MAX PLANCK INSTITUTE FOR INFORMATICS

SIC Saarland Informatics Campus

Lecturers: **Evangelos Kipouridis and Tomasz Kociumaka** **Winter 2025/26**
Tutors: **Anouk Duyster and Yonggang Jiang**

Randomized and Approximation Algorithms, Exercise Sheet 3

https://www.mpi-inf.mpg.de/departments/algorithms-complexity/teaching/winter-2025/26/randomized-and-approximation-algorithms

Total Points: **100** + 80 bonus points Due: **14:00** on Thursday, **November 13**, 2025

Exercise 1 — 15 points

Consider a Las-Vegas algorithm $\mathcal{A}$ that runs in expected time t and correctly evaluates some function f of its input x. Formally, this means that:

- the running time $T(x, r)$ (as a function of the input x and randomness r) satisfies $\mathbb{E}_r[T(x, r)] = t(x)$ for every input x, and
- the output $\mathcal{A}(x, r)$ (as a function of the input x and randomness r) satisfies $\mathcal{A}(x, r) = f(x)$ for every input x and randomness r.

You have seen how to derive an algorithm $\mathcal{A}'$ that, for any $\delta \in (0, \frac{1}{2}]$, runs in time $\mathcal{O}(t \cdot \delta^{-1})$ and declares a failure (outputs a value $\perp$ outside the image of f) with probability at most δ. Formally:

- the running time $T'(x, r, \delta)$ satisfies $T'(x, r, \delta) = \mathcal{O}(t(x) \cdot \delta^{-1})$ for every x, r, and $\delta \in (0, \frac{1}{2}]$,
- the output $\mathcal{A}'(x, r, \delta)$ satisfies $\mathcal{A}'(x, r, \delta) \in \{f(x), \perp\}$ for every x, r, and $\delta \in (0, \frac{1}{2}]$, and
- the output $\mathcal{A}'(x, r, \delta)$ also satisfies $\mathbb{P}_r[\mathcal{A}'(x, r, \delta) = \perp] \leq \delta$ for every x and $\delta \in (0, \frac{1}{2}]$.

Show how to improve the running time to $\mathcal{O}(t(x) \cdot \log \delta^{-1})$ without violating the other constraints. The formal statements are provided for reference; your analysis can operate on the informal statements.
Hint: Use the slower version of $\mathcal{A}'$ in a black-box fashion.

Exercise 2 — 10 + 15 points

Random variables $X_1, \ldots, X_n$ are called *pairwise independent* if every two of them are independent.

a Prove that $\mathrm{Var}[X_1 + \cdots + X_n] = \mathrm{Var}[X_1] + \cdots + \mathrm{Var}[X_n]$ still holds for pairwise independent variables.

b For a prime number p and integers $\alpha, \beta \in \mathbb{Z}_p$ sampled independently and uniformly at random, define a function $f(x) = (\alpha x + \beta) \bmod p$. Prove that the variables $f(0), \ldots, f(p-1)$ are uniformly random and pairwise independent elements of $\mathbb{Z}_p$.

This exercise provides tools that will be helpful in the next exercises.

Exercise 3 — 20 + 10 points

Consider a uniformly random subset $S \subseteq V$ of vertices in an *unweighted*, *undirected*, simple graph $G = (V, E)$. Let $C := |E[S, V \setminus S]|$ be a random variable representing the size of the cut-set defined by G (the number of edges between S and its complement).

a Prove that $\mathbb{P}[|C| < 0.99 \cdot \mathbb{E}[C]] = \mathcal{O}\left(\frac{1}{|E|}\right)$. *Hint:* Use pairwise independence.

b Disprove the analogous claim for directed graphs (where C counts directed edges from S to $V \setminus S$).
Hint: Construct directed graphs with arbitrarily many edges for which $\mathbb{P}[|C| < 0.99 \cdot \mathbb{E}[C]] = \Omega(1)$.

Exercise 4 — 20 + 10 points + 40 bonus points

Consider two finite sets U and W. For a *multiset* A consisting of elements of U and a function $f : U \to W$, let $f(A) = \{f(a) : a \in A\}$ denote a multiset obtained by mapping each element using f (note that $|A| = |f(A)|$, even though the two multisets may differ in the number of distinct elements they contain). Moreover, let $|S|_s$ denote the multiplicity of s in a multiset S (the number of copies of s contained in S).

a Suppose that $f : U \to W$ is chosen uniformly at random, that is, for all $u \in U$, the values $f(u)$ are drawn from W independently and uniformly at random. Prove that the following holds for every multiset A, element $u \in U$, and $c \in \mathbb{R}_+$:

$$\mathbb{P}\left[|A|_u \le |f(A)|_{f(u)} < |A|_u + \frac{c|A|}{|W|}\right] \ge 1 - \frac{1}{c}.$$

b Now suppose that the values $f(u)$ are *pairwise* independent and satisfy $\mathbb{P}[f(u) = w] \le \frac{2}{|W|}$ for every $u \in U$ and $w \in W$. Prove that the following holds for every multiset A, element $u \in U$, and $c \in \mathbb{R}_+$:

$$\mathbb{P}\left[|A|_u \le |f(A)|_{f(u)} < |A|_u + \frac{c|A|}{|W|}\right] \ge 1 - \frac{2}{c}.$$

c (*) Suppose that the elements of A arrive in a stream in arbitrary order (for example, A may represent the source IP addresses of all packets passing through a router; the router does not have enough memory to store the entire A). Design an algorithm that, for any given parameters $\epsilon, \delta \in (0, \frac{1}{2})$:

- uses $\mathcal{O}(\epsilon^{-1} \cdot \log \delta^{-1} \cdot \log(|A| + |U|))$ bits of space, and
- when queried with any $u \in U$, can estimate $|A|_u$ up to an additive error of $\epsilon|A|$, correctly with probability at least $1 - \delta$.

Hint: First, use Exercise 2b to design a function f so that both f and $f(A)$ can be stored in $\mathcal{O}(\epsilon^{-1}\delta^{-1} \cdot \log(|A| + |U|))$ bits. Then, improve the dependence on δ^{-1}.

Exercise 5 — 15 + 5 + 5 + 15 bonus points

Recall the Floyd–Rivest median selection algorithm from the lecture (consult the slides for reference). Assume that the input array is $A[1..n]$ and that the s samples in the first step are selected as $A[X_1], \ldots, A[X_s]$, where $X_1, \ldots, X_s \in [1..n]$ are sampled independently and uniformly at random. We showed that the algorithm makes $\frac{3}{2}n + \mathcal{O}(n^{3/4} \log n)$ comparisons, takes $\mathcal{O}(n)$ time, and fails with probability $\mathcal{O}(n^{-1/4})$.

a (*) Prove that the above guarantees remain valid when $X_1, \ldots, X_s$ are only pairwise independent.
Hint: Explain what changes in the analysis and rework only the affected parts.

b (*) Explain how to implement the Floyd–Rivest algorithm so that it achieves the above guarantees when n is prime, even if the only source of randomness is a uniformly random integer between 1 and n^2.

c (*) Explain how to implement the Floyd–Rivest algorithm so that it achieves the above guarantees when n is prime, even if the only source of randomness is a sequence of $\mathcal{O}(\log n)$ uniformly random bits.
Hint: Reduce to the previous case at the cost of increasing the failure probability by $\mathcal{O}(n^{-1})$.

d (*) Explain how to implement the Floyd–Rivest algorithm so that it achieves the above guarantees for arbitrary n, even if the only source of randomness is a sequence of $\mathcal{O}(\log n)$ uniformly random bits.