Institute of Theoretical Computer Science
Prof. Karl Bringmann, Prof. Rasmus Kyng, Prof. Vera Traub
Manuel Christalla, Christoph Grunau, Paul Paschmanns

Algorithms and Probability **Session Sheet S9** **FS 2026**

For the exercise sessions on 23 April 2026.

Exercise S9.1 – *Boosting the success probability of Monte Carlo Algorithms*

Alice, Bob and Claire have learned that Monte Carlo algorithms can often be improved by running them multiple times. They are each given a blackbox [1] Monte Carlo algorithm $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{C}$ with a certain bound on the error. The task is to construct new algorithms $\mathcal{A}$*_improved*, $\mathcal{B}$*_improved*, and $\mathcal{C}$*_improved* resp. (using $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$ resp. as subroutine) with a success probability of at least 99%. Try to keep the number of calls to the subroutines small.

(a) Given a graph (V, E), Alice tries to find a vertex set $\emptyset \neq S \subsetneq V$ that minimizes $|\delta(S)|$. She is given a Monte Carlo Algorithm $\mathcal{A}$ that, given a graph, returns some vertex set $\emptyset \neq S \subsetneq V$. With probability at least $p_{\mathcal{A}} \geq 1/\binom{n}{2}$ this set minimizes $|\delta(S)|$. Use this algorithm $\mathcal{A}$, to construct another algorithm $\mathcal{A}$*_improved* for the same problem, that has a success probability of at least 99%. *Hint:* $1 + x \leq e^x$ *for* $x \in \mathbb{R}$.

(b) Bob wants to check if a number is prime. He already knows a Monte Carlo algorithm $\mathcal{B}$ which takes as input a natural number N. If N is prime, the algorithm always returns 'prime'. If N is not prime, the algorithm returns 'not a prime' with probability $p_B \geq 1/2$. Use Bobs algorithm $\mathcal{B}$, to construct another algorithm $\mathcal{A}$*_improved* for the same problem, that has a success probability of at least 99%.

(c) Claire chose the most difficult problem. She wants to determine if a given graph has an Hamiltonian cycle. She thought of an algorithm $\mathcal{C}$ that, given a graph, outputs 'YES' or 'NO'. If a graph G has (does not have) a Hamiltonian cycle, the algorithms $\mathcal{C}(G)$ returns 'YES' ('No') with probability $p_{\mathcal{C}} \geq 3/4$. Help Claire to find an algorithm C*_improved* that is correct with probability at least 99%.

(d) Unfortunately, you forgot what Clair's initial algorithm $\mathcal{C}$ was. Thus, you have to find an initial algorithm yourself. Describe a fast Monte Carlo algorithm that, given a graph, outputs the correct answer 'YES' or 'NO' with probability 1/2. Can you boost this algorithm to a success probability of 99%?

(e) **Difficult**: Assume you are given the Monte Carlo algorithm $\mathcal{C}$ from part (c). Can you construct an algorithm that not only determines whether there is a Hamiltonian cycle, but even computes one if it exists. Of course, again only with a success probability of 99%.

Solution S9.1 – *Boosting the success probability of Monte Carlo Algorithms*

(a) Let k be a value (natural number) that we fix later. The algorithm $\mathcal{A}$*_improved* we propose does the following. We call algorithm $\mathcal{A}$ k times (such that the calls are independent of each

[1] meaning they can use the algorithm, but do not know how it works internally

other). This gives us k (not necessarily different) vertex sets S_1 to S_k. Among them, we return the one that minimizes $|\delta(S_i)|$.

The algorithm $\mathcal{A}$*_improved* returns the correct result if at least one of the k calls to $\mathcal{A}$ returned the correct result. Hence,

$$\begin{aligned} \Pr[\mathcal{A} \text{ returns a correct set}] &= 1 - \Pr[\text{a single call to } \mathcal{A} \text{ returns a wrong set}]^k \\ &\geq 1 - (1 - \frac{2}{n(n-1)})^k \\ &\geq 1 - (e^{-\frac{2}{n(n-1)}})^k . \end{aligned}$$

Choosing $k = \frac{n(n-1)}{2} \cdot \ln 100$, we get $\Pr[\mathcal{A} \text{ returns a correct set}] \geq 1 - e^{-\ln 100} = 0.99$.

(One could also apply 2.72 instead.)

(b) We proceed similar to (a). Our algorithm $\mathcal{B}$*_improved* calls $\mathcal{B}$ k times. If at least one of these calls return 'not a prime', we also return 'not a prime'. Otherwise, we return 'prime'. If our input N is a prime, each call to $\mathcal{B}$ and hence, also $\mathcal{B}$*_improved* will correctly return 'prime'. If N is not prime, each call to $\mathcal{B}$ has a probability of $1/2$ to correctly return 'not a prime'. Thus, our algorithm $\mathcal{B}$*_improved* only fails, if all k calls to $\mathcal{B}$ wrongly return 'prime'. This happens with probability

$$\Pr[\text{wrongly return 'prime'}] = (1/2)^k .$$

For $k = 7 > \log_2 100$ this probability is less than 0.01. In other words, for $k = 7$ the algorithm $\mathcal{B}$*_improved* correctly detects non-prime numbers with probability at least 0.99.

(One could also apply 2.74)

(c) This task is a bit more difficult. In contrast to (b) there is no answer $\mathcal{C}$ could give, for which we know it is true. Instead, we will return the answer that occurs more often: The algorithm $\mathcal{C}$*_improved* calls $\mathcal{C}$ k times. If at least $k/2$ of these calls return 'YES', we also return 'YES'. Otherwise, we return 'NO'.

Let X be the random variable counting the number of calls to $\mathcal{C}$ that return the correct result. Then $\mathbb{E}[X] \geq \frac{3}{4}k$. The algorithm $\mathcal{C}$*_improved* only returns the wrong answer, if $X \leq \frac{1}{2}k$. In particular, X would have to deviate from its expected value by a factor of $\frac{1}{2}/\frac{3}{4} = 1 - \frac{1}{3}$. Because X is the sum of independent Bernoulli variables, we can apply the Chernoff bound:

$$\Pr[X \leq (1 - 1/3)\,\mathbb{E}[X]] \leq e^{-\frac{1}{2}\frac{1}{3^2}\,\mathbb{E}[X]} \leq e^{-\frac{1}{24}k} .$$

Thus, we can choose $k = 111 > 24 \cdot \ln 100$.

(One could also apply 2.75)

(d) A possible such algorithm is the following: Ignore the input, flip a coin, return 'YES' if the coin shows "tails" and 'NO' otherwise. Let this algorithm be called $\mathcal{D}$. Any other algorithm that only relies on calling $\mathcal{D}$ will return 'YES' resp. 'NO' with a probability that is independent of the input graph. Hence, it cannot be boosted to a success probability above $\frac{1}{2}$.

(e) We only sketch a possible algorithm. Assume first, that the algorithms $\mathcal{C}$ always answers correctly. First decide, if G has a Hamiltonian cycle. If yes, process the edges one after another. If the current graph without edge e still contains a hamiltonian cycle, we remove

e. If not, we keep it. After processing all edges, this leaves us with a Hamiltonian cycle. If $\mathcal{C}$ only return the correct answer with a certain probability, we again have to be careful to call $\mathcal{C}$ sufficiently often.