CS 3343 Analysis of Algorithms – Spring 09

1/20/09

# 1. Homework

Due **1/27/09** before class

1. **Code snippets (4 points)**

   For each of the code snippets below give their Θ-runtime depending on $n$. Justify your answers.

   (a) **(2 points)**

   ```
   for(i=n; i>=1; i=i-1){
     for(j=1; j<=i; j=j+1){
         print(" ");
     }
   }
   ```

   (b) **(2 points)** Note that `log` is base 2.

   ```
   for(i=1; i<=n; i=i+1){
     for(j=1; log(j)<=n; j=j+1){
         print(" ");
     }
   }
   ```

   (c) **(2 points)**

   ```
   for(i=5; i<=n; i=2*i){
     print(" ");
   }

   for(i=5; i<=5*n; i=i+1){
     print(" ");
   }
   ```

2. ***O* and Ω (4 points)**

   Show using the definitions of big-Oh and Θ:

   (a) If $f_1(n) \in O(g(n))$ and $f_2(n) \in O(g(n))$ then $f_1(n) + f_2(n) \in O(g(n))$.

   (b) If $f_1(n) \in \Theta(g(n))$ and $f_2(n) \in \Theta(g(n))$ then $f_1(n) + f_2(n) \in \Theta(g(n))$.

Flip over to back page ⟹

3. **Smart linear search (4 points)**

Let $A[1..n]$ be an array of $n$ numbers that is sorted in ascending order. Consider searching $A$ for an element $x$ using *smart linear search*. Smart linear search goes through the array from beginning to end as follows: It compares $A[1]$ to $x$. If both are equal then $x$ has been found and the index 1 is returned. Only if $x$ is greater than $A[1]$ the search continues in the same way with $A[2]$, otherwise the algorithm ends reporting that $x$ was not found. This continues with $A[3]$, $A[4]$,..., either until $x$ was found, the algorithm reports that $x$ was not found, or the end of the array has been reached.

- (2 points) Write pseudocode for this algorithm.
- (2 points) Give best-case and worst-case running times (and example inputs attaining these runtimes) in $\Theta$-notation. Note that the input consists of both $A$ and $x$.

4. **Big-Oh ranking (7 points)**
Rank the following functions by order of growth, i.e., find an arrangement $f_1, f_2, ...$ of the functions satisfying $f_1 \in O(f_2)$, $f_2 \in O(f_3)$,... . Partition your list into equivalence classes such that $f$ and $g$ are in the same class if and only if $f \in \Theta(g)$. For every two functions $f_i, f_j$ that are adjacent in your ordering, prove shortly why $f_i \in O(f_j)$ holds. And if $f$ and $g$ are in the same class, prove that $f \in \Theta(g)$.

$$2n\sqrt{n},\ n^2,\ 2n^2+2n+2,\ \log n,\ 2^n,\ n\log n,\ 2^{n+1}$$

Bear in mind that in some cases it might be useful to show $f(n) \in o(g(n))$, since $o(g(n)) \subset O(g(n))$. If you try to show that $f(n) \in o(g(n))$, then it might be useful to apply the rule of l'Hôpital which states that

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{f'(n)}{g'(n)}$$

if the limits exist; where $f'(n)$ and $g'(n)$ are the derivatives of $f$ and $g$, respectively.