CS 173: Discrete Mathematical Structures, Spring 2009
Homework 9

Due at class on Friday, April 24, 2009

---

1. **[10 points] Pigeonhole Principle**
   Let S be a set of ten distinct integers between 1 and 50. (Distinct means that no two elements of S are the same.) Use the pigeonhole principle to show that there are two different 5-element subsets of S with the same sum.

   The "sum" of a set of numbers is what you get if you add up all the numbers in the set. For example, suppose that $S = \{1, 2, 3, 4, 5, 30, 31, 32, 33, 34\}$. Consider the two subsets $\{30, 2, 3, 4, 5\}$ and $\{31, 1, 3, 4, 5\}$. The sum of the first subset is $30 + 2 + 3 + 4 + 5 = 44$, which is the same as the sum of the second subset, i.e. $31+1+3+4+5$. So these two subsets are different but they have the same sum. You need to show that this works no matter how we choose the ten elements of S.

2. **[10 points] Graph Theory and Degree Sequences**
   The degree sequence for a graph G is the sequence of the degrees of the vertices in the graph arranged in non-increasing order. For example the degree sequence for W_4 would be 4,3,3,3,3.

   (a) Let m and n be positive integers with $n > m$. What is the degree sequence for the graph $K_{m,n}$?

   (b) What simple graph with n vertices has a degree sequence of $n-1, n-1, ..., n-1$?

   (c) Consider a full binary tree T with n vertices. Suppose there are i internal vertices. What would the degree sequence for T look like?

   (d) A sequence $d_1, d_2, ..., d_n$ is called *graphic* if it could be the degree sequence of a simple graph. Is the sequence $6, 5, 4, 3, 2, 1$ graphic? Explain your answer.

3. **[10 points] Conditional Probability and Independence**

   (a) Let E be the event that the bit string of length 5 contains an odd number of 1s. Let F be the event that the string ends with a 0. Are E and F independent events?

   (b) What is the conditional probability that exactly 4 heads appear when a fair coin is flipped 5 times, given the first flip came up tails?

4. **[10 points] Expectation**

   Imagine there are N couples at a party, and suppose m people get sleepy and have to go home. When a person goes home, his or her date has to leave with them. We need to find the expected number of couples left at the party.

   (a) First, let's consider a single couple, couple i. Define a random variable X_i to be:

$$X_i = \begin{cases} 1 & \text{if couple } i \text{ stays} \\ 0 & \text{if couple } i \text{ leaves} \end{cases}$$

   This sort of function is known as an *indicator*.

   What is the expected value of X_i? Clearly explain your answer.

   (b) What is the expected number of couples left at the party? Your answer should be a function of the variables N and m.

   Hint: Suppose we have n random variables X_i and we wish to compute the expected value of $\sum_{i=1}^{n} X_i$. This means we have sum of n functions and we wish to find the expected value of the sum. We can do this by using the *linearity of expectation* which establishes that $\mathbf{E}(\sum_{i=1}^{n} X_i) = \sum_{i=1}^{n} \mathbf{E}(X_i)$. This means that we can find the answer by computing the expectation for each X_i separately and then summing the results.

5. **[10 points] An Algorithm**

Consider the problem of writing a program to verify polynomial identities of the form:

$$(a_1x + a_2y)^n = b_0x^n + b_1x^{n-1}y^1 + ... + b_{n-1}x^1y^{n-1} + b_ny^n$$

where n is a positive integer and the a_i and b_i are real numbers.

One way to verify the identity is to use the binomial theorem to find the coefficients generated by the left hand side and see if they match the b_i's on the right hand side. Consider the following pseudo-code procedures to do that:

```
procedure factorial(k)
fac := 1
for i := 1 to k
      fac := fac · i
return fac


procedure VerifyBinomial( a_1, a_2,b_0..., b_n)
matches := true
for i := 0 to n
begin
      a1pow := 1
      for j := 1 to n − i
            a1pow := a1pow · a_1
      a2pow := 1
      for j := 1 to i
            a2pow := a2pow · a_2
      c := factorial(n)/(factorial(n − i) · factorial(i))
      c := c · a1pow · a2pow
      if(c ≠ b_i)then
            matches := false
end
return matches
```

(a) State a big-Θ bound on the number of multiplications done by the procedure **factorial** in terms of the input k.

(b) State a big-Θ bound on the number of multiplications done by the procedure **VerifyBinomial** in terms of the input n. Use your answer from part (a) when considering how many multiplications **factorial** does.