Randomness and Computation 2018/19
Coursework 1 (formative)

Issue date: Thursday, 31st January, 2019

The deadline for this coursework is 4pm on Thursday, 14th February 2019 (Thursday of week 5). Please submit your solutions either *electronically via* `submit` *or by hand to the ITO in Forrest Hill.* **Remember to check the School's policy on late coursework.**

This coursework should be your own individual work. *You may discuss understanding of the questions with your classmates, but may not share solutions, or give strong hints. If you use any resources apart from the course slides/notes, you* must *cite these in detail (on a per-question basis.)*

This is a formative coursework for RC, and the mark you obtain will not be included in the calculation of your grade for this course. However, our feedback will help you prepare a better submission for your second coursework, which does contribute to your course grade.

1. Imagine that we have a fair coin and want to generate a stream of (uniform) random bits. Instead of using the natural process for generating (say) the N needed bits (flip the coin N times), we have decided we will attempt to "reuse" the randomness and generate the N bits using only $n = \lceil 2\sqrt{N} \rceil$ random flips.

   To do this, we first generate $n = \lceil 2\sqrt{N} \rceil$ fair random bits $Z_1, \ldots, Z_{\lceil 2\sqrt{N} \rceil}$ with the fair coin. Next, we consider the index set of pairs $P = \{\{i, j\} : 1 \leq i < j \leq n\}$ and for every $p \in P$ we define Y_p as the exclusive-or $Z_i \oplus Z_j$ of the particular variables of the pair $p = \{i, j\}$ ($\oplus$ being the exclusive-or operator).

   This will give rise to $|P|$ different Y_p random variables.

   We will also consider the "total" random variable $Y = \sum_{p=1}^{|P|} Y_p$.

   (a) Show that for every $p \in P$, Y_p is 0 with probability $1/2$ and 1 with probability $1/2$. *[4 marks]*
   So despite the unusual definition with $\oplus$, each $Y_{i,j}$ is a fair coin flip.

   (b) Show that the number of different Y_p variables (the cardinality of the set P) will be greater than N. *[4 marks]*

   (c) Show that every pair of the Y_p variables satisfy the definition of *pairwise independence*, and hence that $\mathrm{E}[Y_p Y_q] = \mathrm{E}[Y_p]\mathrm{E}[Y_q]$. *[4 marks]*
   (you will need to consider 2 cases to show this)

   (d) Show that the collection of $\{Y_p : p \in P\}$ variables do *not* satisfy the definition of mutual independence. *[4 marks]*

   (e) What is the expected value $\mathrm{E}[Y]$? *[4 marks]*

   (f) It is well-known that if we have a collection of random variables $\{X_i : 1 \leq i \leq k\}$ such that the X_i are pairwise independent, that $\mathrm{Var}[\sum_{i=1}^{k} X_i] = \sum_{i=1}^{k} \mathrm{Var}[X_i]$. *[4 marks]*
   Use this fact to calculate $\mathrm{Var}[Y]$ for $Y = \sum_{p=1}^{|P|} Y_p$.

   (g) Using Chebyshev's inequality, prove an upper bound on $\Pr[|Y - \mathrm{E}[Y]| \geq n]$. *[6 marks]*

2. Consider a variation on the coupon collector problem where the pupils aim to fill a sticker book (with different footballers), but the book only has space for $n/2$ players. How many cereal packets would we expect pupils to buy before they fill their sticker book? *[10 marks]*

3. Consider a function $F : \{0, 1, \ldots, n-1\} \rightarrow \{0, 1, \ldots, m-1\}$ and suppose we know that for $0 \leq x, y \leq n-1$, $F((x+y) \bmod n) = (F(x) + F(y)) \bmod m$. The only way we know to evaluate $F(\cdot)$ is to examine the values in an array where the $F(\cdot)$ values have been stored (with entry i holding the value of $F(i)$). Unfortunately, a system failure has corrupted up to a $1/5$-fraction of the entries of the array, so we no longer have reliable values in all positions.

   Describe a simple randomized algorithm that, given an input $z \in \{0, \ldots, n-1\}$, outputs a value that equals $F(z)$ with probability at least $1/2$. Your algorithm should guarantee this $1/2$ probability of being correct for every value of z, regardless of which specific array entries were corrupted. Your algorithm should use as few lookups and as little computation as possible. Justify the $1/2$ correctness guarantee. *[10 marks]*

   Suppose you are allowed to repeat your initial algorithm three times before you return a result. What should you do in this case? Justify your answer. *[5 marks]*

4. Recall our analysis of the simple "Max-Cut" (or $\frac{|E|}{2}$-cut) algorithm in Lecture 6, and remember we chose to place each $v \in V$ into S or $V \setminus S$ with even (and independent) probabilities $1/2$; recall also that this generation of S could be considered as choosing a random subset of V (with every individual subset having the probability 2^{-n}, regardless of its size). We showed that when we generate S this way, the expected size of the cut $(S, V \setminus S)$ is exactly $\frac{|E|}{2}$.

   Come up with a different algorithm to generate $(S, V \setminus S)$ in such a way that the expected size of the cut will be the slightly larger value $|E|\frac{|V|}{2|V|-1}$ (hence showing that there is at least one cut of this size). *[20 marks]*

   Note - there will be two slightly different cases, for odd n and even n, and the factors for these will be different (but at least $|E|\frac{|V|}{2|V|-1}$ in each case).

5. (a) Suppose that we can obtain independent samples $X_1, ..., X_m$ of a random variable X supported in $[0, 1]$. We want to use these samples to estimate E[X] to within an additive error of ϵ with failure probability at most δ, for $\delta, \epsilon \in (0, 1)$. Give an algorithm for this problem and analyze its performance. What is the number of samples m used by your algorithm? *[10 marks]*

   (b) Given an undirected graph $G = (V, E)$, where $|V| = n$, we want to assign labels to each vertex such that the sum of the labels of each vertex and its neighbours modulo $n+1$ is nonzero.

   Consider the following randomized algorithm for this problem: a label in the range $\{0, 1, 2, \ldots, n\}$ to each vertex independently and uniformly at random. If the random labelling does not satisfy the required property try again. Show that this algorithm will find a correct labelling with expected time that is polynomial in n. *[15 marks]*

Mary Cryan, 31st January 2019