Homework 2 - Analysis of Runtimes

Algorithms I - CSCI 700 - Prof. Rosenberg

Due February 10 at 6:30pm

Problem 1) (10 points) Show that if \( T(n) = 6n^4 + 5n^3 + n^2 + 6 \), \( T(n) = \Theta(n^4) \).

Problem 2) (20 points) Based on Corman, et al. Problem 3-3a. Ordering by asymptotic growth rates. Rank the following function by order of growth; that is, find an arrangement of \( g_1, g_2, \ldots, g_{15} \) of the functions satisfying \( g_1 = \Omega(g_2), g_2 = \Omega(g_3), \ldots, g_{14} = \Omega(g_{15}) \). Partition your list into equivalence classes such that \( f(n) \) and \( g(n) \) are in the same class if and only if \( f(n) = \Theta(g(n)) \).

\[
\begin{array}{cccccc}
   n^4 & \sum_{i=1}^{n} 1 & \log \log n & 2010 & \sum_{i=1}^{n} i \\
   2^n & \sqrt{n} & \log n & n^2 & n \log n \\
   n^n & \sum_{i=1}^{n} \frac{1}{i} & n! & e^n & n \\
\end{array}
\]

Problem 3) Assume an input size of \( n \). Both Algorithms A and B perform the same function. Algorithm A has a runtime of \( f(n) \), and Algorithm B has a runtime of \( g(n) \). If \( f(n) = \Omega(g(n)) \) then, for sufficiently large values of \( n \), Algorithm B should be preferred, all else being equal. However, for small values of \( n \) this is not always true. For some small inputs, Algorithm A will produce faster runtimes. In this problem, you are asked to identify the "sufficiently large" integral input size \( k \) where \( f(n) > g(n) \) for all values of \( n \) such that \( n > k \). If \( f(n) \neq \Omega(g(n)) \) show why not.

3a. (5 points) \( f(n) = n; g(n) = 64\sqrt{n} \)
3b. (5 points) \( f(n) = n!; g(n) = 256n^5 + 128n^2 - 16n + 1024 \)
3c. (5 points) \( f(n) = \frac{1}{1023}n; g(n) = n\log 16n \)
3d. (5 points) \( f(n) = n^2; g(n) = \sum_{i=1}^{n} 2i + 1 \)
Problem 4) (25 points) Let $S_n$ be a sequence of numbers for all $n \geq 0$. $S_0 = 0$. Let $S_n = 2S_{n-1} + 1$ for all $n > 0$. Prove by induction that $S_n = 2^n - 1$ for all values of $n \geq 0$.

Problem 5) Based on Corman Problem 2-3. Horner’s rule for computing polynomials. The following code fragment implements Horner’s rule for evaluating a polynomial

$$P(x) = y = \sum_{k=0}^{n} a_k x^k$$

$$= a_0 + x(a_1 + x(a_2 + \ldots + x(a_{n-1} + xa_n)\ldots)),$$

given the coefficients $a_0, a_1, \ldots, a_n$ and a value for $x$:

1: $y \leftarrow 0$
2: $i \leftarrow n$
3: while $i \geq 0$ do
4: $y \leftarrow a_i + x \cdot y$
5: $i \leftarrow i - 1$
6: end while
7: return $y$

5a. (10 points) What is the asymptotic running time of this code fragment for Horner’s rule? Show the derivation.

5b. (15 points) Prove that the following is a loop invariant for the while loop in lines 3-5,

$$y = \sum_{k=0}^{n-(i+1)} a_{k+i+1} x^k$$

Interpret a summation with no terms as equalling 0. Your proof should follow the structure of the loop invariant proof presented in Corman section 2.1 and should show that, at termination, $y = \sum_{k=0}^{n} a_k x^k$. 