COP4530 Recitation Fall 2012
Week 4

Objective

1. Time Complexity Review

Formal Definitions

A. Big O

$f(n) = O(g(n))$ iff $\exists\, c, n_0 > 0 \mid 0 \leq f(n) \leq cg(n)\ \forall\, n \geq n_0$
f(n) is asymptotically upper bounded by g(n).

B. Big Ω

$f(n) = \Omega(g(n))$ iff $\exists\, c, n_0 > 0 \mid 0 \leq cg(n) \leq f(n)\ \forall\, n \geq n_0$
f(n) is asymptotically lower bounded by g(n).

C. Big Θ

$f(n) = \Theta(g(n))$ iff $\exists\, c_1, c_2, n_0 > 0 \mid 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)\ \forall\, n \geq n_0$
f(n) has the same long-term growth rate as g(n).

Review

Find the time complexity of the following scenarios.

A. Loops

```
for (i = 0; i < n; ++i) {
    // 4 atomics
}
```

B. Loops with break

```
for (i = 0; i < n; ++i) {
    // 4 atomics
    if (condition) break;
}
```

C. Sequential search of unsorted vector

```
for (i = 0; i < n; ++i) {
    if (a[i] == x) return true;
}
return false;
```

D. If-then-else

```
if (condition) {
      // 3 atomics;
} else {
    for (i = 0; i < n; ++i) {
        // 2 atomics
    }
}
```

E. Consecutive statements

```
for (i = 0; i < n; ++i) {
    // 10 atomics
}
for (i = 0; i < n; ++i) {
    // 5 atomics
}
```

D. Nested statements

```
for (i = 0; i < n; ++i) {
    // 10 atomics
    for (j = 0; j < n; ++j) {
        // 5 atomics
    }

}
```

Exercises

Question 1

Assume that the time complexity of an algorithm on input of size n is $4n^3$. If the algorithm takes s seconds to execute on some computer, on an input of size n, then how many seconds will it take on an input of size 3n?

Question 2

Suppose we know the time complexity of two search algorithms:

$$t_1(n) = 100n + n^2$$
$$t_2(n) = 10n^2$$

Which algorithm should we use if n is typically less than 10? If n is typically greater than 100?

Question 3

Suppose we know the time complexity of two algorithms for inserting, deleting, and searching information from a database

A: insert = n, delete = $\log n$, search = 1
B: insert = $\log n$, delete = $\log n$, search = $\log n$

*Note that insertion and deletion actually change the number of database entries, but we will ignore that for the time being. Just assume that n is some fixed, large number.

Which algorithm will you use if you will be doing frequent insertions and deletions but rarely a search? Which algorithm will you use if you will be doing frequent searches but rarely an insert or delete?

Question 4

Prove $n^3 + n = O(n^3)$ directly from the definition of Big-O. Show c, n that will satisfy the definition.

Question 5

What is the time complexity of a binary search on a sorted vector?

Question 6

What is the time complexity of the following recursive codes?

```
int factorial (int n) {
    if (n <= 1)
        return 1;
    else
        return (n * factorial(n - 1));
}
```

```
unsigned int fibonacci (unsigned int n) {
    if (n <= 1)
        return 1;
    else
        return fib(n - 1) + fib(n - 2);
}
```