Chapter 0
Prologue

Solutions

0.1. In each of the following situations, indicate whether $f = O(g)$, or $f = \Omega(g)$, or both (in which case $f = \Theta(g)$).

	$f(n)$	$g(n)$	
a	$n - 100$	$n - 200$	$f = \Theta(g)$
b	$n^{1/2}$	$n^{3/2}$	$f = O(g)$
c	$100n + log\, n$	$n + (log\, n)^2$	$f = \Theta(g)$
d	$n + log\, n$	$10n\, log\, 10n$	$f = \Theta(g)$
e	$log\, 2n$	$log\, 3n$	$f = \Theta(g)$
f	$10\, log\, n$	$log(n^2)$	$f = \Theta(g)$
g	$n^{1.01}$	$n\, (log^2 n)$	$f = \Omega(g)$
h	$n^2 / log\, n$	$n\, (log\, n)^2$	$f = \Omega(g)$
i	$n^{0.1}$	$(log\, n)^{10}$	$f = O(g)$
j	$(log\, n)^{log\, n}$	$n / log\, n$	$f = \Omega(g)$
k	$\sqrt{n}$	$(log\, n)^3$	$f = \Omega(g)$
l	$n^{1/2}$	$5^{log_2 n}$	$f = O(g)$
m	$n2^n$	3^n	$f = O(g)$
n	2^n	2^{n+1}	$f = \Theta(g)$

o	$n!$	2^n	$f = \Omega(g)$
p	$(log\ n)^{log\ n}$	$2^{(log_2 n)^2}$	$f = O(g)$
q	$\sum_{i=1}^{n} i^k$	n^{k+1}	If k = 0, $f = \Theta(g)$ If k < 0, $f = O(g)$ If k > 1, $f = \Omega(g)$ If k = 1, $f = \Theta(g)$ See working below

- If k = 0,

  $f(n) = 1^0 + 2^0 + 3^0 + \cdots + n^0 = 1 + 1 + 1 + \ldots + 1$, total n terms

  $f(n) = n$

  $g(n) = n^{0+1} = n^1 = n$

  Hence $f = \Theta(g)$
- If k < 0,

  $f(n) = 1 + \frac{1}{2^k} + \frac{1}{3^k} + \cdots + \frac{1}{n^k}$, the sum converges towards 1

  $g(n) = \frac{1}{n^{k+1}}$

  Hence $f = O(g)$
- If k > 1,

  $f(n) = 1 + 2^k + 3^k + \cdots + n^k$

  $g(n) = n^{k+1}$

  Hence $f = \Omega(g)$
- If k = 1,

  $f(n) = 1 + 2^1 + 3^1 + \cdots + n^1$, sum of first n natural numbers

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

  $g(n) = n^2$

  Hence $f = \theta(g)$

0.2 Show that, if c is a positive real number, then

$g(n) = 1 + c + c^2 + \cdots + c^n$ **is:**

a. Θ(1) if c < 1.

Let's assume $c = 0.5$, which is less than 1. The sequence will become:

$$g(n) = 1 + 0.5^1 + 0.5^2 + \cdots + 0.5^n$$

$$g(n) = 1 + 0.5 + 0.25 + 0.0625 + \cdots + 0.5^n$$

As we expand more terms value is decreasing i.e. become closer to 0. So the sequence is bound by the first value. So asymptotically $g(n) = \Theta(1)$

b. Θ(n) if c = 1.

For $c = 1$ the sequence is:

$$g(n) = 1 + 1 + 1^2 + \cdots + 1^n$$

$$g(n) = 1 + 1 + 1 + \cdots + 1, n \text{ terms}$$

$$g(n) = n$$

Hence $g(n) = \Theta(n)$

(c) Θ(c^n) if c > 1.

For $c > 1$, each power of c will be greater than 1. Hence c^n will be the dominating term.

So sum of sequence will be bounded by last term i.e. c^n

Therefore $g(n) = \Theta(c^n)$

0.3. The Fibonacci numbers F_0, F_1, F_2, . . . , are defined by the rule

$$F_0 = 0,\ F_1 = 1,\ F_n = F_{n-1} + F_{n-2}.$$

In this problem we will confirm that this sequence grows exponentially fast and obtain some bounds on its growth.

(a) Use induction to prove that $F_n \geq 2^{0.5n}$ for $n \geq 6$.

Consider the base case first.

$n = 6$

$$F_6 \geq 2^{0.5\times6}$$

$8 \geq 8$, Hence true.

$n = 7$

$$F_7 \geq 2^{0.5\times7}$$

$F_7 \geq 2^{3.5}$

$13 \geq 11.313$, Hence true.

Let's say $k \geq 7$, then replacing n by k our inequality becomes.

$F_k \geq 2^{0.5k}$ ---- 1

Now we have to prove that our inequality holds for $k + 1$ too. Replacing k by $k + 1$ we get:

$F_{k+1} \geq 2^{0.5(k+1)}$ ---- 2

Similarly for $k - 1$,

$F_{k-1} \geq 2^{0.5(k-1)}$ ---- 3

Adding inequality 1 and 3 will yield

$F_{k+1} = F_k + F_{k-1} \geq 2^{0.5k} + 2^{0.5(k-1)}$

$F_{k+1} \geq 2^{0.5k} + 2^{0.5(k-1)}$

$F_{k+1} \geq 2^{0.5k} + 2^{0.5k-0.5}$

$F_{k+1} \geq 2^{0.5k} + \frac{2^{0.5k}}{2^{0.5}}$

$F_{k+1} \geq 2^{0.5k}\left(1 + \frac{1}{2^{0.5}}\right)$

The term $\left(1 + \frac{1}{2^{0.5}}\right) = 1.707106781$. Which is approximately equal to

$2^{0.5} = 1.41421356$.

$F_{k+1} \geq 2^{0.5k} \times 2^{0.5}$

$F_{k+1} \geq 2^{0.5k+0.5}$

$F_{k+1} \geq 2^{0.5(k+1)}$

Hence proved.

c. Find a constant $c < 1$ such that $F_n \leq 2^{cn}$ for all $n \geq 0$. Show that your answer is correct

$F_n \leq 2^{cn}$ --- 1

For finding the value of c replace n by $n - 1$ in above inequality we get:

$F_{n-1} \leq 2^{c(n-1)}$

Similarly replacing n by $n + 1$ we get:

$$F_{n+1} \le 2^{c(n+1)}$$

The original inequality $F_n \le 2^{cn}$ will become:

$$F_{n-1} + F_{n-2} \le 2^{cn-c} + 2^{cn-2c}$$

$$F_n \le 2^{cn-c} + 2^{cn-2c}$$

$$F_n \le \frac{2^{cn}}{2^c} + \frac{2^{cn}}{2^{2c}}$$

$$F_n \le 2^{cn}\left(\frac{1}{2^c} + \frac{1}{2^{2c}}\right)$$

For initial condition (inequality 1) to hold the expression $\left(\frac{1}{2^c} + \frac{1}{2^{2c}}\right)$ must be equal to 1. So that when multiplied with 2^{cn}, the result remains the same.

Solving for $\left(\frac{1}{2^c} + \frac{1}{2^{2c}}\right) = 1$

$$\frac{2^{2c}+2^c}{2^{3c}} = 1$$

$$2^{2c} + 2^c = 2^{3c}$$

Taking 2^c common on both sides

$$2^c(2^c + 1) = 2^c(2^{2c})$$

$$2^{2c} = 2^c + 1$$

$$2^{2c} - 2^c - 1 = 0$$

$$(2^c)^2 + (-1)(2^c) + (-1) = 0$$

Let $x = 2^c$

$$x^2 + (-1)(x) + (-1) = 0$$

Hence we have a quadratic equation of the form $ax^2 + bx + c = 0$, such that

$$a = 1,\ b = -1,\ c = -1$$

$$x = \frac{-(-1) \pm \sqrt{(-1)^2 - 4(1)(-1)}}{2(1)}$$

$$x = \frac{1 \pm \sqrt{5}}{2}$$

$$x = \frac{1 \pm 2.236}{2}$$

$x = \frac{1+2.236}{2}$ $x = 1.618$ Substituting value of x $2^c = 1.618$ Taking log on both sides $log_2(2^c) = log_2(1.618)$ $c = log_2(1.618)$ $c = 0.694$	$x = \frac{1-2.236}{2}$ $x =- 0.618$ Substituting value of x $2^c =- 0.618$ Since log on -ve values are not defined. So we can not conclude the value of c from this path

So value of c is defined as : $0.694 \ <= \ c \ < \ 1$

We can rewrite 1 as: $F_n \leq 2^{0.694n} --- 2$

To prove that our approximation is correct we will use induction.
Consider the bases cases:

For $n = 0$, $F_0 \leq 2^0 \ \Rightarrow \ 0 \leq 1$ Hence true

For $n = 1$, $F_1 \leq \ 2^{0.694} \Rightarrow 1 \leq 1.6$ Hence true

For $n = 2$, $F_2 \leq 2^{0.694 * 2} \ \Rightarrow \ 1 \leq 2.617$

Let's assume $k \geq 2$ we can rewrite 2 as:

$$F_k \leq 2^{0.694k} --- 3$$

Now we need to prove for $k + 1$, we will show that $F_{k+1} = 2^{c(k+1)}$

$$F_{k+1} = F_k + F_{k-1}$$

$F_k \leq 2^{0.694k}$ and $F_{k-1} \leq 2^{0.694(k-1)}$

$$F_{k+1} \leq 2^{0.694k} + 2^{0.694k-0.694}$$

$$F_{k+1} \leq 2^{ck}\left(1 + 2^{-c}\right)$$

Since $c = 0.694$, $\left(1 + 2^{-c}\right) = 1.618 = 2^c = 2^{0.694}$

$$F_{k+1} \leq 2^{0.694k} \times 2^{0.694}$$

$$F_{k+1} \leq 2^{ck} \times 2^{c}$$
$$F_{k+1} \leq 2^{c(k+1)}$$
Hence proved.

0.4. Is there a faster way to compute the nth Fibonacci number than by fib2 (page 13)? One idea involves matrices. We start by writing the equations $F_1 = F_1$ and $F_2 = F_0 + F_1$ in matrix notation:

$$\begin{pmatrix} F_1 \\ F_2 \end{pmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} \cdot \begin{pmatrix} F_0 \\ F_1 \end{pmatrix}$$

Similarly,

$$\begin{pmatrix} F_2 \\ F_3 \end{pmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} \cdot \begin{pmatrix} F_1 \\ F_2 \end{pmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}^2 \cdot \begin{pmatrix} F_0 \\ F_1 \end{pmatrix}$$

and in general

$$\begin{pmatrix} F_n \\ F_{n+1} \end{pmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}^2 \cdot \begin{pmatrix} F_0 \\ F_1 \end{pmatrix}$$

So, in order to compute F_n, it suffices to raise this 2 × 2 matrix, call it X, to the nth power.

(a) Show that two 2 × 2 matrices can be multiplied using 4 additions and 8 multiplications.

Consider the following matrices:

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \text{and } B = \begin{bmatrix} e & f \\ g & h \end{bmatrix}$$
$$X = A \cdot B = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \cdot \begin{bmatrix} e & f \\ g & h \end{bmatrix}$$
$$X = \begin{bmatrix} ae + bg & af + bh \\ ce + dg & cf + dh \end{bmatrix}$$

8 multiplications that are required are : $ae,\ bg,\ af,\ bh,\ ce,\ dg, cf$ and dh.
4 additions that are required are : $ae + bg,\ af + bh,\ ce + dg$ and $cf + dh$

b. Show that $O(log\ n)$matrix multiplications suffice for computing X^n. (Hint: Think about computing X^8.)

Since we know that matrix multiplication is not commutative i.e. $AB \neq BA$ but it is associative i.e. $ABC = (AB)C = A(BC)$.
Let's start with the hint

$$X^8 = X \cdot X \cdot X \cdot X \cdot X \cdot X \cdot X \cdot X$$

To calculate this apparently we need 8 matrix multiplications but considering associativity we can avoid some recalculations.

$X^8 = X^4 \cdot X^4$ ---- 1 multiplication will be done here

To compute X^4

$X^4 = X^2 \cdot X^2$ ---- 1 multiplication will be done here

Similarly to computer X^2

$X^2 = X \cdot X$ ---- 1 multiplication will be done here

So to calculate X^8, we performed at most 3 multiplications. Hence X^8 can be computed in at most $O(log\ n) = O(log_2 8) = 3$ matrix multiplications.

Similarly we can use halving technique to calculate X^n using at most $O(log\ n)$ matrix multiplications.