Crypto Engineering
Discrete probability

2021-09-23

Exercise 1: (multi-)collisions

In all of this exercise we let $\mathcal{S}$ be an arbitrary finite set of size N, and we denote by $X \twoheadleftarrow \mathcal{S}$ the process of drawing X from $\mathcal{S}$ uniformly at random, and independently of any other process.

Let $X \twoheadleftarrow \mathcal{S}$, $Y \twoheadleftarrow \mathcal{S}$, $Z \twoheadleftarrow \mathcal{S}$.

1. Compute $\Pr[(X = x) \wedge (Y = y)]$ for any $x, y \in \mathcal{S}$.
2. Compute $\Pr[X = Y]$.
3. Compute $\Pr[X = Y = Z]$.

Exercise 2: (non-)uniform masks

Let X and Y be two independent random variables drawn from $\mathbb{F}_2$ with a uniform law for X and an unknown arbitrary law for Y.

1. What is the distribution of $X + Y$? (That is, compute $\Pr[X + Y = 0]$)

We now draw X and Y independently from a finite group $(\mathbb{G}, +)$ of size N.

2. What is (again) the distribution of $X + Y$? (Note that the distribution of $X + Y$ is given here by the discrete convolution of the distributions of X and Y).

Remark. The result shown in those two questions is essential in cryptography, and is used to justify the security of many constructions.

We go back to X and Y being drawn independently over $\mathbb{F}_2$, but consider this time arbitrary laws for both of them. We write c_X the *correlation bias* of X defined as $c_X = |2\Pr[X = 0] - 1|$, and the same for c_Y.

3. Compute c_{X+Y}, the correlation bias of $X + Y$.
4. By induction, give a formula for the correlation bias of the sum $X_1 + \cdots + X_N$ of N independent variables of correlation biases $c_1, \ldots, c_N$.

Remark. This last result is known in (symmetric) cryptography as the *piling-up lemma*.

Exercise 3: For my birthday I got a coupon for a pair of socks

Let again $\mathcal{S}$ be an arbitrary finite set of size N, which we sample repeatedly by drawing $X_1, \ldots, X_k$ uniformly and independently.

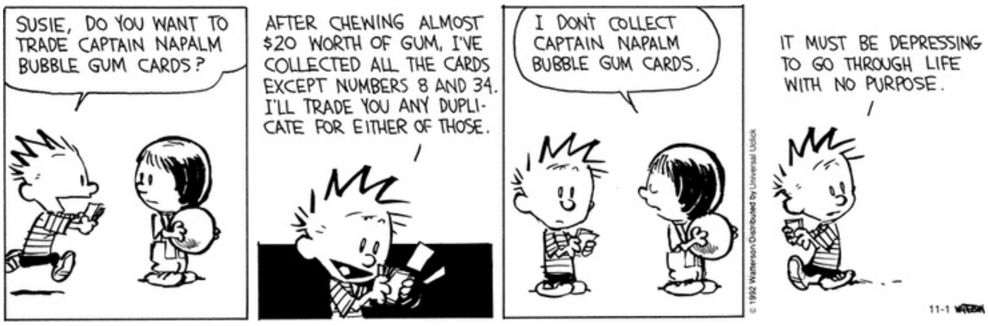

Figure 1: The coupon collector's problem: a Calvin & Hobbes illustration

Q.1 (*Pigeonhole principle*, or *lemme des chaussettes*): How many samples are necessary to ensure that $\exists\, i, j \neq i$ s.t. $X_i = X_j$ with probability 1?

Q.2 (*Birthday paradox*): How many samples are approximately needed to ensure that $\exists\, i, j \neq i$ s.t. $X_i = X_j$ with "high" probability (e.g. constant in function of N)?

Hint: You are not required to show this rigorously. You may also consider the probability that two lists L_1 and L_2 of elements of $\mathcal{S}$ contain a common one in function of their size, assuming independence of some well-chosen events.

Q.3 (*Coupon collector's problem, cf. Figure 1*): How many samples are approximately needed to ensure that $\forall \alpha \in \mathcal{S}, \exists\, i$ s.t. $X_i = \alpha$?

Hint: Consider the complementary event and use the approximation (for "large" x) $\left(1 - \frac{1}{x}\right)^x \approx e^{-1}$ and the union bound. Alternatively use the linearity of expectations and the fact that the expected number of drawings needed to pick a new coupon after k have been collected is $\left(\frac{n-k}{n}\right)^{-1}$.

Exercise 4: (close-to) uniform permutations $\star$

We consider the following algorithm to generate a random permutation of $[\![1, N]\!]$ (or more generally, of N arbitrary elements): 1) build a list of N pairs (r_i, i), where $r_i \twoheadleftarrow \mathbb{Z}/q\mathbb{Z}$; 2) sort the list according to the first element of the pairs; 3) return the list of the second element of the pairs in the sorted order.

Q.1 : Compute the number of sorted lists of N elements of $\mathbb{Z}/q\mathbb{Z}$.

Hint: Map all such possible lists to paths from $(0, 1)$ to (N, q) in the 2-dimensional discrete grid, where only horizontal and vertical steps are allowed.

Q.2 :

1. For every possible permutation generated by the algorithm, compute a non-trivial upper-bound for the number of drawings for $(r_1, \ldots, r_N)$ that lead to it.
2. What is then an upper-bound for the probability of occurence of any permutation?
3. Express this probability as $\delta/N!$ for δ of the form $\prod_{i=1}^{N-1}(1 + x_i/q)$.
4. For a fixed N, give an approximative criterion on q for δ to be close to 1 (for instance using the approximation (for "large" x) $\left(1 + \frac{1}{x}\right)^x \approx e$).

We now consider a variant of the algorithm, where one is interested in drawing a random combination of weight w. This is done as follows: 1) build a list of N pairs

$(r_i, i$ `<= w ? 1 : 0`$)$, where $r_i \twoheadleftarrow \mathbb{Z}/q\mathbb{Z}$; 2) sort the list according to the first element of the pairs; 3) return the list of the second element of the pairs in the sorted order.

Q.3 :

1. For every possible combination generated by the algorithm, compute a non-trivial upper-bound for the number of drawings for $(r_1, \ldots, r_N)$ that lead to it.
2. What is then an upper-bound for the probability of occurence of any combination?
3. Express this probability as $\delta/\binom{N}{w}$ for δ of the form $\prod_{i=1}^{N-1}(1 + x_i/q)$.
4. How could this have been found directly by using the result of **Q.2**?

Remark. Generating (close-to) uniform permutations and combinations is an important step in code- and lattice-based cryptosystems. The quantity δ computed above corresponds to the *divergence* between the uniform distribution and the one obtained with the above algorithm. This exercise is based on: `https://ntruprime.cr.yp.to/divergence-20180430.pdf`.