Lecture 22: Analysis of Algorithms

Date: October 23, 2019.

input: `array A[1 .. n] of integers`

```
for j = 2 to n do
    m = A[j]
    if (m < A[j-1]) then
        A[j] = A[j-1]
        A[j-1] = m
return A[n]
```

Figure 1: Algorithm 1

input: `array A[1 .. n] of integers`

```
for j = 2 to n do
    m = A[j]
    i = j-1
    while (A[i] > m) do
        A[i+1] = A[i]
        A[i] = m
        i = i-1
return A[n]
```

Figure 2: Algorithm 2

Problem 1. Describe the computation of Algorithms 1 and 2 on input $A = [4, 5, 3, 2, 1]$. What problems do Algorithms 1 and 2 solve?

Measuring Efficiency of Algorithms.

Question 1. What is the running time of Algorithms 1 and 2?

Problem 2. Intel's P5 Pentium chip had a clock speed of 100 MHz. AMD's FX-8370 chip has a clock speed of 8.723 GHz.

1. How many steps does Algorithm 1 take on the worst case input of size n?
2. How many steps does Algorithm 2 take on the worst case input of size n?
3. How much time does Algorithm 1 running on a Pentium take on an input of size 10^6 in the worst case?
4. How much time does Algorithm 2 running on a AMD FX-8370 take on an input of size 10^6 in the worst case?

Big Oh. For $f, g : \mathbb{N} \to \mathbb{N}$, we say that $f = O(g)$ iff there is c, k such that for every $n \geq k$, $f(n) \leq cg(n)$.

Problem 3. Show that $\frac{(n-1)n}{2} = O(n^2)$ and $n^2 = O(\frac{(n-1)n}{2})$.

Proposition 1. *For any k, and $a_0, a_1, \ldots a_k$, $\sum_{i=0}^{k} a_i x^i = O(x^k)$.*

Theta Notation. For function $f, g : \mathbb{N} \to \mathbb{N}$, $f = \Theta(g)$ iff $f = O(g)$ and $g = O(f)$.

Little Oh. For functions $f, g : \mathbb{N} \to \mathbb{N}$, we say $f = o(g)$ (f is **asymptotically smaller** than g) iff

$$\lim_{n \to \infty} \frac{f(n)}{g(n)} = 0$$

Problem 4. Show that $n^2 = o(2^n)$.