CS 561, HW 4

Prof. Jared Saia, University of New Mexico

1. In this problem, you will prove that for a recurrence of the form $T(n) = aT(n/b) + f(n)$, if $af(n/b) \geq Kf(n)$ for some constant $K > 1$, then $T(n) = n^{\log_b a}$. Assume that for any $x \geq b$, $af(x/b) \geq Kf(x)$ for some fixed $K > 1$. Let the last level of the recursion tree be $\ell = \log_b n$.

   (a) Prove by induction that for any $i \in [0, \ell]$, $a^i f(n/b^i) \geq K^i f(n)$. Don't forget to say which variable you're doing induction on, and label the BC, IH and IS.

   (b) Using the above result, upper bound the sum of all levels of the recursion tree, i.e. upper bound $\sum_{i=0}^{\ell} a^i f(x/b^i)$. Hint: Let $L = \Theta(a^\ell)$ be the cost of the last level and argue that the cost of the j-th level of the recursion tree above ℓ is at most L/K^j. Now use what we showed in class about geometric summations. Finally use log facts to simplify L.

2. Consider the recurrence $f(n) = 4f(n/3) + \sqrt{n}$

   (a) Use the Master method to solve this recurrence

   (b) Now use annihilators (and a transformation) to solve the recurrence.

3. Consider the function:

```
int f (int n){
  if (n==0) return 2;
  else if (n==1) return 5;
  else{
    int val = 2*f (n-1);
    val = val - f (n-2);
    return val;
  }
}
```

(a) Write a recurrence relation for the *value* returned by f. Solve the recurrence exactly. (Don't forget to check it)

(b) Write a recurrence relation for the *running time* of f. Get a tight upper bound (i.e. big-O) on the solution to this recurrence.

4. A bakery sells donuts in boxes of three different quantities, x_1, x_2, and x_3. In the Donut Buying problem, you are given the numbers x_1, x_2 and x_3, and an integer n and you should return either 1) the minimum number of boxes needed to obtain exactly n donuts if this is possible, along with a set of boxes that obtains this minimum; or 2) "DOH!" if it is not possible to obtain exactly n donuts.

   For example if $x_1 = 4$, $x_2 = 6$, $x_3 = 9$ and $n = 17$, then you should return that 3 boxes suffices, with 2 boxes of size 4, and 1 box of size 9. However, if $n = 11$, you should return "DOH!" since it is not possible to buy exactly 11 donuts with these box sizes.

   (a) For any positive x, let $m(x)$ be the minimum number of boxes needed to buy x donuts if this is possible, or INFINITY otherwise. Write a recurrence relation for the value of $m(x)$. Don't forget the base case(s)!

   (b) Give an efficient algorithm for solving Donut Buying. How does its running time depend on x_1, x_2, x_3, and n? Is it an algorithm that runs in polynomial time in the input sizes?

5. A thief repeatedly robs the same bank. To avoid capture, he decides to never rob the bank fewer than 10 days after the last robbery. He has obtained information, for the next n days, on the amount of money b_i that is held at the bank on day i.

   (a) Let $r(i)$ be the maximum amount of revenue the thief can safely obtain from day 1 through day i. Give a recurrence relation for $r(i)$.

   (b) Describe a dynamic program based on this recurrence. What is the runtime of your algorithm?