COMP3270 Algorithms, Sample Midterm

Richard Chapman

Fall Semester, 2014

Directions The test is open book and open notes, but NOT open phone or open computer (e-reader or computer used solely as e-reader is ok). For each problem, show your work completely. Give reasons for all answers – this is how I give partial credit. **Each part of each question is worth 10 points, 100 total points)**

1. Suppose an algorithm, along the lines of Strassen's algorithm, were discovered that allowed one to multiply two matrices of size $n \times n$ by doing six multiplications of matrices of size $n/2 \times n/2$ and twenty additions of $n/2 \times n/2$ matrices. Give a big-theta bound of the time complexity to multiply two $n \times n$ matrices (using this hypothetical algorithm) as a function of n.

2. For each of the following recurrence relations, give as tight a bound as possible on the running time of an algorithm whose running time $T(n)$ is described by the recurrence relations. Use the Master Method.

   (a) $T(n) = T(4n/5) + O(1)$

(b) $T(n) = 2T(n/2) + O(n)$

(c) $T(n) = T(n-1) + O(1)$

3. For part (b) in the previous problem, solve the recurrence using the Recursion Tree method.

4. For the following bit of pseudocode, give a big-theta running time boundary

```
for i = 1 to n
   for j = i to n
      c[i][j] = a[i][j]+b[i][j]
for i = 1 to n
   for j = i to n
      c[j][i] = 0
```

5. Express the following function in terms of big-theta notation

$$f(n) = n^3 lgn - 2n^3 + n^4$$

6. True or false? $n^2 = O(n^2 lgn)$ Justify your answer

7. In the previous question, is n^2 polynomially bounded by $O(n^2 lgn)$ Why or why not?

8. Give a function $f(n)$, such that $f(n) = o(n^3)$ and $f(n) = \omega(n^2)$. Justify your answer.