Karlsruhe Institute of Technology

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

Exercise Sheet 8 – Bounded Differences and Bloom Filters

Probability and Computing

Exercise 1 – Balls, Bins and Bounded Differences

Let $\lambda > 0$ be a constant, $m = \lambda n$ the number of balls, and n the number of bins. The j-th ball is placed in bin X_j, where $X_1, \dots, X_m \sim \mathcal{U}([n])$ are independent random variables. The collision count is defined as $C = |\{(i,j) \mid 1 \leq i < j \leq m, X_i = X_j\}|$. The goal of this exercise is to derive a concentration bound for C.

(i) Show that $\mathbb{E}[C] = \Theta(n)$.

For $i \in [n]$, let $L_i = |\{j \in [m] \mid X_j = i\}|$ denote the load of bin i. It is neither difficult nor interesting to show that $\Pr[\max_{i\in[n]} L_i \leq \log n] \geq 1 - \mathcal{O}(n^{-100})$. Assume this to hold in the following (proof provided in the sample solution).

(ii) Define $C_{\text{cap}} := \sum_{i\in[n]} \binom{\min(\log n, L_i)}{2}$. Show that $\Pr[C_{\text{cap}} = C] = 1 - \mathcal{O}(n^{-100})$.

(iii) Show that $\Pr[C_{\text{cap}} - \mathbb{E}[C_{\text{cap}}] \geq t] \leq \exp(-2t^2/(m \cdot \log^2 n))$.

(iv) Show that $\Pr[C - \mathbb{E}[C] \geq n^{2/3}] = \mathcal{O}(n^{-100})$.

(v) Assume an algorithm inserts n keys into a hash table and then queries every key exactly once. Show that the algorithm runs in time $\mathcal{O}(n)$ except with probability $\mathcal{O}(n^{-100})$ under suitable assumptions. Note that we are *not* talking about *expected* running time.

The following exercise is not directly related to randomized algorithms, except that we used the bound in lecture. Hence, "Bonus".

Exercise 2 – Bonus: Approximations of *e*

You know that $e = \lim_{n\to\infty}(1 + \frac{1}{n})^n$ (this is even a common *definition* of e). Derive from this that the following two inequalities hold for all $n \in \mathbb{N}$:

$$(1 + \tfrac{1}{n})^n \leq e \leq (1 + \tfrac{1}{n})^{n+1}$$
$$(1 - \tfrac{1}{n})^n \leq e^{-1} \leq (1 - \tfrac{1}{n})^{n-1}.$$

The following steps are suggested:

(i) Prove the left-hand inequalities.
Hint: Use again $1 + x \leq e^x$.

(ii) Show that the right-hand sides are monotonically decreasing in n.

(iii) Deduce the right-hand inequalities from (ii) and a limit argument.

Exercise 3 – Counting Bloom Filter (a.k.a.: Count-Min Sketch)

A Counting Bloom Filter is initially like a standard Bloom Filter: it manages n keys in an array of size m, with load factor $\alpha := \frac{n}{m}$ and k hash functions. The parameters are chosen as in lecture so that a standard Bloom Filter would have false-positive probability ε. The array now contains natural numbers instead of bits. In addition to insert and query, a delete operation is now available.

```
Algorithm insert($x$):
    for $i \in [k]$ do
        $A[h_i(x)] + +$
```

```
Algorithm delete($x$):
    for $i \in [k]$ do
        $A[h_i(x)] - -$
```

```
Algorithm query($x$):
    for $i \in [k]$ do
        if $A[h_i(x)] = 0$ then
            return false
    return true
```

We allow a key to be inserted *multiple times* into the Counting Bloom Filter. Hence, the managed object S is now a *multiset* with n elements $\{x_1, \ldots, x_n\}$, each with multiplicity $a_1, \ldots, a_n$. The operation insert(x) adds one copy of x to the multiset S, i.e., increments the multiplicity of x if x is already in S, or adds a new element with multiplicity 1 if x is not yet in S. The meaning of delete is analogous. As before, query may return false positives but not false negatives.

(a) We require that delete is called only for elements that are actually in S (with multiplicity at least 1). What breaks if we do not enforce this?

Additional useful operations can be implemented with Counting Bloom Filters.

(b) Implement an operation count that, for $x \in D$, returns an estimate count(x) of the multiplicity a of x in S. Show that $\Pr[a \neq \text{count}(x)] \leq \varepsilon$.

(c) The primary argument for using (Counting) Bloom Filters is their low memory footprint compared to exact data structures. We should therefore avoid using large integer types (e.g., 64 bits) for counters. Consider an application where "most" counters never exceed 8 bits. We therefore use an array A of 8-bit counters. Briefly discuss the following proposals for handling counter overflows. What are the advantages and what compromises are made?

- Alice merely prevents counter overflows, i.e., adapts the $A[h_i(x)] + +$ operation in insert and the $A[h_i(x)] - -$ operation in delete so that the counter is incremented/decremented only if the maximum/minimum representable value has not yet been reached.

- Bob proposes to "freeze" a counter that reaches $(11111111)_2 = 255$, i.e., no future insert or delete operations will modify this counter.
- Carol proposes to use the bit string $(11111111)_2 = 255$ as a marker indicating that the true counter value exceeds 254. In this special case, the true counter value is stored in a hash table.

Exercise 4 – Estimating Set Intersections with Bloom Filters

Let $n, m, k \in \mathbb{N}$. Alice and Bob wish to estimate how similar their musical tastes are. Let X be Alice's n favorite songs and Y be Bob's n favorite songs. The goal is to estimate $\gamma := \frac{|X \cap Y|}{n} \in [0, 1]$. They proceed as follows:

- Alice constructs a Bloom filter $A[1..m] \in \{0, 1\}^m$ for X using k hash functions $h_1, \ldots, h_k$.
- Bob constructs a Bloom filter $B[1..m] \in \{0, 1\}^m$ for Y using the *same* k hash functions.
- Alice and Bob exchange their filters and compute $\delta := \frac{|\{i \in [m] \mid A[i] \neq B[i]\}|}{m}$.
- Alice and Bob compute an estimate $\bar{\gamma}$ for γ based on δ.

Solve the following subproblems:

(a) Discuss: What advantages and disadvantages might this method have compared to direct exchange of X and Y?

(b) Gain intuition: What values of δ do you expect (approximately) for the extreme cases $\gamma = 1$ and $\gamma = 0$?
Hint: You may assume here and in the following that the Bloom filters use an "optimal" configuration with $\alpha k = \ln(2)$.

(c) Compute $\mathbb{E}[\delta]$ as a function of γ. You may drop lower-order terms, e.g., write $(1 - \frac{1}{m})^m \approx e^{-1}$ without carrying an $o(1)$ term.
Hint: At first glance, other parameters (e.g., $n, m, k, \alpha, \varepsilon$) might appear relevant. Their influence vanishes in lower-order terms.

(d) Discuss: Which concentration bound is suitable for proving that δ is close to $\mathbb{E}[\delta]$ with high probability?

(e) Rearrange the equation from (c) to show how an estimate $\bar{\gamma}$ for γ can be computed from δ.

(f) Speculate: What role does the choice of k (or ε) play in this context?