Homework 2

Out: 27 Jan. Due: 3 Feb.

Submit your solutions in pdf format on Gradescope by **5pm on Friday, February 3**. *Solutions may be written either in LaTeX (with either machine-drawn or hand-drawn diagrams) or* **legibly** *by hand. (The LaTeX source for this homework is provided in case you want to use it as a template.) Please be sure to begin the solution for each problem on a new page, and to tag each of your solutions to the correct problem! Per course policy, no late solutions will be accepted. Take time to write* **clear** *and* **concise** *answers; confused and long-winded solutions may be penalized. You are encouraged to form small groups (two to four people) to work through the homework, but you* **must** *write up all your solutions on your own. Depending on grading resources, we reserve the right to grade a random subset of the problems and check off the rest; so you are advised to attempt all the problems.*

1. A monkey types on a 26-letter keyboard. At each keystroke, each of the 26 letters is equally likely to be hit. The monkey types 2^{20} letters. What is the expected number of times the sequence "ape" appears in this text? [HINT: Let X be the number of occurrences. Write X as the sum of indicator random variables and use linearity of expectation. This should be a very simple calculation!]

2. Suppose we toss a coin with Heads probability p until we observe the kth Heads. Let the random variable X denote the number of tosses.

(a) Show that the distribution of X is

$$\Pr[X = t] = \binom{t-1}{k-1} p^k (1-p)^{t-k}.$$

[NOTE: This is known as the *negative binomial* distribution.]

(b) What is the expectation of X? [HINT: Use linearity of expectation and the formula for the expectation of a geometric r.v. Again this should be a very simple calculation.]

3. Andrew and Betty have a fair coin. They want to use it to generate a random sequence of 1000 coin tosses containing exactly 500 heads and 500 tails. Each such sequence should be equally likely.

(a) Andrew suggests the following scheme: flip the coin 1000 times; if you get exactly 500 heads, output the sequence; otherwise, try again. How many tosses do you expect to have to make under this scheme? [NOTE: you may assume that $n = 1000$ is large enough that asymptotic results hold; so, for example, instead of computing large factorials you should use Stirling's approximation: $n! \sim (\frac{n}{\mathrm{e}})^n \sqrt{2\pi n}$.]

(b) Betty claims that the following scheme is much more efficient: flip the coin until you have either 500 heads or 500 tails (one of these must happen before 1000 tosses); output this sequence, padded at the end with tails or heads respectively to make the total length 1000. Obviously this scheme requires at most 1000 tosses. Is this a good scheme? Justify your answer with a precise calculation. (Vague reasoning will not receive much credit.)

(c) Suggest a simple scheme for solving this problem that is better than both Andrew's and Betty's. What is the expected number of tosses required by your scheme? [NOTE: There is a scheme with an expected number of tosses as low as 2000. However, you will get credit for any scheme that is correct and substantially better than part (a).]

[Turn over for problem 4!]

4. Generating random factored integers

In cryptographic applications we often need to generate a random integer $r \in \{1, \ldots, n\}$ *together with* the factorization of r. Note that the obvious method of just generating r uniformly at random and then factoring r is not useful because we do not know how to factor integers (even with the aid of randomization) in polynomial time[1]. Here is a mysterious and remarkably simple algorithm for this problem:

1. generate a sequence of integers $n \geq s_1 \geq s_2 \geq \cdots \geq s_\ell = 1$ by choosing $s_1 \in \{1, \ldots, n\}$ u.a.r. and $s_{i+1} \in \{1, \ldots, s_i\}$ u.a.r. until 1 is reached
2. let r be the product of the s_i that are prime
3. **if** $r \leq n$ **then** output r with probability r/n **else** fail

(a) In preparation for analyzing the algorithm, consider the following scheme for generating a random $r \in \{1, \ldots, n\}$ using coins $c_1, c_2, \ldots, c_n$, where $\Pr[c_i \text{ comes up Heads}] = 1/i$. Flip coins $c_n, c_{n-1}, c_{n-2}, \ldots$ in sequence until the first Heads appears; if this happens on coin c_i then output $r = i$. Show that this scheme generates $r \in \{1, \ldots, n\}$ u.a.r.

(b) Now suppose we represent the sequence (s_i) generated by the algorithm as a vector $(m_1, \ldots, m_n)$, where m_j is the number of times the number j occurs in the sequence. (For example, if $n = 10$ and the sequence is $s_1 = 8$, $s_2 = 5$, $s_3 = 5$, $s_4 = 1$ then the vector would be $(1, 0, 0, 0, 2, 0, 0, 1, 0, 0)$.) Show that the probability of generating the sequence (s_i) is given by

$$\prod_{j=2}^{n} \left(\frac{1}{j}\right)^{m_j} \left(1 - \frac{1}{j}\right).$$

[HINT: Imagine implementing the picking of each s_i using the method of part (a). Then the entire process can be thought of as tossing a sequence of coins as in part (a).]

(c) Deduce from part (b) that the algorithm outputs each $r \in \{1, \ldots, n\}$ with equal probability α_n/n, where $\alpha_n = \prod_p (1 - 1/p)$ and the product is over all primes $p \leq n$. [NOTE: Avoid handwaving arguments and heavy calculations! Think carefully about part (b).]

(d) A standard theorem from number theory says that $\alpha_n^{-1} \sim 1.8 \ln n$. Suppose we repeat the algorithm until it outputs some r. What is the expected number of trials needed?

(e) The running time of each trial of the algorithm is dominated by the time to test the sequence (s_i) for primality. Show that the expected number of primality tests performed in one trial is the harmonic number $H_n = 1 + 1/2 + \ldots + 1/n \sim \ln n + \Theta(1)$.

(f) Deduce from parts (d) and (e) that the expected number of primality tests performed before an output is obtained is $O(\log^2 n)$. [NOTE: You may use Wald's equation, which says that, if the X_i are independent, identically distributed (iid) random variables, and the random variable T is a *stopping time* for the X_i (i.e., the event $T = t$ depends only on the values of $X_1, \ldots, X_t$, and not on future values X_i for $i > t$), then $\mathrm{E}[\sum_{i=1}^{T} X_i] = \mathrm{E}[T]\mathrm{E}[X_1]$, assuming these expectations are both finite. We will prove Wald's equation later in the course.]

[1] Note that by "polynomial time" here we mean polynomial in $\log n$, which is the number of bits in the representation of n. In practice n may be as large as 2^{512} for 512-bit security, so being polynomial in n is not very useful!