CSE 598, Fall 2024, Homework #1

Instructor: Zilin Jiang

Question 2(b) Given $\varepsilon > 0$, pick a natural number n such that $1/2^n < \varepsilon$ and $2^n - 1$ is divisible by 3. This can be achieved by taking n to be a sufficiently large even number. The idea is to pick a number n from a set S of 2^n numbers uniformly at random. We split the set S into three sets of size 1, $(2^n - 1)/3$ and $2(2^n - 1)/3$. Depending on which set contains n, our program outputs FAILURE, 0 or 1 respectively. Therefore our subroutine goes as follows.

Algorithm 1: Simulating a biased coin

```
Output: r
n = 0;
for i ← 1 to n do
    b ← 0 or 1 equally likely;
    n ← 2n + b;
end
/* Now n is a random n-digit binary number.                */
if n = 0 then
    r ←FAILURE;
else
    if n%3 = 0 then
        r ← 0;
    else
        r ← 1;
    end
end
```

Question 3(c) Expressing the two operators in $\mathbb{Z}/2\mathbb{Z}$, we have $\text{NOT}(x) = 1 + x$ and $x_1 \text{XOR} x_2 = x_1 + x_2$. Thus a Boolean formula with variables $x_1, \ldots, x_n$ built with NOT and

XOR only can be rewritten as $c + \sum_{i \in I} x_i$ for some $c \in \{0,1\}$ and some $I \subseteq \{1,2,\ldots,n\}$. It is easy to see that such a Boolean function is either constant, or balanced (meaning, assumes 0 or 1 equally likely). However, one can easily construct infinitely many Boolean functions (or equivalently, truth tables) that are neither constant nor balanced.

Question 4(b) The algorithm C'' repeatedly runs C until it outputs 0 or 1, and then C'' returns the last output. The number of runs of C follows the geometric distribution with success probability $p = 10\%$. Therefore the average (or mean) of the number of runs is $1/p = 1/10\% = 10$.

Question 4(c) Let n be a natural number to be determined later. The algorithm C' repeatedly runs C for up to n times, and C' returns 0 as soon as C outputs 0; otherwise C' return 1. Since C has no false negatives, when $f(x) = 1$, C always outputs 1, and so C' always outputs 1, which implies that C' has no false negatives, either. When $f(x) = 0$, the probability that C' outputs 1 is equal to the probability that C outputs 1 for n runs, which is equal to $(90\%)^n$. To ensure that the failure probability of C' is below 2^{-500}, it suffices to take $n = \lceil -500/\log_2 90\% \rceil = 3290$.

Question 4(d) Let n be an odd number to be determined later. The algorithm C' repeatedly runs C for n times, and returns the majority of the outputs. Suppose $f(x) = 1$. Then the probability that $C'(x) = 0$ is equal to the probability that more than $n/2$ outputs of C are 0. Let X be the sum of all the n outputs of C. The average of X, denoted by μ, is $0.6n$. The Chernoff bound for Bernoulli random variable X says that $\Pr(X \leq (1-\delta)\mu) \leq e^{-\delta^2\mu/2}$. Thus $\Pr(X \leq 0.5n) \leq e^{-(1/6)^2(0.6n)/2} = e^{-n/120}$. Due to symmetry, when $f(x) = 0$, the probability that $C'(x) = 1$ can also be bounded by $e^{-n/120}$. To ensure that the failure probability of C' is below 2^-500, it suffices to take $n = 120 \times 500 \times \ln 2 = 41,589$.