I. Core / A. Correctness and efficiency of algorithms

- Review of algorithms, correctness, and efficiency (last week Friday and this past Wednesday)
- Asymptotics (**today** and next week Monday)

Today:

- Introduce first problem set
- Problem 2-3 (daily work included part c)
- Meaning of asymptotic notation
- Theorem 3.1; Ex 3.1-(4 & 5)
- Properties of asymptotic notation

2-3. Horner's rule for evaluating a polynomial:

$$
\begin{aligned}
P(X) &= \sum_{k=0}^{n} a_k x^k \\
&= a_0 + x(a_1 + x(a_2 + \cdots x(a_{n-1} + x a_n)\cdots))
\end{aligned}
$$

a. $\Theta(n)$.

b. What's the naïve way? How naïve?

$y = 0$
$z = 1$
$i = 0$
while $i \leq n$
    $y = y + a_i \cdot z$
    $z = z \cdot x$
    $i = i + 1$

z keeps a running power of x. If we were computing x^n froms scratch, this would make each exponentiation $\Theta(n)$, so we would have $\Theta(n^2)$ total. (However, the book does say on pg 24 that we can assume exponentiation with small integer exponents are constant time.)

Both the given and my way are $\Theta(n)$. The difference is in the constant: 3 ops and 2 assignments vs 4 ops and 3 assignments.

Proof of Horner's rule loop invariant $\left(y = \sum_{k=0}^{n-(i+1)} a_{k+i+1} x^k \right)$:

Init. *After 0 iterations, $y = 0$, $i = n$ by assignment. So*

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

Maint. *Now, suppose this holds true after N iterations, that is*

$$y_{old} = \sum_{k=0}^{n-(i_{old}+1)} a_{k+i_{old}+1} x^k$$

where y_{old} and i_{old} are y and i after N iterations. Likewise, let y_{new} and i_{new} be the values after $N+1$ iterations.

By assignment $i_{new} = i_{old} - 1$. *Then*

$$
\begin{aligned}
y_{new} &= a_{i_{old}} + x \cdot y_{old} && \textit{by assignment} \\
&= a_{i_{old}} + x \cdot \sum_{k=0}^{n-(i_{old}+1)} a_{k+i_{old}+1} x^k && \\
&= a_{inew+1} + x \cdot \sum_{k=0}^{n-(inew+2)} a_{k+inew} x^k && \textit{by substitution} \\
&= a_{inew+1} + \sum_{k=0}^{n-(inew+2)} a_{k+inew} x^{k+1} && \textit{by distribution} \\
&= a_{inew+1} + \sum_{k=1}^{n-(inew+1)} a_{k+inew+1} x^k && \textit{by change of variables} \\
&= a_{inew+1} x^0 + \sum_{k=1}^{n-(inew+1)} a_{k+inew+1} x^k && \\
&= \sum_{k=0}^{n-(inew+1)} a_{k+inew+1} x^k && \square
\end{aligned}
$$

Formal definition of big-Theta:

$$\Theta(g(n)) = \{f(n) \mid \exists\ c_1, c_2, n_0 \in \mathbb{N} \text{ such that } \forall\ n \geq n_0, 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)\}$$

$g(n) = \frac{1}{2}n^2 - 3n = \Theta(n^2)$.

Proof. *Let* $c_1 = \frac{1}{14}$, $c_2 = \frac{1}{2}$ *and* $n_0 = 7$. *Suppose* $n > 7$. *Then*

$$\begin{array}{rcccl} \frac{1}{14} & = & \frac{1}{2} - \frac{3}{7} < \frac{1}{2} & & \\ \frac{1}{14} & \leq & \frac{1}{2} - \frac{3}{n} & \leq & \frac{1}{2} \\ \frac{n^2}{14} & \leq & \frac{1}{2}n^2 - 3n & \leq & \frac{n^2}{2} \\ c_1 n^2 & \leq & g(n) & \leq & c_2 n^2 \end{array}$$

Therefore $g(n) = \Theta(n^2)$ *by definition.* $\square$

Theorem 3.1. For any two functions $f(n)$ and $g(n)$, we have $f(n) = \Theta(g(n))$ iff $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$.

Theorem 3.1. For any two functions $f(n)$ and $g(n)$, we have $f(n) = \Theta(g(n))$ iff $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$.

Proof. *Suppose $f = \Theta(g(n))$. Then, by definition of Θ, there exist constants c_1, c_2, and n_0 such that for all $n \geq n_0$,*

$$0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)$$

Let $c = c_2$. Then $0 \leq f(n) \leq c \cdot g(n)$, hence $f(n) = O(g(n))$ by definition. Similarly, let $c = c_1$. Then $0 \leq c \cdot g(n)$, hence $f(n) = \Omega(g(n))$.

Conversely, suppose $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$. By the definitions, there exist c, and n_1 such that for all $n \geq n_1$, $0 \leq f(n) \leq c \cdot g(n)$, and there exist c', and n_1' such that for all $n \geq n_1'$, $0 \leq c' \cdot g(n) \leq f(n)$.
Let $c_1 = c'$, $c_2 = c$, and $n_0 = max(n_1, n_1')$. Hence $f(n) = \Theta(g(n))$. $\square$

3.1-4. Is $2^{n+1} = O(2^n)$? Is $2^{2n} = O(2^n)$?

3.1-4. Is $2^{n+1} = O(2^n)$? Is $2^{2n} = O(2^n)$?

To see that $2^{n+1} = O(2^n)$, note that $2^{n+1} = 2 \cdot 2^n$. Thus 2 is the constant we're looking for, and we're done.

Let's attempt a proof that $2^{2n} = O(2^n)$. Does $\exists\ c, n_0 \mid \forall\ n \leq n_0, 2^{2n} \leq c \cdot 2^n$? If so, then

$$\begin{array}{rcl} 2^n \cdot 2^n & \leq & c \cdot 2^n \\ 2^n & \leq & c \end{array}$$

...which is impossible.

For next time

Do Problems 3-1.d and 3-4.(a,b,c)

Read Section 4.1