Universitat Politècnica de Catalunya
Facultat d'Informàtica de Barcelona

**Degree:** Grau en Enginyeria Informàtica **Academic year:** 2024–2025 (Mid-term Exam)
**Course:** Randomized Algorithms (RA-MIRI) **Date:** October 31, 2024
**Time:** 2h

1. **(2.5 points)** Back in 1951, John von Neumann proposed a very clever way to obtain fair coin tosses (probability of heads and of tails equal to $1/2$ both), out of some biased coin. Imagine that you have a coin that shows heads with probability $p \neq 1/2$. Neumann's scheme is:

   (a) Flip the coin twice (independent flips).

   (b) If both flips are the same, e.g., HH, discard and repeat.

   (c) Otherwise, the "output" is the result of the first flip (ignoring the second).

   Questions:

   (a) Prove that the scheme will output heads and tails with identical probability, no matter the value of $p$.

   (b) Compute the average number of rounds needed to produce an output in the scheme above: recall that each time the two flips land the same, we have to make a new round. Give your answer in terms of $p$ and/or $q$.

---

2. **(2.5 points)** In this exercise, we consider mappings (functions) from $\{1, \ldots, n\}$ to $\{1, \ldots, n\}$. A *fixed point* of a mapping $f$ is any value $i$, $1 \leq i \leq n$, such that $f(i) = i$.

   Questions:

   (a) Compute the number $F_n$ of mappings of $\{1, \ldots, n\}$ to $\{1, \ldots, n\}$,

   (b) Compute the probability that a random mapping has no fixed point. Does this probability converge to a limit as $n \to \infty$? If so, what is that limit?

   We now consider the algorithm below, based on the rejection method. It will produce a random pair of mappings $\langle f_1, f_2 \rangle$, such that neither $f_1$ nor $f_2$ have fixed points; moreover, $f_1(i) \neq f_2(i)$ for all $i$, $1 \leq i \leq n$. Such a pair is called *compatible*.

```
// Mapping = array[1..n] of integer
procedure GenCompatibleMappings(n : integer) : pair<Mapping,Mapping>
    while true do
      Mapping f1 := RandomMapping(n); // Time: Theta(n)
      Mapping f2 := RandomMapping(n); // Time: Theta(n)
      if not HasFixedPoint(f1) and not HasFixedPoint(f2) and
         allImagesDistinct(f1, f2) then
         // allImagesDistinct(f1, f2)==true if and only if
         // f1(i) != f2(i) for all i; Time: Theta(n)
         return {f1, f2};
      endif
    endwhile
endprocedure
```

Let $R_n$ be the number of iterations of `GenCompatibleMappings` until a *compatible* pair of mappings is found and returned. Since the cost of the algorithm is $\Theta(n \cdot R_n)$, we shall focus on $R_n$.

(d) What is the probability that we produce a compatible pair in a given round?

(e) Compute $\mathbb{E}[R_n]$.

(f) Suppose that we design a variant of the previous algorithm which has two loops. The first loop keeps generating random mappings until a mapping $f_1$ without fixed points is produced. The second loop generates random mappings until we find $f_2$ without fixed points and such that `allImagesDistinct(f1,f2)` is true. Like in the previous case, we are interested in the sum $R'_n = R_n^{(1)} + R_n^{(2)}$, where $R_n^{(1)}$ and $R_n(2)$ are the number of rounds of the first and second loops, respectively. Which algorithm would you prefer? Which one is smaller: $\mathbb{E}[R_n]$ or $\mathbb{E}[R'_n]$?

---

3. **(2.5 points)** A server processes incoming requests by adding them to any of its $M$ buffers (= queues). Each request is enqueued in a randomly chosen buffer, so that if at any given time there are $N$ pending requests, then we expect each buffer to hold $N/M$ requests on average. Try to give the best possible bounds you can.

   (a) Compute an upper bound for the probability that one buffer holds $\geq (1+\delta)N/M$ requests for some $\delta > 0$ when $N = cM \ln M$ for some $c > 0$. What is the value of $\delta$ (as a function of $c$ and $M$) such that the probability of one buffer having more of $(1+\delta)N/M = (1+\delta)c \ln M$ requests is $\leq 1/M^2$?

   (b) Now suppose $N = M \cdot \phi(M)$ for some function $\phi(M)$, and let $A > 1$ be a constant. How large must be $\phi(M)$ so that you can guarantee that $\mathbb{P}[\exists i : b_i > A \cdot \mathbb{E}[b_i]]$ tends to 0 as $M$ grows? Is your bound working for some $\phi(M)$ in $\Theta(1)$?

4. **(2.5 points)** Consider some hash table with $M$ slots, in which we insert $n$ items. We will say an item $x$ is *happy* if $x$ does not collide with any other item at the time of its insertion. Let $F_{n,M}$ be the number of happy items in the table after $n$ items have been inserted. Compute the expected number of happy items $\mathbb{E}[F_{n,M}]$ for the following two hash tables; in both cases, give your answer in terms of $M$ and $n$ (or $\alpha = n/M$).

   (a) When the hash table is implemented with separate chaining (N.B. an item is happy if the corresponding list of synonyms was empty when we insert the item).

   (b) When the hash table is implemented with random hashing. Recall that random hashing is an abstract/ideal model for open addressing hashing, in which we have an unbounded number of hash functions. To insert an item $x$, we try to do it in position $i_0 = h_0(x)$. If the position is occupied, then we try to insert $x$ at $i_1 = h_1(x)$, and so on, until a free position at $i_k = h_k(x)$, $k \geq 0$, is found.