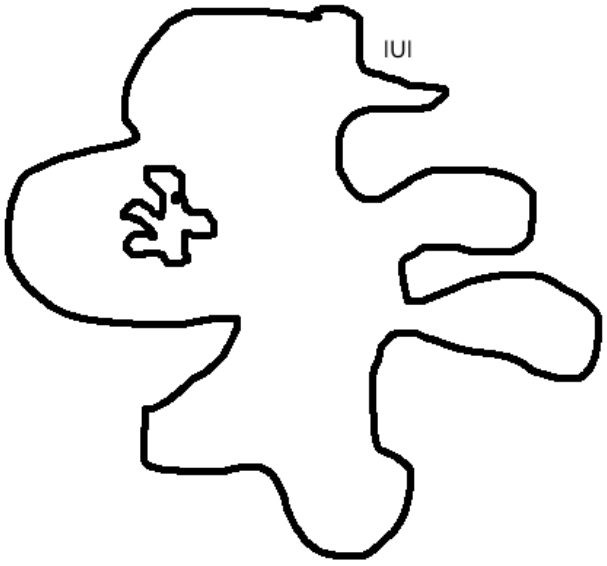
Def.: Bei einem kombinatorischen Zählproblem $\#\Pi$ ist ein kombinatorisches (Optimierungs-)Problem Π gegeben. Die Aufgabe besteht darin, zur Instanz I die Anzahl $\#(I)$ der zulässigen Lösungen zu bestimmen.

Def.:

- $\#DNF$: Bei $\#DNF$ ist die Probleminstanz Ψ eine Boolesche Formel in Disjunktiver Normalform über den Variablen $V=\{x_1,...,x_n\}$, d. h. $\Psi=C_1\vee...\vee C_m$ und die Monome C_i sind Und-Formeln aus Literalen aus Variablen aus V.
  Eine zulässige Lösung ist eine Belegung der Variablen, die Ψ erfüllt. Gesucht ist die Anzahl der Ψ erfüllenden Belegungen.
- $\#COL_k$: $k\in\mathbb{N}$, $\#COL_k$ ist das Knotenfärbungsproblem.
  Ziel ist es, die Anzahl der korrekten Färbungen zu bestimmen, die höchstens k Farben benutzen.

$\#P$ ist die Klasse der Zählprobleme, für die $\#(I)$ bei Instanz I polynomiellen Platz in $|I|$ zum Hinschreiben braucht. #SAT ist $\#P$–vollständig. Die "klassische" Masterreduktion beweist das bereits.

5.1.1. Relative Güte, die Expansion und die Wahl des Universums



$\frac{1}{|U|}$: Wahrscheinlichkeit für jeden Punkt in U, ausgewählt zu werden.

Def.: Ein Algorithmus A für $\#\Pi$ hat bei Eingabe I die individuelle relative Güte $\rho_A(I) = \max\left\{\frac{A(I)}{\#(I)}, \frac{\#(I)}{A(I)}\right\}$

Def.: Sei I eine Eingabe von $\#\Pi$. Sei U_I eine Obermenge der Menge $S(I)$ der zu I zulässigen Lösungen. U_I wird Universum zu $S(I)$ genannt. Zudem sei $|U_I|$ bekannt. ($\#(I) = |S(I)|$ ist gesucht, $S(I) \subseteq U_I$.) $\xi = \frac{|U_I|}{\#(I)}$ ist die Expansion des Universums.

Satz: Sei s eine Abschätzung für ξ mit $\xi \leq s$. Dann approximiert $A(I) := \frac{|U_I|}{\sqrt{s}}$ den gesuchten Wert $\#(I)$ mit der relativen Güte $\sqrt{s}$.

Beweis: Übung □

#DNF hat zwei wichtige Eigenschaften:

- erfüllende Belegungen sind ganz einfach zu finden, da es ausreicht, ein einzelnes Monom zu erfüllen.
- wir können für ein einzelnes Monom C die Zahl $\#(C)$ unmittelbar berechnen.

Lemma: Sei $C = l_1 \vee ... \vee l_k$ ein Monom der Booleschen (n,m)–Formel Ψ in DNF aus k Literalen. Dann gibt es genau 2^{n-k} Variablenbelegungen, die C erfüllen.

Sei k^* die Länge eines kürzesten Monoms in Ψ. Dann gilt: $\#(\Psi) \geq 2^{n-k^*}$.
Sei $U_\Psi^{\text{blind}} = \{u \,|\, u \text{ ist Belegung der Variablen}\}$. $\left|U_\Psi^{\text{blind}}\right| = 2^n$.
Die Expansion von U_Ψ^{blind} ist $\xi_{\text{blind}} = \frac{|U_\Psi^{\text{blind}}|}{\#(\Psi)} \leq \frac{2^n}{2^{n-k^*}} = 2^{k^*}$.
$A_1(\Psi) = \frac{2^n}{\sqrt{2^{k^*}}} = 2^{n-\frac{k^*}{2}}$ approximiert $\#(\Psi)$ mit relativer Güte $2^{\frac{k^*}{2}} = \left(\sqrt{2}\right)^{k^*}$.

$\Psi_{\text{bad}} = x_1 \wedge ... \wedge x_{\frac{n}{2}}$ bei n Variablen. $k^* = \frac{n}{2}$, $\#(\Psi_{\text{bad}}) = 2^{\frac{n}{2}}$.
$\xi_{\text{blind}} = 2^{\frac{n}{2}}$, $A_1(\Psi_{\text{bad}}) = 2^{n-\frac{n}{2}\cdot\frac{1}{2}} = 2^{\frac{3}{4}n}$; $\rho_{A_n}(\Psi_{\text{bad}}) = 2^{\frac{n}{4}}$.

$$\Psi = C_1 \vee ... \vee C_m.$$

$$\begin{aligned} S(\Psi) &= \{u \,|\, u \text{ erfüllt } \Psi\} \\ &= \bigcup_{j=1}^{m} \{u \,|\, u \text{ erfüllt } C_j\} \\ &= \bigcup_{j=1}^{m} \{u \,|\, u \text{ erfüllt } C_j, \text{ aber kein } C_k \text{ mit } k < j\} \\ S'(\Psi) &= \bigcup_{j=1}^{m} \{(u,j) \,|\, u \text{ erfüllt } C_j, \text{ aber kein } C_k \text{ mit } k < j\} \end{aligned}$$

$\#(\Psi) = |S(\Psi)| = |S'(\Psi)|$

$$U_\Psi^{\text{clever}} = \{(u,j) \,|\, u \text{ erfüllt } C_j\} = \dot{\bigcup_{j=1}^{m}} \{(u,j) \,|\, u \text{ erfüllt } C_j\}$$

Es ist $S'(\Psi) \subseteq U_\Psi^{\text{clever}}$. $\left|U_\Psi^{\text{clever}}\right| = \sum_{j=1}^{m} 2^{n-k_j}$, k_j die Länge des Monoms C_j.
Die in U_Ψ^{clever} vorkommenden Belegungen der Variablen sind alle erfüllend.

Lemma (Expansionslemma):

$$\xi_{\text{clever}} = \frac{|U_\Psi|}{|S'(\Psi)|} \leq m$$

Beweis: Eine Ψ erfüllende Belegung kommt in $S'(\Psi)$ nur einmal vor, in U_Ψ^{clever} aber bis zu m Mal. □

$A_2(\Psi) = \frac{1}{\sqrt{m}} \cdot \sum_{j=1}^{m} 2^{n-k_j}$ approximiert $\#(\Psi)$ mit relativer Güte $\sqrt{m}$.

5.1.2. Randomisierte Approximationsschemata und Wahrscheinlichkeitsverstärkung

Def.: Sei $\#\Pi$ ein kombinatorisches Zählproblem. Sei A ein Algorithmus, der als Eingabe eine Instanz I von $\#\Pi$ und ein ϵ, $0 < \epsilon < 1$, und die Zahl $A(I, \epsilon)$ berechnet.

- A ist ein polynomielles Zähl-Approximationsschema (P(T)ASC, polynomial (time) approximation scheme for counting) für $\#\Pi$, falls A deterministisch ist, polynomielle Laufzeit in $|I|$ hat und gilt: $|A(I, \epsilon) - \#(I)| \leq \epsilon \cdot \#(I)$. $\left(\text{hier noch erlaubt: } \mathcal{O}\left(|I|^{\frac{1}{\epsilon}}\right)\right)$
- A ist ein streng polynomielles Zähl-Approximationsschema (FP(T)ASC, fully P(T)ASC), falls A ein PASC ist und die Laufzeit polynomiell in $|I|$ und ϵ ist. (hier erlaubt: $\mathcal{O}\left(|I|^4 \cdot \left(\frac{1}{\epsilon}\right)^7\right)$.
- A ist ein polynomielles randomisiertes Zähl-Approximationsschema (PRASC), falls die Laufzeit polynomiell in $|I|$ ist und gilt:

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

- A ist ein streng polynomielles Zähl-Approximationsschema (FPRASC), falls A ein PRASC und die Laufzeit polynomiell in $|I|$ und $\frac{1}{\epsilon}$ ist.
- A ist ein (ϵ, δ)–FPRASC, wenn A ein FPRASC ist, zusätzlich die Eingabe δ, $0 < \delta < 1$ bekommt, und in Zeit polynomiell in $|I|$, $\frac{1}{\epsilon}$, $\log\left(\frac{1}{\delta}\right)$ die Ausgabe $A(I, \epsilon, \delta)$ berechnet mit

$$\Pr\left(|A(I, \epsilon, \delta) - \#(I)| \leq \epsilon \cdot \#(I)\right] \geq 1 - \delta$$

Algorithmus Ampl
input: A, ϵ, δ, I
for $\tau := 1$ to τ_δ do
    $N_\tau := A(I, \epsilon)$
end
gib den Median von N_1, ..., N_{τ_δ} aus.

Satz: Sei $\#\Pi$ ein kombinatorisches Zählproblem und A ein FPRASC für $\#\Pi$. Sei $\delta < 1$. Mit $\tau_\delta = 8 \cdot \left\lceil \ln\left(\frac{1}{\delta}\right) \right\rceil$ ist $\text{Ampl}(A, \epsilon, \delta, I)$ ein (ϵ, δ)–FPRASC für $\#\Pi$.

Beweis: Übung □

5.1.3. Die klassische Monte-Carlo-Methode

$\#\Pi$; Zu I von $\#\Pi$ ist $\#(I)$ die gesuchte Anzahl der zu I zulässigen Lösungen, d. h., $\#(I) = |S(I)|$. U_I ist das Universum, aus dem die zulässigen Lösungen stammen, also $S(I) \subseteq U_I$. $|U_I|$ ist bekannt! $\xi = \frac{|U_I|}{\#(I)}$ ist die Expansion des Universums. Sei $\chi : U_I \to \{0, 1\}$ die charakteristische Funktion zu $S(I)$, d. h.:

$$\chi(u) = \begin{cases} 1 & \text{falls } u \in S(I) \\ 0 & \text{sonst} \end{cases} \quad \text{für alle } u \in U_I$$

Um die klassische Monte-Carlo-Methode anwenden zu können, fordern wir:

- Es gibt einen randomisierten Algorithmus UG (Uniformer Generator), der in Polynomzeit in $|I|$ Elemente $u \in U_I$ generiert, sodass für alle $u \in U_I$ gilt: $\Pr[u \text{ wird von UG ausgegeben}] = \frac{1}{|U_I|}$.
- Es gibt einen deterministischen Algorithmus BEANTWORTER, der χ in Polynomzeit in $|I|$ berechnet.

input: T
output: Z, Schätzung für $\#(I)$
for $i := 1$ to T do
    ziehe eine Stichprobe u mittels UG
    $Y_i := \chi(u)$ mittels BEANTWORTER
done
$R := \frac{1}{T} \cdot \sum_{i=1}^{T} Y_i$
return $Z := R \cdot |U_I|$

Lemma:

a) $E[Y_i] = \frac{1}{\xi}$; $E[R] = \frac{1}{\xi}$

b) $E[MC(T)] = E[Z] = \#(I)$

Lemma:

a) $\mathrm{Var}[R] = \frac{\xi^{-1} \cdot (1-\xi^{-1})}{T}$

b) $\mathrm{Var}[MC(T)] = |U_I|^2 \cdot \frac{\xi^{-1} \cdot (1-\xi^{-1})}{T}$

Beweis:

a) $\mathrm{Var}[Y_i] = E\left[Y_i^2\right] - E[Y_i]^2 = E[Y_i] \cdot (1 - E[Y_i]) = \xi^{-1} \cdot \left(1 - \xi^{-1}\right)$
$\mathrm{Var}[R] = \mathrm{Var}\left[\frac{1}{T} \sum_{i=1}^{T} Y_i\right] = \frac{1}{T^2} \cdot \sum_{i=1}^{T} \mathrm{Var}[Y_i] = \frac{\xi^{-1} \cdot (1-\xi^{-1})}{T}$.

b) $\mathrm{Var}[MC(T)] = \mathrm{Var}[R \cdot |U_I|] = |U_I|^2 \cdot \frac{\xi^{-1} \cdot (1-\xi^{-1})}{T}$.

Satz (Estimator Theorem der Monte-Carlo-Methode): Sei $\epsilon > 0$ beliebig. Mit $T_\xi(\epsilon) = \left\lceil \frac{4}{\epsilon^2} \cdot (\xi - 1) \right\rceil$ gilt:

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

Beweis: Tschebyscheff liefert:

$$\Pr[|MC(T_\xi(\epsilon)) - \#(I)| \geq \epsilon \cdot \#(I)] \leq \frac{1}{\epsilon^2} \cdot \frac{\mathrm{Var}[z]}{E[Z]^2} = \ldots = \frac{\xi - 1}{\epsilon^2} \cdot \frac{1}{T_\xi(\epsilon)} \leq \frac{1}{4}.$$

□

Achtung:

- Laufzeit ist linear in ξ. Wenn die Expansion groß ist, ist es auch die Laufzeit.
- ξ hängt von der gesuchten Zahl ab, muss also nach oben "klein" abgeschätzt werden.

#DNF und Importance Sampling

- Uniformer Generator: Jedes Element aus U mit Wahrscheinlichkeit $\frac{1}{|U|}$ ausgeben können.
- Beantworter: Für $u \in U$ in Polynomzeit entscheiden, ob $u \in S(I)$ (die Menge, deren Kardinalität wir suchen) ist.

$\Psi = C_1 \vee \ldots \vee C_m$.
$U_\Psi^{\text{blind}} = \{u \,|\, u \text{ ist Belegung der Variablen aus } \Psi\}$.
U_Ψ^{blind} kann exponentielle Expansion haben. Die klassische Monte-Carlo-Methode benötigt also auch exponentielle Laufzeit.
$U_\Psi^{\text{clever}} = \{(u,j) \,|\, u \text{ erfüllt } C_j\}$.
$S'(\Psi) = \{(u,j) \,|\, u \text{ erfüllt } C_j \text{ und kein } c_i, i < j\} \Rightarrow |S(\Psi)| = |S'(\Psi)|; \xi_{\text{clever}} \leq m$.

Kürzere Monome haben größeren Anteil an den U_Ψ^{clever} als längere. Beim Stichprobenziehen müssen die kürzeren Monome daher auch häufiger drankommen.

$\mathrm{UG}_{\mathrm{DNF}}$
- würfele ein $j \in \{1, \ldots, n\}$ mit $\frac{2^{n-k_j}}{|U_\Psi^{\text{clever}}|}$
- setze die Variablen in C_j so, dass C_j erfüllt wird
- würfele eine zufällige Belegung der restlichen, d. h. der nicht in C_j vorkommenden Variablen (gleichverteilt)
- gib j und die berechnete Belegung aus

Lemma: $\mathrm{UG}_{\mathrm{DNF}}$ ist ein uniformer Generator.

Beweis:

$$\Pr\left[(u,j) \in U_\Psi^{\text{clever}} \text{ wird ausgegeben}\right] = \Pr\left[\text{die } C_j \text{ erfüllende Belegung wird ausgegeben und } j \in \{1,...,m\} \text{ wurde gewählt}\right]$$
$$= \frac{2^{n-k_j}}{\left|U_\Psi^{\text{clever}}\right|} \cdot \frac{1}{2^{n-k_j}} = \frac{1}{\left|U_\Psi^{\text{clever}}\right|}$$

□

Satz: Mit $S'(\Psi)$ für BEANTWORTER und U_Ψ^{clever} (für UG) gilt:
Mit $T(\epsilon,\delta) = \left\lceil n \cdot \frac{4}{\epsilon^2} \cdot \ln\left(\frac{2}{\delta}\right)\right\rceil$ ist der Algorithmus $\text{MC}(T(\epsilon,\delta))$ ein (ϵ,δ)–FPRASC für #DNF der Laufzeit $\mathcal{O}\left(m^2 \cdot n \cdot \frac{1}{\epsilon^2} \cdot \log\frac{1}{\delta}\right)$.

Fundamentale Beziehung zwischen uniformem Sampling und Zählen.

Die Markov-Ketten-Monte-Carlo-Methode (MCMC)

Hilfsmittel: Random Walks auf regulären Graphen.
Sei $G = (V,E)$ ein zusammenhängender ungerichteter Graph mit Schleifen (self loops) und M ein Random Walk mit $m_{\text{u,w}} = \frac{1}{\Delta_G(u)}$. Ist G regulär, d. h., alle Knoten haben den gleichen Grad, dann ist die stationäre Verteilung (falls sie existiert) $\pi = \left(\frac{1}{|V|},...,\frac{1}{|V|}\right)$. D. h.: Jeder Knoten von V ist im Unendlichen mit gleicher Wahrscheinlichkeit der Aufenthaltsort.

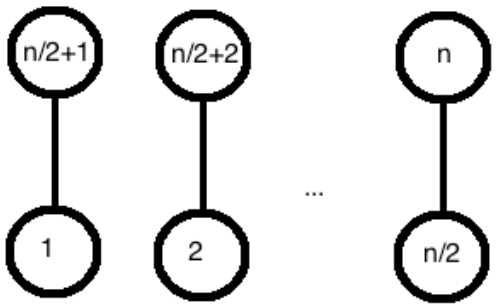
Um einen Knoten zu G mit Wahrscheinlichkeit $\frac{1}{|V|}$ zu samplen, lassen wir die Markov-Kette M "unendlich lange" laufen und geben den "gefundenen" Knoten aus. Wir interessieren uns dafür, wie lange es dauert, bis wir "nahe" an der stationären Verteilung sind.

Abstandsmaß: Variationsabstand $\delta_u(t) = \frac{1}{2}\sum_{w\in V}\left|\Pr\left[M \text{ gestartet in } u \text{ führt in } t \text{ Schritten zu } w\right] - \pi_W\right|$.

Def.: Sei M ein ergodischer (konvergierender) Random Walk auf $G = (V,E)$. Sei $t(|V|,\epsilon_0)$ so, dass für alle $u \in V$ und alle $t \geq t(|V|,\epsilon_0)$ gilt: $\delta_u(t) \leq \epsilon_0$. Die Zeit $t(|V|,\epsilon_0)$ heißt Mischzeit. Ist $t(|V|,\epsilon_0) = \text{poly}\left(\log|V|, \frac{1}{\epsilon_0}\right)$, so ist M eine schnell mischende Markov-Kette (rapidly mixing Markov chain). Da G üblicherweise als Knoten das Universum enthält, das üblicherweise exponentiell groß ist, ist das $\log|V|$ erst polynomiell in der Eingabegröße.

Zählen zulässiger Knotenfärbungen

$\#\text{COL}_k$: Gegeben ist ein Graph. Gesucht: Die Anzahl der verschiedenen Knotenfärbungen mit maximal k Farben.



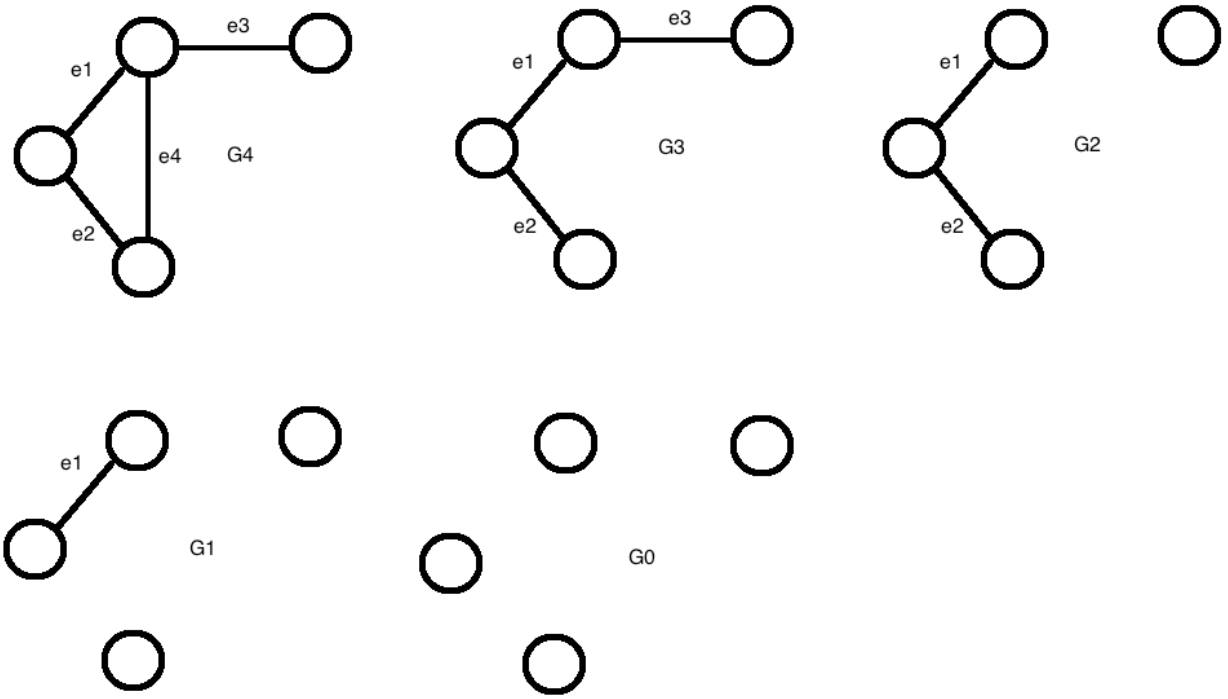
$k = 3 \Rightarrow U_{\text{blind}} = \{c | c : V \to \{1,2,3\}\}, |U_{\text{blind}}| = 3^n$.
$\#(G_{\text{exp}}) = \left(2 \cdot \binom{k}{2}\right)^{\frac{n}{2}} = 6^{\frac{n}{2}}$; Expansion: $\xi_{\text{blind}} = \frac{|U_{\text{blind}}|}{\#(G_{\text{exp}})} = \frac{3^n}{6^{\frac{n}{2}}} \approx (1,22)^n$.
Sei nun immer $k \geq 2 \cdot \Delta(G) + 1$.

Markov Chain → benutzt zum Stichprobenziehen

Monte-Carlo-Methode: → wie viele Stichproben? ⇒ MCMC

$G = (V, E)$ sei der Eingabegraph, $E = \{e_1, ..., e_m\}$.

- $G_m = G = (V, E)$
- $G_i = (V, E_i)$, wobei $E_i = E_{i+1} \setminus \{e_{i+1}\}$
- $G_0 = (V, \varnothing)$



Gesucht: $\#(G_m)$.
Wir lösen: $\#(G_m)$, $\#(G_{m-1})$, $\#(G_{m-2})$, ..., $\#(G_1)$, $\#(G_0)$. $S(G_m) \subseteq S(G_{m-1}) \subseteq ... \subseteq S(G_1) \subseteq S(G_0)$.
Wir verwenden $S(G_i)$ als Universum für $S(G_{i+1})$.

$$\#(G) = \#(G_m) = \frac{\#(G_m)}{\#(G_{m-1})} \cdot \#(G_{m-1}) = ... = \frac{\#(G_m)}{\#(G_{m-1})} \cdot \frac{\#(G_{m-1})}{\#(G_{m-2})} \cdot ... \cdot \#(G_0) = \prod_{i=1}^{m} \xi_i^{-1} \cdot k^n.$$

Lemma: Für alle $i \in \{1, ..., m\}$ gilt: $\frac{\#(G_i)}{\#(G_{i+1})} = \xi_i \leq 2$.

Algorithmus FÄRBUNGEN_ZÄHLEN(G, ϵ)
input: $G = (V, E)$
output: C: Schätzung für $\#(G)$
$m := |E|$; $n := |V|$; $G_m := G$;
for $i := m$ downto 1 do
    G_{i-1} := lösche Kante e_i in G_i
    R_i := VERHÄLTNIS(i) (Monte-Carlo-Methode)
end
$Z := \prod_{i=1}^{m} R_i$
return $C := Z \cdot k^n$

Algorithmus VERHÄLTNIS(i)
output: R_i: Schätzung für ξ_i^{-1}
for $t := 1$ to T do
    "Ziehe eine Färbung c von G_{i-1}" ← Markov Chain ($\rightarrow \text{MARKOV}_t(G_{i-1})$)

    if $c \in S(G_i)$ then $X_t^{(i)} := 1$ else $X_t^{(i)} := 0$;
end
return $R_i := \frac{1}{T} \sum_{t=1}^{T} X_t^{(i)}$

Algorithmus MARKOV$_t(G)$ (diese Markov-Kette ist schnell mischend, das liefert ein "kleines" t)
input: $G = (V, E)$ Graph
output: c: zulässige Färbung von G (mit höchstens k Farben)
$c :=$ erzeuge beliebige zulässige Färbung von G_i
for $j := 1$ to t do
    würfele $u \in V$ und eine Farbe $f \in \{1, ..., k\}$
    $c' :=$ ersetze in c die Farbe von u durch f
    if c' ist zulässige Färbung von G
    then $c := c'$
end
gib c aus

Im "Konfigurationsgraphen", der aus allen Färbungen besteht, ist die Übergangswahrscheinlichkeit für jeden Knoten zu jedem anderen Knoten gegeben durch $\frac{1}{|V| \cdot k}$. Der Gesamtkonfigurationsgraph ist regulär vom Grad $|V| \cdot k$ (inklusive der self-loops).
$\Rightarrow$ im Unendlichen ist jeder Knoten gleich wahrscheinlich!

$$S(G_i) \subset S(G_{i-1}) \Rightarrow \frac{\#(G_i)}{\#(G_{i-1})}.$$

Des Weiteren: $T := \left\lceil \frac{74 \cdot m}{\epsilon^2} \right\rceil$; $t = \left\lceil \frac{k - \Delta(G)}{k - 2 \cdot \Delta(G)} \cdot n \cdot \ln \frac{6 \cdot n \cdot m}{\epsilon} \right\rceil$
$\Rightarrow$ damit (ϵ, δ) –FPRASC für #COL$_k$.