CPSC-320 Intermediate Algorithm Design and Analysis Winter 2000

Solutions to Practice Homework # 1

Solutions were contributed by John Bent, Yee Eileen Chan, Brian Eith, Jaewoo Kang, Colin Lau, Hillary Paul, Russell Reid, Ian Sadkovich, Dan Sorin, and Fan Yang.

1. For a $k \times k$ tilted subsquare, we need $2k-1$ consecutive rows and columns. The number of ways to choose $2k-1$ consecutive rows or columns from an $n \times n$ square is $n-(2k-1)+1 = n-2k+2$. Therefore the number of $k \times k$ tilted subsquare in $n \times n$ square is $(n-2k+2)^2$. To get all the tilted subsquares, sum over k to get

$$\sum_{k=1}^{\lfloor (n+1)/2) \rfloor} (n-2k+2)^2.$$

To find a closed form expression for this summation, we guess that the summation is a cubic polynomial of the form $An^3 + Bn^2 + Cn + D$. To find the four coefficients, make 4 equations by using 4 different values of n, say 1, 2, 3, 4.

$$\begin{aligned} \text{for } n = 1: \quad & A + B + C + D = 1, \\ \text{for } n = 2: \quad & 8A + 4B + 2C + D = 4, \\ \text{for } n = 3: \quad & 27A + 9B + 3C + D = 10, \\ \text{and for } n = 4: \quad & 64A + 16B + 4C + D = 20. \end{aligned}$$

Solving these equations yields $A = 1/6, B = 1/2, C = 1/3, D = 0$. Therefore the closed form of the summation is:

$$1/6n^3 + 1/2n^2 + 1/3n.$$

Here is another solution. By looking at the pattern for small values of n, one can see that if n is odd, there are $1^2 + 3^2 + 5^2 + \ldots + n^2$ tilted subsquares. If n is even, there are $0^2 + 2^2 + 4^2 + \ldots + n^2$ tilted subsquares.

We claim that, both for n even and n odd, these summations are equal to $n(n+1)(n+2)/6$. To see this, note that the sum for n even is 4 times $\sum_{k=1}^{n/2} k^2$, for which a closed form is already in the lecture notes. The odd sum can be viewed as $\sum_{k=1}^{n} k^2$ minus the sum $0^2 + 2^2 + 4^2 + \ldots + (n-1)^2$. But we already know the closed form expressions for both of these summations; by taking the difference we get the answer.

2. (a)
```
Choose 6 golf balls uniformly at random
   Place 3 of the chosen balls on each side of the balance
   If one of the arms is the heavier arm (50% probability), then
   a)  Choose 2 of the three heavier golf balls and weigh them
   b)  If one of the balls is heavier, we are done
   c)  If the balls weigh the same, then the third ball is the heavier ball
   Otherwise, place 3 of the 6 remaining balls on each side of the balance
```

```
If one of the arms is the heavier arm, then
a) Choose 2 of the three heavier golf balls and weigh them
b) If one of the balls is heavier, we are done
c) If the balls weigh the same, then the third ball is the heavier ball
```

With this algorithm, half of the time, the heavy ball will be in the first group of balls, and therefore two weighings will be required. The other half of the time, the heavy ball will not be in the first group of balls and therefore three weighings will be required. The expected number of weighings is: $(1/2) \times 2 + (1/2) \times 3 = 2.5$.

(b)(Thirteen Golf Balls:)

```
Choose 6 golf balls uniformly at random
Place 3 of the chosen balls on each side of the balance
If one of the arms is the heavier arm (6/13) probability), then
a) Choose 2 of the three heavier golf balls and weigh them
b) If one of the balls is heavier, we are done
c) If the balls weigh the same, then the third ball is the heavier ball
Otherwise choose 6 of the remaining 7 uniformly at random
Place 3 of the chosen balls on each side of the balance
If one of the arms is the heavier arm (6/13) probability), then
   a) choose 2 of the three heavier golf balls and weigh them
   b) If one of the balls is heavier, we are done
   c) If the balls weigh the same, then the third ball is the heavier ball
Otherwise (1/13) the remaining ball is the heavier ball and we are done
```

With this algorithm, (7/13)ths of the time the heavy ball can be found with two measurements (6/13 from the original 6, plus 1/13 from the remaining ball), and 6/13ths of the time three measurements are required. Thus the number of expected weighings is

$$(6/13) \times 2 + (1/13) \times 2 + (6/13) \times 3 = 32/13 = 2.46.$$

A very smart solution for 12 balls is as follows. Choose 6 balls first, place 3 of the chosen balls on each side of the balance. If one arm is heavier (probability 1/2), use the same way as mentioned above (2 weighings are sufficient.) Otherwise,we know no one of those 6 chosen balls is heavier.Let us call the left 6 balls as suspects. Weight 3 suspects against two suspects and one known. If one arm is heavier(probability 5/6), one more weigh is needed(3 weighings altogether). Otherwise(probability 1/6), we are done (2 weighings).

So average weighing times is

$$2 \times 1/2 + 1/2 \times (3 \times 5/6 + 2 \times 1/6) = 2.42$$

One thing to be careful about when analyzing these algorithms is to make sure that each case gets the appropriate weight. Can you see the problem with the following analysis of the 13-ball algorithm as described above:

The expected number of weighings is less than 2.5 because we have two cases of 2 weighings and one 3 weighings case. The first 2 weighing case is if the heavy ball is in the first 6 balls chosen. The second 2 weighing case is if the ball is not in the first or second group of 6 but the one not chosen. The 3 weighing case is if the ball is in the second group of 6. So, $1/3 \times 2 + 1/3 \times 2 + 1/3 \times 3 = 2.3333$.

3. Exhaustively try all combinations of colors. At 3 colors per node, there are 3^n arrangements of colors to test. If testing a fixed color scheme is counted as one step, the algorithm grows as 3^n. (In fact, testing a fixed color scheme could take time proportional to the size of the graph, because it is necessary to examine each edge of the graph and check that the nodes at the endpoints of the edge have distinct colors, but for simplicity assume we are only counting the number of color schemes tested.)

   b) For 10 nodes, the time is 3^{10}ns = 59 usec. For 50 nodes, the time is 3^{50}ns = 7.2×10^{14} seconds = 22831050 years. For 100 nodes, the time is 3^{100}ns = 5.2×10^{38} seconds = 1.61×10^{31} years.

   Can you find a solution that tests a number of color schemes that is at most 2^{n-1}? Hint: Start by thinking of the case where the input graph is connected. Once the color of a node a is fixed by a given color scheme, there are at most two choices for the colors of the nodes adjacent to a. How can you take advantage of that?