CS 70 Discrete Mathematics for CS
Fall 2004 Rao

HW 8

Due on Thursday, Nov. 11 at 3:29PM

1. Coupon Collecting

(a) Let X be the number of tosses of a biased coin with Heads probability p until the first Head appears (i.e. X is a geometric r.v. with parameter p). We have seen in lecture that $\mathbf{E}[X] = \frac{1}{p}$. Show that $\mathbf{Var}[X] = \frac{1-p}{p^2}$. (Hint: You will need to sum a series like $S = \sum_{i=1}^{\infty} i^2 q^i$. One way to do this is to multiply S by q and subtract the result from S: this gives you a series for $(1-q)S$. Now if you look at this series carefully, you will see that you can split it into a series of the form $\sum_i iq^i$ and one of the form $\sum_i q^i$. But you know how to sum both of these:the first is like the expectation of X and the second is just a geometric series.)

(b) Now let X be the r.v. in the coupon collecting problem, i.e. X is the number of cereal boxes we need to buy before we have collected one copy of each of n baseball cards. Recall from lecture that $\mu = \mathbf{E}[X] = n\sum_{i=1}^{n} \frac{1}{i} \approx n(\ln n + \gamma)$. Use the result of part (a) to compute the variance $\mathbf{Var}[X]$. [Note: your answer should contain a sum of the form $\sum_{i=1}^{n} \frac{1}{i^2}$.]

(c) It turns out that the series $\sum_{i=1}^{\infty} \frac{1}{i^2}$ converges to a constant value $C = \frac{\pi^2}{6} \approx 1.645$. Deduce that $Var[X] \leq Cn^2$. Hence deduce the smallest value of β for which you can say that the probability we need to buy more than $\mu + \beta n$ boxes is less than $\frac{1}{100}$.

2. Random bit strings

Consider a random bit string S of length n.

(a) For a given position j in S, what is the probability that it is a starting point of a run of at least l ones?

(b) What is the expected number of places j at which runs of at least l ones start?

(c) Use Markov's inequality to show that the probability that there exists a run of at least $c \log n$ ones is less than $\frac{1}{n^{c-1}}$.

(d) We now consider runs of alternating ones and zeroes that start with a one (e.g. 101010). What is the expected number of places j at which alternating runs that begin with a one and have at least l bits start?

3. The Martingale

Consider a *fair game* in a casino: on each play, you may stake any amount $\$X$; you win or lose with probability $\frac{1}{2}$ each (all plays being independent); if you win you get your stake back plus $\$X$; if you lose you lose your stake.

(a) What is the expected number of plays before your first win (including the play on which you win)?

(b) The following gambling strategy, known as the "martingale," was popular in European casinos in the 18th century: on the first play, stake \$1; on the second play \$2; on the third play \$4; and in general, on the kth play $\$2^{k-1}$. Stop (and leave the casino!) when you first win. Show that if you follow this strategy, and assuming you have unlimited funds available, then you will leave the casino \$1 richer with probability 1. (Maybe this is why the strategy is banned in most modern casinos).

(c) To discover the catch in this seemingly infallible startegy, let X be the r.v. that measures your maximum loss before winning (i.e., the amount of money you have lost *before* the play on which you win.) Show that $\mathbf{E}[X] = \infty$. What does this imply about your ability to play the martingale strategy in practice?

4. Load Balancing Extra Credit

This problem asks you to try out in practice the load balancing scheme discussed in class and to compare the results with the theory. It will also lead you on to try out a a more sophisticated strategy that works even better.

(a) Write a program in any language that simulates the balls and bins experiment, where n balls are thrown into n bins, for a given value of n. Perform 20 simulations with $n = 1000$ and 20 simulations with $n = 10^6$ and draw up histograms of the maximum loads you observe (i.e., each histogram contains 20 data points). How well do these values correspond to those that we derived in class? Hand in your source code.

(b) Now consider the following alternative scheme, which uses a minimal amount of communication: jobs arrive in sequence as before, but instead of simply choosing a single processor at random, each job now chooses *two* processors at random, inspects their current loads and goes to the less heavily loaded of the two. (If both loads are the same, the job chooses one of the two arbitrarily.) Modify the program to implement this scheme. Again, perform 20 simulations with $n = 1000$ and $n = 10^6$ and tabulate the maximum loads you observe. Hand in your source code.