Complexity for Probabilists

Homework 3

1. In the lecture on January 28, I proved that if the decision problem f is in BPP, then, for any $\delta > 0$, there exists a bounded polynomial time probabilisitc algorithm A^* such that for all x,
$$\mathbf{Pr}[A^*(x) = f(x)] \geq 1 - \delta \,. \tag{1}$$
A^* was constructed as follows. By the definition of BPP, there is an algorithm A such that for all x,
$$\mathbf{Pr}[A(x) = f(x)] \geq \frac{2}{3} \,.$$
A^* consists of running A n times to get answers $A_1(x), \ldots, A_n(x)$, and then outputting $majority(A_1(x), \ldots, A_n(x))$. I showed that (1) holds for n large enough, but I gave no indication of *how* large n needs to be.

   (a) Use Chebyshev's inequality[1] to give an explicit upper bound on how large n needs to be taken. (The answer depends on δ.)

   (b) **Bonus question for the ambitious student:** Can the Chebyshev-based bound be substantially improved?

2. **Bounded polynomial time Las Vegas algorithms can be replaced by polynomial time deterministic algorithms[2].** Here's an example: Let f be a decision problem, assume that there exists a bounded polynomial time probabilistic algorithm A such that for all x,
$$\mathbf{Pr}[A(x) = f(x)] = 1 \,,$$
and show how to modify A in order to get a polynomial time deterministic algorithm A' which computes f.

3. Another thing I did in the lecture on January 28, was to show how to extract biased random bits (with a given bias) from unbiased random bits. I now want you to do the other way around: Suppose that we are given a source of i.i.d.[3] random variables $X_1, X_2, \ldots$ with
$$\mathbf{Pr}[X_i = 1] = 1 - \mathbf{Pr}[X_i = 0] = p \,,$$
where $p \in (0,1)$ is *unknown*. Find a scheme[4] for using $(X_1, X_2, \ldots)$ to obtain another i.i.d. sequence $(Y_1, Y_2, \ldots)$ with
$$\mathbf{Pr}[Y_i = 1] = \mathbf{Pr}[Y_i = 0] = 1/2 \,.$$

[1] See any introductory text on probability.

[2] This problem may give the impression that Las Vegas algorithms with bounded running time are useless. This impression is false, however: Think of the QUICKSORT example of January 28, where the *expected* running time was reduced from $O(n^2)$ to $O(n \log n)$ by means of randomization. The randomized QUICKSORT that I described is a Las Vegas algorithm with running time bounded by $O(n^2)$.

[3] Independent and identically distributed.

[4] In case you cannot think of one, there are several suggestions in the literature; see e.g. Peres, Y. (1992) Iterating von Neumann's procedure for extracting random bits, *Annals of Statistics* **20**, 590–597. However, I much prefer that you think up your own procedure rather than looking into the literature, so please try hard!