CS 473 ✧ Spring 2017

Homework 5

Due Wednesday, March 8, 2017 at 8pm

---

1. ***Reservoir sampling*** is a method for choosing an item uniformly at random from an arbitrarily long stream of data.

```
GETONESAMPLE(stream S):
  ℓ ← 0
  while S is not done
      x ← next item in S
      ℓ ← ℓ + 1
      if RANDOM(ℓ) = 1
          sample ← x        (⋆)
  return sample
```

At the end of the algorithm, the variable ℓ stores the length of the input stream S; this number is *not* known to the algorithm in advance. If S is empty, the output of the algorithm is (correctly!) undefined.

In the following, consider an arbitrary non-empty input stream S, and let n denote the (unknown) length of S.

(a) Prove that the item returned by GETONESAMPLE(S) is chosen uniformly at random from S.

(b) What is the *exact* expected number of times that GETONESAMPLE(S) executes line ($\star$)?

(c) What is the *exact* expected value of ℓ when GETONESAMPLE(S) executes line ($\star$) for the *last* time?

(d) What is the *exact* expected value of ℓ when either GETONESAMPLE(S) executes line ($\star$) for the *second* time (or the algorithm ends, whichever happens first)?

(e) Describe and analyze an algorithm that returns a subset of k distinct items chosen uniformly at random from a data stream of length at least k. The integer k is given as part of the input to your algorithm. Prove that your algorithm is correct.

For example, if $k = 2$ and the stream contains the sequence ⟨♠, ♥, ♦, ♣⟩, the algorithm should return the subset {♦, ♠} with probability 1/6.

2. ***Tabulated hashing*** uses tables of random numbers to compute hash values. Suppose $|\mathscr{U}| = 2^w \times 2^w$ and $m = 2^\ell$, so the items being hashed are pairs of w-bit strings (or $2w$-bit strings broken in half) and hash values are ℓ-bit strings.

Let $A[0\,..\,2^w - 1]$ and $B[0\,..\,2^w - 1]$ be arrays of independent random ℓ-bit strings, and define the hash function $h_{A,B} \colon \mathscr{U} \to [m]$ by setting

$$h_{A,B}(x, y) := A[x] \oplus B[y]$$

where $\oplus$ denotes bit-wise exclusive-or. Let $\mathscr{H}$ denote the set of all possible functions $h_{A,B}$. Filling the arrays A and B with independent random bits is equivalent to choosing a hash function $h_{A,B} \in \mathscr{H}$ uniformly at random.

(a) Prove that $\mathscr{H}$ is 2-uniform.

(b) Prove that $\mathscr{H}$ is 3-uniform. *[Hint: Solve part (a) first.]*

(c) Prove that $\mathscr{H}$ is ***not*** 4-uniform.

Yes, "see part (b)" is worth full credit for part (a), but only if your solution to part (b) is correct.