CS 461, HW2

Prof. Jared Saia, University of New Mexico

Due: September 23rd, 2003

If you have an older version of the book, check with a friend to make sure you get the right numbers for exercises (the numbers are different for the first and second editions)

1. Consider the recurrence $T(n) = 2T(n/4) + n^2$
   (a) Use the recursion tree method to get a tight upper bound (i.e. big-O) on the solution to this recurrence
   (b) Now use annihilators (and a transformation) to get a tight upper bound on the solution to this recurrence. Show your work. (Note that your two bounds should match)
2. Consider the recurrence $T(n) = 2T(n/2) + \log^2 n$
   (a) Use the Master method to get a general solution to this recurrence.
   (b) Now use annihilators (and a transformation) to get a tight upper bound on the solution to this recurrence. Show your work. (Note that your two bounds should match)
3. Consider the following function:

```
int f (int n){
  if (n==0) return 0;
  else if (n==1) return 1;
  else{
    int val = 6*f (n-1);
    val = val - 9*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 upperbound (i.e. big-O) on the solution to this recurrence.

4. Give the table for the optimal string alignment of the strings "abbaba" and "ababbab". Make sure you include arrows in your table the same way we did in the table given in lecture, these arrows will help you reconstruct the solution. List all optimal alignments of these two strings.

5. Find the optimal parenthesization for a matrix-chain product whose sequence of dimensions is: $(2, 3, 1, 3, 1)$. (Don't forget to include the table used to compute your result)

6. Exercise 15.2-3

7. Exercise 15.2-5 (hint: use proof by induction)

8. Exercise 15.4-1

9. Exercise 15.4-5 (note: monotonically increasing means non-decreasing, e.g. 1, 2, 2, 4, 5, 5, 7)

10. Extra Credit: Exercise 15.4-6

11. Extra Credit: Problem 15-2