Exercise T4.1 – *Recursive dice*

Consider the following experiment: We start by rolling a (fair, six sided) die. Let X be the random variable that equals the number the first die shows. Next, we roll a fair X sided die (that is a die with numbers 1 to X on it, each of which is equally likely to appear). Let Y be the random variable that equals the number shown by this second die.

(a) Compute the joint distribution (gemeinsame Dichte) of X and Y.

(b) Compute the marginal distribution (Randdichte) of Y.

(c) What are the expected value and variance of Y?

a) $f_{Y,X}(x,y) = \begin{cases} \frac{1}{6x} & \text{falls } 1 \le y \le x \le 6 \\ 0 & \text{sonst} \end{cases}$

$E[X] = \sum_{x \in W_X} Pr[X=x]$

b) $f_Y(y) = \sum_{x=y}^{6} f_{X,Y}(x,y) = \sum_{x=y}^{6} \frac{1}{6x}$

c) $Var[Y] = E[(E[Y]-Y)^2] = E[Y^2] - E[Y]^2$

$= \frac{275}{144}$

Exercise T4.2 – *Upper Bounds*

We toss a fair coin $n \ge 1$ times. Let X be the number of times the coin shows "heads". provide upper bounds on $\Pr[X \ge 0.75\,n]$ using the following tools.

(a) Use Markov's inequality.

(b) Use Chebychev's inequality.

(c) Use Chernoff's bound.

Make sure to justify, why the respective inequality can be applied!

a) $Pr[X \ge t] \le \frac{E[X]}{t}$, X non-negative

$Pr[X \ge \frac{3}{4}n] \le \frac{\frac{1}{2}n}{\frac{3}{4}n} = \frac{2}{3}$

b) $Var[X] = np(1-p)$ ← Binomialverteilung

$Pr[X \ge \frac{3}{4}n] = Pr[X - E[X] \ge \frac{1}{4}n]$

$\le Pr[|X - E[X]| \ge \frac{1}{4}n]$

$\le \frac{Var[X]}{(\frac{1}{4}n)^2} = \frac{4}{n}$

c) $Pr[X \ge (1+\delta)E[X]] \le e^{-\frac{1}{3}\delta^2 E[X]}$, X Summe von Bernoullivariablen

$Pr[X \ge \frac{3}{4}n] = Pr[X \ge (1+\underbrace{\frac{1}{2}}_{\delta})\underbrace{\frac{n}{2}}_{E[X]}] \le e^{-\frac{1}{3}\cdot\frac{1}{4}\cdot\frac{n}{2}} = e^{-\frac{n}{24}}$

Schranken

Satz 2.67. *(Ungleichung von Markov)* Sei X eine Zufallsvariable, die nur nicht-negative Werte annimmt. Dann gilt für alle $t \in \mathbb{R}$ mit $t > 0$, dass

$$\Pr[X \geq t] \leq \frac{\mathbb{E}[X]}{t}.$$

Oder äquivalent dazu $\Pr[X \geq t \cdot \mathbb{E}[X]] \leq 1/t$.

Satz 2.68. *(Ungleichung von Chebyshev)* Sei X eine Zufallsvariable und $t \in \mathbb{R}$ mit $t > 0$. Dann gilt

$$\Pr[|X - \mathbb{E}[X]| \geq t] \leq \frac{\mathrm{Var}[X]}{t^2}$$

oder äquivalent dazu $\Pr[|X - \mathbb{E}[X]| \geq t\sqrt{\mathrm{Var}[X]}] \leq 1/t^2$.

Satz 2.70 (Chernoff-Schranken). Seien $X_1, \ldots, X_n$ unabhängige Bernoulli-verteilte Zufallsvariablen mit $\Pr[X_i = 1] = p_i$ and $\Pr[X_i = 0] = 1 - p_i$. Dann gilt für $X := \sum_{i=1}^{n} X_i$:

(i) $\Pr[X \geq (1+\delta)\mathbb{E}[X]] \leq e^{-\frac{1}{3}\delta^2 \mathbb{E}[X]}$ für alle $0 < \delta \leq 1$,

(ii) $\Pr[X \leq (1-\delta)\mathbb{E}[X]] \leq e^{-\frac{1}{2}\delta^2 \mathbb{E}[X]}$ für alle $0 < \delta \leq 1$,

(iii) $\Pr[X \geq t] \leq 2^{-t}$ für $t \geq 2e\mathbb{E}[X]$.

Randomisierte Algorithmen

randomisierter Algorithmus: Eingabe $I \rightarrow$ Algorithmus A mit Zufallszahlen $R \rightarrow$ Ausgabe $A(I,R)$

deterministisch: selbe Eingabe, selber Output
nicht-deterministisch: selbe Eingabe, vielleicht unterschiedlicher Output

Monte Carlo	vs	Las Vegas
ZV: Korrektheit		ZV: Laufzeit
immer gleiche Laufzeit		manchmal zu langsam / "???" ausgeben
manchmal falsches Ergebnis		immer korrekte Antwort
Beispiele: Primzahltest, Target-Shooting		Beispiele: Quicksort

Reduktion von Fehlerwahrscheinlichkeiten

① Las Vegas

nie falsche Antwort, aber manchmal "???" als Ausgabe
$Pr[\mathcal{A}(I) \text{ korrekt}] \geq \varepsilon$

Es gilt $\forall \delta > 0$, $\mathcal{A}_\delta$ ein Algo welcher $\mathcal{A}$ solange aufruft bis richtige Antwort oder $N = \frac{1}{\varepsilon} \ln(\frac{1}{\delta})$ erfolglose Aufrufe, dann

$$Pr[\mathcal{A}_\delta(I) \text{ korrekt}] \geq 1-\delta$$

ϵ	δ	N
0.1	0.01	47
0.5	0.01	10
0.5	10^{-80}	369
0.9	10^{-30}	77

Beweis:

$$Pr[\mathcal{A}_\delta \text{ gibt } N\text{-mal "???" aus}] \leq (1-\varepsilon)^N \overset{1-x \leq e^{-x}}{\leq} e^{-\varepsilon N} = e^{-\varepsilon \cdot \frac{1}{\varepsilon} \cdot \ln(\delta^{-1})} = e^{\ln(\delta)} = \delta$$

$$Pr[\mathcal{A}_\delta \text{ korrekt}] \geq 1-\delta$$

② Monte Carlo (einseitiger Fehler)

$\mathcal{A}$ rand. Algo. mit Ausgabe $\in \{\text{Ja}, \text{Nein}\}$

$Pr[\mathcal{A}(I) = \text{Ja}] = 1$ falls I Ja-Instanz
$Pr[\mathcal{A}(I) = \text{Nein}] \geq \varepsilon$ falls I Nein-Instanz

$\forall \delta > 0$ $\mathcal{A}_\delta$ = Algo., welcher $\mathcal{A}$ solange aufruft, bis Nein oder $N = \frac{1}{\varepsilon} \ln(\frac{1}{\delta})$ - mal Ja ausgegeben wird

Dann gilt: $Pr[\mathcal{A}_\delta(I) \text{ korrekt}] \geq 1-\delta$

Beweis analog zu ①

③ Monte Carlo (zweiseitiger Fehler)

$\mathcal{A}$ rand. Algo. mit Ausgabe $\in \{Ja, Nein\}$

$\Pr[\mathcal{A}(I) \text{ korrekt}] \geq \frac{1}{2} + \varepsilon$

$\forall \delta > 0$ $\mathcal{A}_\delta$:= Algo, welcher $\mathcal{A}$ $N = \frac{4}{\varepsilon^2} \ln(\frac{1}{\delta})$ - mal aufruft und die Mehrheit der erhaltenen Antworten ausgibt

Dann gilt: $\Pr[\mathcal{A}_\delta(I) \text{ korrekt}] \geq 1 - \delta$

Aufgabe: Broken Servers

Sie sind mit einem Netzwerk verbunden, das aus n Servern besteht, die von 1 bis n nummeriert sind. Sie können jeden Server i in Zeit $\mathcal{O}(1)$ kontaktieren und erhalten als Antwort entweder eine '0' oder eine '1'. Leider sind einige der Server kaputt uns Sie sollen herausfinden welche Server betroffen sind.

- Falls Server i kaputt ist, sendet er bei jeder Anfrage ein unabhängig gleichverteiltes Bit.
- Falls Server i intakt ist, dann antwortet er auf jede Anfrage mit dem gleichen Bit $a_i \in \{0,1\}$.

  Allerdings ist der Wert a_i unbekannt und kann von Server zu Server variieren.

(a) Seien $\delta > 0$ und $i \in [n]$ gegeben. Beschreiben Sie einen Monte-Carlo Algorithmus, der herausfindet, ob Server i kaputt ist. Berechnen Sie die Fehlerwahrscheinlichkeiten (abhängig davon ob der Server kaputt/intakt ist) Ihres Algorithmus und stellen Sie sicher, dass Ihr Algorithmus Fehlerwahrscheinlichkeit höchstens $\frac{\delta}{n}$ hat.

(b) Sei $\delta > 0$ gegeben. Beschreiben Sie einen Monte-Carlo Algorithmus, der eine Liste aller kaputten Server erstellt und Fehlerwahrscheinlichkeit höchstens δ hat. Hierbei sagen wir, dass der Algorithmus erfolgreich ist, wenn die Liste alle kaputten Server und keinen intakten Server enthält.

a) k Anfragen, wenn alle Antworten gleich Ausgabe "Server i ist intakt"

X_k = alle k Anfragen bekommen gleiche Antwort

$\Pr[X_k \mid \text{Server i ist kaputt}] = (\frac{1}{2})^{k-1} \overset{!}{\leq} \frac{\delta}{n}$

$\Leftrightarrow k \geq -\log_2(\frac{\delta}{n}) + 1$

b) Nutzen von Teilaufgabe a) für jeden Server
$Y_i = 1$ falls Server i kaputt ist und wir ihn als intakt ausgeben

Liste inkorrekt: $Y = \sum_{i=1}^{n} Y_i \geq 1$

$\Pr[Y \geq 1] \overset{\text{Markov}}{\leq} \frac{E[Y]}{1} \leq \delta$

$E[Y] = \sum_{i=1}^{n} E[Y_i] \leq n \cdot \frac{\delta}{n} = \delta$

(Linearität des Erwartungswerts)

TARGET SHOOTING

Gegeben: endliche Mengen S, U so dass $S \subseteq U$, $I_S : U \to \{0,1\}$ $I_S(u) = 1 \iff u \in S$

Gesucht: $\frac{|S|}{|U|}$

TARGET-SHOOTING

1: Wähle $u_1, \ldots, u_N \in U$ zufällig, gleichverteilt und unabhängig

2: **return** $N^{-1} \cdot \sum_{i=1}^{N} I_S(u_i)$

Sei $\varepsilon > 0$ beliebig klein. Wie groß muss N sein, damit der Algorithmus mit Wahrscheinlichkeit $\geq 1-\delta$ eine Antwort im Intervall $\left[(1-\varepsilon)\frac{|S|}{|U|}, (1+\varepsilon)\frac{|S|}{|U|}\right]$ ausgibt?

Satz 2.79. Seien $\delta, \varepsilon > 0$. Falls $N \geq 3\frac{|U|}{|S|} \cdot \varepsilon^{-2} \cdot \ln(2/\delta)$, so ist die Ausgabe des Algorithmus TARGET-SHOOTING mit Wahrscheinlichkeit mindestens $1 - \delta$ im Intervall $\left[(1-\varepsilon)\frac{|S|}{|U|}, (1+\varepsilon)\frac{|S|}{|U|}\right]$.

PRIMZAHLTEST

übliches Finden von Primzahlen: Aussuchen einer random Zahl n gegebener Länge, dann testen, ob sie eine Primzahl ist

naive Option: alle Teiler bis $\sqrt{n}$ testen → ineffizient

random Zahl $a \in \{2, \dots, \sqrt{n}\}$ auswählen und schauen ob $\gcd(a,n) > 1$

Kleiner fermatscher Satz: Ist $n \in \mathbb{N}$ prim, so gilt für alle Zahlen $0 < a < n$ $a^{n-1} \equiv_n 1$.

Carmichael-Zahl n: für alle teilerfremden Zahlen a gilt $a^{n-1} \equiv_n 1$
kleinste Carmichael-Zahl: $561 = 3 \cdot 11 \cdot 17$

Miller-Rabin-Primzahltest

Idee: wenn n prim, dann ist $(\mathbb{Z}_n, +, *)$ ein Körper
→ $x^2 \equiv_n 1$ hat Lösungen $x = 1$ und $x = -1 \equiv_n n-1$

$n - 1 = d \cdot 2^k$, d ungerade

n prim $\Leftrightarrow \forall a \in \{1, \dots, n-1\}$ gilt $a^{n-1} = (a^d)^{2^k} \equiv_n 1$

MILLER-RABIN-PRIMZAHLTEST(n)

```
1: if n = 2 then
2:     return 'Primzahl'
3: else if n gerade oder n = 1 then
4:     return 'keine Primzahl'
5: Wähle a ∈ {2, 3, ..., n − 1} zufällig und
6: berechne k, d ∈ ℤ mit n − 1 = d2^k und d ungerade.
7: x ← a^d (mod n)
8: if x = 1 or x = n − 1 then
9:     return 'Primzahl'
10: repeat k − 1 mal
11:     x ← x^2 (mod n)
12:     if x = 1 then
13:         return 'keine Primzahl'
14:     if x = n − 1 then
15:         return 'Primzahl'
16: return 'keine Primzahl'
```

Laufzeit $O(\text{ld}^2(n))$

$\Pr[\text{"nicht prim"} \mid \text{nicht prim}] \geq \frac{3}{4}$