Karlsruhe Institute of Technology

Stefan Walzer, Stefan Hermann
ITI Prof. Sanders
winter term 2025/2026

Exercise Sheet 3 – Important Random Variables and How to Sample Them

Probability and Computing

Exercise 1 – Ber(1/3) from Ber(1/2)

Design an algorithm that, given a sequence $B_1, B_2, \ldots \sim \mathrm{Ber}(1/2)$ of random bits, computes a sample $B \sim \mathrm{Ber}(1/3)$ in expected time $\mathcal{O}(1)$.

Exercise 2 – Ber(p) and $\mathcal{U}(\{1,\ldots,n\})$ from $\mathcal{U}([0,1])$

We now assume a machine model that can handle real numbers and allows us to sample $U \sim \mathcal{U}([0,1])$. Show that we can also sample $B \sim \mathrm{Ber}(p)$ for $p \in [0,1]$ and $X \sim \mathcal{U}(\{1,\ldots,n\})$ for $n \in \mathbb{N}$.

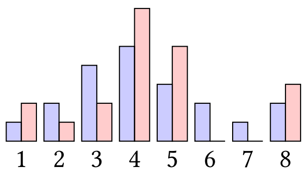
Hint: For the rest of this sheet and the course, we take this result as given.

Exercise 3 – Rejection Sampling in General

Let $\mathcal{D}_1$ and $\mathcal{D}_2$ be distributions over a finite set D. Assume:

1. We can sample $X \sim \mathcal{D}_1$ in time $\mathcal{O}(1)$.
2. For any $x \in D$, $p_1(x) := \Pr_{X\sim\mathcal{D}_2}[X = x]$ as well as $p_2(x) := \Pr_{X\sim\mathcal{D}_1}[X = x]$ can be computed in $\mathcal{O}(1)$.
3. There exists $C > 0$ such that for all $x \in D$,

$$p_2(x) \leq C \cdot p_1(x).$$



Possible histogram for $\mathcal{D}_1$ (blue, left) and $\mathcal{D}_2$ (red, right). It always holds that "red $\leq 2 \cdot$ blue", so condition (3) holds with $C = 2$.

Design an algorithm that samples $Y \sim \mathcal{D}_2$ in expected time $\mathcal{O}(C)$.

Exercise 4 – $G \sim \mathrm{Geom}_1(p)$ with Inverse Transform Sampling

Design an algorithm that, for a given $p \in (0,1]$, samples a random variable $G \sim \mathrm{Geom}_1(p)$ in time $\mathcal{O}(1)$.

Exercise 5 – Sampling without Replacement

We consider algorithms that, for $k, n \in \mathbb{N}$ with $0 \leq k \leq n/2$, compute a set $S \subseteq [n]$ of size k, chosen uniformly at random among all subsets of $[n]$ of size k.

(a) Why can we assume $k \leq n/2$ without loss of generality?

(b) Describe an algorithm that has an expected runtime of $O(k \log k)$.
Hint: Rejection sampling and search tree.

(c) **Bonus:** Design an algorithm that has a worst-case runtime of $O(k \log k)$.

(d) **Bonus:** Research how to achieve a worst-case runtime of $O(k)$:

`https://stackoverflow.com/a/67850443`