Graphenalgorithmen
Blatt 6

Markus Vieth   Christian Stricker

14. Dezember 2016

1 Aufgabe 1

Pseudocode Sei `E` das Ergebnis des Monte Carlo Algorithmus, `I` die Eingabe und `boolean test(Ergebnis E)` der Algorithmus, welcher das Ergebnis überprüft.

```
Ergebnis E;
do {
E = MonteCarloAlgorithmus(I);
}while(!test(E));
return E;
```

Laufzeit Sei $q(n) = 1 - p(n)$ und $P(X = i) := q(n)^{i-1} \cdot p(n)$ die Wahrscheinlichkeit, dass der Monte Carlo Algorithmus nach i Durchläufen das richtige Ergebnis liefert (also zuerst $i - 1$ falsche und dann ein richtiges). Dann beträgt die zu erwartende Anzahl an Durchläufen:

$$E(X) = \sum_{i=1}^{\infty} i \cdot q(n)^{i-1} \cdot p(n) = {}^{1}p(n) \cdot \sum_{i=1}^{\infty} i \cdot q(n)^{i-1} = {}^{2}p(n) \cdot \frac{1}{(1 - q(n))^2} = \frac{p(n)}{(1 - 1 + p(n))^2} = \frac{1}{p(n)}$$

Sei $T_{LV}(n)$ die Laufzeit des oben beschriebenen Algorithmus:

$$\Rightarrow T_{LV}(n) = E(X) \cdot (T(n) + t(n)) = \frac{T(n) + t(n)}{p(n)}$$

Sei X die Anzahl der Durchläufe $X \sim \mathrm{geom}_p \Rightarrow E[X] = \frac{1}{p(n)}$

$$P(X = k) = (1 - p)^{k-1} \cdot p$$

$$E[X] = \sum_{k=1}^{\infty} (1-p)^{k-1} pk = p \sum_{k=1}^{\infty} (1-p)^{k-1} k = p \sum_{n=0}^{\infty} \frac{-d}{dp} (1-p)^k = -p \frac{d}{dp} \sum_{n=0}^{\infty} (1-p)^k = -p \frac{d}{dp} \frac{1}{p} = \frac{1}{p}$$

q.e.d.

2 Aufgabe 2

2.1 Pseudocode

```
procedure contract(G = (V, E), t):
    while |V| > t
        choose e ∈ E uniforly at random
        G ← E \ {e}
    return G

findMin3Cut(G = (V, E))
    (V,E) = contract(G,3)
    return E

min(E_1, E_2)
    return E_1 < E_2 ? E_1 : E_2

betterFindMin3Cut(G = (V, E))
    return min(findMin3Cut(G), findMin3Cut(G))
```

2.2 Max Anzahl an Kanten in einem 3-Cut

z.z. max Anzahl ist $\left(1 - (1 - \frac{2}{n}) \left(1 - \frac{2}{n-1}\right)\right) m$

[1] $\sum_{i=1}^{\infty} i \cdot q(n)^{i-1} \cdot p(n)$ ist absolut konvergent, wenn $0 < p(n) < 1$, kann per Quotientenkriterium gezeigt werden

[2] Ableitung der geometrischen Reihe

Beweis Angenommen wir wählen 2 Knoten zufällig gleichverteilt. Wir nehmen nur an das die beiden Knoten je einen Teil des 3-Cuts ausmachen und die restlichen Knoten den 3. Teil. Eine Kante gehört zum 3-Cut, wenn sie in verschiedenen so definierten Partitionen endet. Die Wahrscheinlichkeit, dass eine Kante nun zum 3-Cut gehört ist die Wahrscheinlichkeit, dass einer der Endpunkte in einer der 2 vorher gewählten Knoten ist, also:

$$P = 1 - \left(1 - \frac{2}{n}\right)\left(1 - \frac{2}{n-1}\right)$$

f sein nun die erwartete Zahl an Kanten im 3-Cut und m die Anzahl an Kanten im Graphen, dann gilt:

$$E[f] = \left(1 - \left(1 - \frac{2}{n}\right)\left(1 - \frac{2}{n-1}\right)\right) m$$

Da es nicht weniger Kanten in einem 3-Cut geben kann als im minimalen 3-Cut, ist auch $E[f]$ größergleich einem minimalen 3-Cut

2.3 Wahrscheinlichkeit zum finden eines min 3-Cuts mit einer Iteration

parallel zur Vorlesung gilt:

$$\begin{aligned} P &= \prod_{i=0}^{n-4}\left(\left(1 - \frac{2}{i}\right)\left(1 - \frac{2}{n-i-1}\right)\right) \\ &= \prod_{i=0}^{n-4}\left(1 - \frac{2}{n-i}\right)\prod_{i=0}^{n-4}\left(1 - \frac{2}{n-i-1}\right) \\ &= \binom{3}{2}\binom{n}{2}^{-1}\binom{n-1}{2}^{-1} \end{aligned}$$

3 Aufgabe 3

3.1 Wahrscheinlichkeit für 1 Iteration falsch

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

3.2 Wahrscheinlichkeit für 2 Iterationen richtig

$$P(n) = 1 - \left(1 - \left(1 - \left[1 - P\left(\left\lceil 1 + \frac{n}{\sqrt{2}}\right\rceil\right)\right]^2\right)\right)^2 = 1 - \left[1 - P\left(\left\lceil 1 + \frac{n}{\sqrt{2}}\right\rceil\right)\right]^4$$

4 Aufgabe 4

Wahrscheinlichkeit zum finden eines Min-Cut:

$$P(n) = \prod_{i=1}^{n-2}\frac{n-i-1}{n-i+1} = \frac{\prod_{i=1}^{n-2} n-i-1}{\prod_{j=1}^{n-2} n-j+1} = \frac{\overbrace{2}^{n-3}\cdot\overbrace{1}^{n-2}}{(n)(n-1)}\cdot\frac{\prod_{i=1}^{n-4} n-i-1}{\prod_{j=3}^{n-2} n-j+1}$$

$$= \frac{2}{n(n-1)}\cdot\frac{\prod_{i=1}^{n-4} n-i-1}{\prod_{i=1}^{n-4} n-(i+2)+1} = \frac{2}{n(n-1)}\cdot\prod_{i=1}^{n-4}\frac{n-i-1}{n-i-1} = \frac{2}{n(n-1)}\cdot 1$$

Seien $C_1, \dots, C_k$ Min-Cuts in G. Sei D_i das Event, dass der Algorithmus C_i findet. Da alle D_i unabhängig sind, gilt

$$\sum_{i=1}^{k} P(D_i) \leq 1$$

mit der Rechnung von oben gilt:

$$\Rightarrow P(D_i) = P(n) \Rightarrow \sum_{i=1}^{k} \frac{2}{n(n-1)} \leq 1 \Rightarrow k = \frac{n(n-1)}{2}$$

q.e.d.