

C241 Homework 10: Time Complexity

Due Monday 11/30/09

1) Find the asymptotic complexity (the closest fitting Big-O: $O(1)$, $O(\log_2(n))$, $O(n \log_2(n))$, $O(n^2)$, etc...) for the following functions:

a) $f(n) = 3n + 7$

b) $f(n) = 3 + \sin(1/n)$

c) $f(n) = 3n^3 - 5n^2 + 25n - 165$

d) $f(n) = 5n^2 + 3n \log_2(n)$

e) $f(n) = n^2 + (n - 1)^3$

f) $f(n) = \frac{n(n+1)(n+2)}{(n+3)}$

g) $f(n) = 2 + 4 + \dots + 2n$

h) $f(n) = 56n^3 + 2^n$

2) Let $f(n) = n + 100$ and $g(n) = n^2$.

a) Use the definition of Big-O to prove that $f(n) \in O(g(n))$

b) Use the definition of Big-O, and a proof by contradiction, to prove that $g(n) \notin O(f(n))$

3) Let $f(n) = n$ and $g(n) = n + (1/n)$.

a) Use the definition of Big-O to prove that $f(n) \in O(g(n))$

b) Use the definition of Big-O to prove that $g(n) \in O(f(n))$

4) Use Numerical Induction on k and the definition of Big-O to prove that for all $k \in \mathbb{N}$: $A_0n^0 + A_1n^1 + A_2n^2 \dots + A_{k-1}n^{k-1} + A_kn^k \in O(n^k)$. Here A_0, A_1, \dots, A_n represent any arbitrary coefficients chosen from \mathbb{N} .

5) Find the time complexity of the following pieces of code:

a)

```
for(i : 1 to n)
```

```
  x = x + 1
```

b)

```
for(i : 1 to n)
```

```
  for (j:1 to m)
```

```
    x = x + 1
```

c) (note that the index k is used in both loops)

```
for (k : 1 to n)
```

```
  for (j: 1 to k)
```

```
    x = x + 1
```

```
d)
for (i : 1 to n)
  y = y + 1
  for (h: 1 to p)
    z = z + 3
  for (k: 1 to q)
    x = x + 1
```

```
e)
for (k : 1 to n)
  for (j: 1 to  $n^2$ )
    x = x + 1
```

Proofs:

6) **Easy Proof:** Prove that following problem can be solved in linear time, by describing in simple pseudo-code how you would solve it and showing that your algorithm takes time $O(n)$: Given an list of n numbers, what's the smallest number in the list?

7) **Medium/Hard Proof:** Describe in simple pseudo-code an algorithm to solve the following problem. Show that you can solve it in $O(n^2)$ time (do better than that for extra-credit): Given an array of n positive integers, return an array with the numbers in sorted order from least to greatest.

8) (extra credit, possibly fortune and fame) **Very Hard Proof:** The 'Satisfaction' problem is the classical example of an NP-complete problem: This means that no one has found an algorithm solve it which takes time less than $O(2^n)$, but we also haven't yet proven that that's impossible. Still, there are tricks which can speed the solution up for most inputs, and some of those aren't too hard to figure out. Take a shot at solving this problem yourself, and I'll give you some extra credit.

The Satisfaction Problem: Given a logic statement with n variables, in a form similar to: $(v_1 \vee v_2 \vee v_3) \wedge (\neg v_2) \wedge (v_2 \vee v_4)$ (A series of 'or' clauses, connected by 'ands', with the 'nots' only applied directly to the variables, not to clauses. This is called 'CNF' form): Is the statement a contradiction, or is it possible for the statement to evaluate to True (be 'satisfied')?