CS 4720/5720 Design and Analysis of Algorithms

Homework #5b

Due: Thursday, December 6, 2018

If written: due in class

If typed and submitted on Canvas: 11:59 pm

Submission requirements:

1. **5% extra credit** if you submit *digital, typed* writeups in PDF format to the "Homework 4 Writeup" assignment on the Canvas site. However, you may also turn in handwritten assignments in class – but assignments submitted in person will not receive the 5% extra credit. The due dates are the same for either submission method, but I will need handwritten assigments turned in to me in class.
2. Note: this is a "short" homework with no programming assignment, and will be worth 50 points, half as much as a normal homework.

Assignment:

1. *Lower bounding the complexity of array search:* We have studied two algorithms for finding the maximum value in an un-ordered array. One of these is iterative (walk down the array and remember the largest value you've seen) and the other is recursive (cut the array in half and recursively find the maximum in both halves). Both have $\Theta(n)$ complexity.

   Show that every comparison-based algorithm for finding the maximum value in an un-ordered array performs at least $n-1$ comparisons.

2. *Tractability.*

   (a) A particular problem can be solved using an algorithm with $\Theta(n \log \log n)$ running time. Is this problem
   - Tractable,
   - Intractable, or
   - Impossible to tell?

   (b) A particular problem can be solved using an algorithm with $\Theta(5^n)$ running time. Is this problem
   - Tractable,
   - Intractable, or
   - Impossible to tell?

   (c) For some particular problem, the best-known algorithm has $\Theta(n!)$ running time. Is this problem
   - Tractable,

- Intractable, or
- Impossible to tell?

3. In Homework 2, we studied a particular greedy algorithm for the Knapsack problem. This algorithm sorted the items decreasing by value, and went down the list putting the best of these into the knapsack. **Show that this greedy algorithm has an arbitrarily-poor performance ratio (that is, show that you can manipulate the numbers to force the greedy algorithm to perform as ineffectively as you wish).**