§ 2 Algorithms

Question: Given a list of integers, say 6, 5, 3, 1, 8, 7, 2, 4 (or n integers in general)

How do we sort them in ascending order?

An algorithm is a finite sequence of precise instructions for performing a computation or for solving a problem.

Sorting Algorithms

1) Bubble Sort

procedure *bubblesort*($a_1, \ldots, a_n$: real numbers with $n \geq 2$)
for $i := 1$ **to** $n - 1$
  for $j := 1$ **to** $n - i$
    if $a_j > a_{j+1}$ **then** interchange a_j and a_{j+1}
$\{a_1, \ldots, a_n$ is in increasing order$\}$

2) Insertion Sort

procedure *insertion sort*($a_1, a_2, \ldots, a_n$: real numbers with $n \geq 2$)
for $j := 2$ **to** n
  $i := 1$
  while $a_j > a_i$
    $i := i + 1$
  $m := a_j$
  for $k := 0$ **to** $j - i - 1$
    $a_{j-k} := a_{j-k-1}$
  $a_i := m$
$\{a_1, \ldots, a_n$ is in increasing order$\}$

3) Merge Sort

procedure *mergesort*($L = a_1, \ldots, a_n$)
if $n > 1$ **then**
  $m := \lfloor n/2 \rfloor$
  $L_1 := a_1, a_2, \ldots, a_m$
  $L_2 := a_{m+1}, a_{m+2}, \ldots, a_n$
  $L := merge(mergesort(L_1),\ mergesort(L_2))$
$\{L$ is now sorted into elements in nondecreasing order$\}$

procedure *merge*(L_1, L_2 : sorted lists)
$L :=$ empty list
while L_1 and L_2 are both nonempty
  remove smaller of first elements of L_1 and L_2 from its list; put it at the right end of L
  if this removal makes one list empty **then** remove all elements from the other list and
   append them to L
return $L\{L$ is the merged list with elements in increasing order$\}$

Question: Which one is better?

Approximation of time required (with respect to the size n)

The Growth of Functions

Definition 2.1

Let $f, g : \mathbb{Z}^+ \longrightarrow \mathbb{R}$ (or $f, g : \mathbb{R} \longrightarrow \mathbb{R}$) be functions.

We say $f(n)$ is $O(g(n))$ (or simply write $f(n) = O(g(n))$)

if there exist $C > 0$ and $K \in \mathbb{Z}^+$ such that $|f(n)| \leq C|g(n)|$ for all $n \geq K$.

Idea: $f(n)$ grows slower than or at the same rate as $g(n)$ as n grows (probably with a constant C).

Give an upper bound of the growth of $f(n)$

Example 2.1

Let $f(n) = 2n^2 + 3n + 5$. Show that $f(n) = O(n^2)$.

Note: When $n \geq 3$, we have $|3n| \leq n^2$ and $5 \leq 9 \leq n^2$, so

$$\begin{aligned} |f(n)| &= |2n^2 + 3n + 5| \\ &\leq 2|n^2| + |3n| + |5| \quad \text{(Triangle inequality)} \\ &\leq 2n^2 + n^2 + n^2 \\ &= 4n^2 \end{aligned}$$

$\therefore$ $|f(n)| \leq 4|n^2|$ for all $n \geq 3$ and so $f(n) = O(n^2)$.

On the other hand, note that $|n^2| \leq |f(n)|$ for all $n \in \mathbb{Z}^+$, we have $n^2 = O(f(n))$

Exercise 2.1

Let $f(n)$ be a polynomial of degree d. Show that $f(n) = O(n^d)$.

Exercise 2.2

Suppose that $f(n) = O(g(n))$ and $g(n) = O(h(n))$. Show that $f(n) = O(h(n))$.

Recall that:

A sequence of real numbers may be defined as a function $f : \mathbb{Z}^+ \rightarrow \mathbb{R}$ by setting $a_n = f(n)$.

Also, $\lim_{n \to \infty} a_n = L$ if (definition)

for all $\varepsilon > 0$, there exists $K \in \mathbb{Z}^+$ such that for all $n \geq K$, we have $|a_n - L| < \varepsilon$

Theorem 2.1

If $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$, then $f(n) = O(g(n))$.

proof:

By definition of $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$,

for all $\varepsilon > 0$, there exists K such that for all $n \geq K$, we have $\left|\frac{f(n)}{g(n)} - 0\right| < \varepsilon$.

In particular, take $\varepsilon = 1$, there exists K such that for all $n \geq K$,

we have $\left|\frac{f(n)}{g(n)}\right| < 1$ i.e. $|f(n)| \leq |g(n)|$ and so $f(n) = O(g(n))$

Direct consequence:

If $\alpha > \beta$, then $\lim_{n\to\infty} \frac{n^\beta}{n^\alpha} = \lim_{n\to\infty} \frac{1}{n^{\alpha-\beta}} = 0$ and so $n^\beta = O(n^\alpha)$.

However, n^α is not $O(n^\beta)$.

Prove by contradiction, suppose that $n^\alpha = O(n^\beta)$.

Then, there exist $C > 0$ and $K \in \mathbb{Z}^+$ such that $|n^\alpha| \leq C|n^\beta|$ for all $n \geq K$. (*)

Since $\alpha > \beta$, i.e $\alpha - \beta > 0$, there exists $N \in \mathbb{Z}^+$ such that $N^{\alpha-\beta} > C$.

Consider $n_0 = \max\{K, N\} \geq K$, we have $n_0^{\alpha-\beta} \geq N^{\alpha-\beta} > C$

so $|n_0^\alpha| > C|n_0^\beta|$ (Contradict to (*))

Example 2.2

Let $f(n) = \log n$ (Here, $\log n$ means $\ln n$), $g(n) = n^d$ where $d > 0$.

How to compute $\lim_{n\to\infty} \frac{f(n)}{g(n)}$?

Trick: $\lim_{x\to\infty} \frac{\log x}{x^d}$ $\left(\frac{\infty}{\infty}\right)$

$$= \lim_{x\to\infty} \frac{\frac{1}{x}}{dx^{d-1}}$$

$$= \lim_{x\to\infty} \frac{1}{dx}$$

$$= 0$$

$\therefore \lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$ and we have $\log n = O(n^d)$ for all $d > 0$.

Exercise 2.3

Let $d > 0$, $\alpha > 1$. Show that $n^d = O(\alpha^n)$.

Example 2.3

Let $\alpha > 0$. Show that $\alpha^n = O(n!)$

(It is not surprising if $0<\alpha\le 1$. When $0<\alpha<1$, $\lim_{n\to\infty}\alpha^n = 0$; When $\alpha=1$, $\lim_{n\to\infty}\alpha^n = 1$.

Therefore the case $\alpha>1$ is what we really interest.)

Claim: $\lim_{n\to\infty}\frac{\alpha^n}{n!} = 0$

Choose $K\in\mathbb{Z}^+$ such that $\alpha<K$ (i.e. $\frac{\alpha}{K}<1$)

When $n\ge K$

$$0 \le \frac{\alpha^n}{n!} = \underbrace{\left(\frac{\alpha}{1}\cdot\frac{\alpha}{2}\cdot\frac{\alpha}{3}\cdots\frac{\alpha}{K-1}\right)}_{\le M}\cdot\left(\frac{\alpha}{K}\,\frac{\alpha}{K-1}\cdots\frac{\alpha}{n}\right)$$

$$\le M\left(\frac{\alpha}{K}\right)^{n-K+1}$$

$$\frac{\alpha}{K}<1 \Rightarrow \lim_{n\to\infty}\left(\frac{\alpha}{K}\right)^{n-K+1} = 0 \Rightarrow \lim_{n\to\infty} M\left(\frac{\alpha}{K}\right)^{n-K+1} = 0$$

$\therefore$ By the sandwich theorem, $\lim_{n\to\infty}\frac{\alpha^n}{n!} = 0$ and so $\alpha^n = O(n!)$

Example 2.4

Show that $n! = O(n^n)$

Obvious! $n! = 1\cdot2\cdot3\cdot\dots\cdot n \le n\cdot n\cdot\dots\cdot n = n^n$ for all $n\in\mathbb{Z}^+$

(i.e. $|n!| \le |n^n|$ for all $n\ge K=1$)

Furthermore, for all $n\in\mathbb{Z}^+$, $n! \le n^n$

$$\log n! \le \log n^n = n\log n$$

$\therefore$ $\log n! = O(n\log n)$

Summary:

Growth of functions in ascending order:

$\log n$, n^d, α^n, $n!$, n^n , where $d>0$, $\alpha>1$.

The Growth of Combinations of Functions

Theorem 2.2

If $f_1(n) = O(g_1(n))$ and $f_2(n) = O(g_2(n))$, then $(f_1+f_2)(n) = O(g(n))$ where $g(n) = \max\{|g_1(n)|, |g_2(n)|\}$ for all $n \in \mathbb{Z}^+$.

proof:

By assumption, there exist $C_1, C_2 > 0$ and $K_1, K_2 \in \mathbb{Z}^+$ such that $|f_1(n)| \geq C_1|g_1(n)|$ for all $n \geq K_1$, $|f_2(n)| \geq C_2|g_2(n)|$ for all $n \geq K_2$.

Take $K = \max\{K_1, K_2\} \geq K_1, K_2$. Then, for all $n \geq K$,

$$\begin{aligned}
|(f_1+f_2)(n)| &= |f_1(n) + f_2(n)| \\
&\leq |f_1(n)| + |f_2(n)| \\
&\leq C_1|g_1(n)| + C_2|g_2(n)| \\
&\leq C_1|g(n)| + C_2|g(n)| \\
&= (C_1+C_2)|g(n)|
\end{aligned}$$

Example 2.5

Note that $2n^2 = O(n^2)$, $3n = O(n)$, and $\max\{|n^2|, |n|\} = n^2$,

so $2n^2+3n = O(n^2)$

Corollary 2.1

If $f_i(n) = O(g_i(n))$ for $i = 1, 2, \cdots, m$, then $(\sum_{i=1}^{m} f_i)(n) = O(g(n))$ where $g(n) = \max_{1\leq i\leq m}\{|g_i(n)|\}$.

(By mathematical induction)

Theorem 2.2

If $f_1(n) = O(g_1(n))$ and $f_2(n) = O(g_2(n))$, then $(f_1 \cdot f_2)(n) = O((g_1 \cdot g_2)(n))$.

Example 2.6

Give big-O estimate for $f(n) = 3n\log(n!) + (n^2+3)\log n$

$3n = O(n)$, $\log(n!) = O(n\log n)$, $n^2+3 = O(n^2)$,

so $f(n) = O(n^2 \log n)$

Besides giving an upper bound of the growth of a function, sometimes we would like to have a lower bound.

Definition 2.2

Let $f, g : \mathbb{Z}^+ \longrightarrow \mathbb{R}$ (or $f, g : \mathbb{R} \longrightarrow \mathbb{R}$) be functions.

We say $f(n)$ is $\Omega(g(n))$ (or simply write $f(n) = O(g(n))$)

if there exist $C > 0$ and $K \in \mathbb{Z}^+$ such that $|f(n)| \leq C|g(n)|$ for all $n \geq K$.

$|f(n)| \geq C|g(n)| \Leftrightarrow |g(n)| \leq \frac{1}{C}|f(n)|$

$\therefore$ $f(n) = \Omega(g(n)) \Leftrightarrow g(n) = O(f(n))$.

Exercise 2.4

Suppose that $f(n) = \Omega(g(n))$ and $g(n) = \Omega(h(n))$. Show that $f(n) = \Omega(h(n))$. (Use exercise 2.2)

Definition 2.3

$f(n) = \Theta(g(n))$ if both $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$.

Exercise 2.5

1) Prove that

a) $f(n) = \Theta(f(n))$ for all $f(n)$

b) If $f(n) = \Theta(g(n))$, then $g(n) = \Theta(f(n))$

c) If $f(n) = \Theta(g(n))$ and $g(n) = \Theta(h(n))$, then $f(n) = \Theta(h(n))$.

Let $S = \{f : \mathbb{Z}^+ \to \mathbb{R}\}$ and define a relation $\sim$ on S by $f \sim g$ if $f(n) = O(g(n))$.

Then, (1) shows that $\sim$ is an equivalent relation.

2) If $f_1(n) = \Theta(f_2(n))$ and $g_1(n) = \Theta(g_2(n))$, then $(f_1 \cdot f_2)(n) = \Theta((g_1 \cdot g_2)(n))$

(2) shows that the product on S induces a product on $S/\sim$.

But note that if $f_1(n) = n^2 + n$, $f_2(n) = -n^2 + n$, then f_1 and f_2 are $\Theta(n^2)$

but $(f_1 + f_2)(n) = 2n = \Theta(n)$ (not $\Theta(2n^2)$)

Time Complexity of Algorithms

Comparing the performance of algorithms in terms of time.

For example, we try to compare the following sorting algorithms.

(Assume the performance depends on the number of comparsions)

1) Bubble Sort

procedure *bubblesort*($a_1, \ldots, a_n$: real numbers with $n \geq 2$)
for $i := 1$ **to** $n - 1$
  for $j := 1$ **to** $n - i$
    if $a_j > a_{j+1}$ **then** interchange a_j and a_{j+1}
$\{a_1, \ldots, a_n$ is in increasing order$\}$

perform n-i comparsions

$$\text{Number of comparsions} = \sum_{i=1}^{n}(n-i) = n^2 - \sum_{i=1}^{n} i = n^2 - \frac{n(n+1)}{2} = \frac{n(n-1)}{2}$$

which is $\Theta(n^2)$

2) Insertion Sort

procedure *insertion sort*($a_1, a_2, \ldots, a_n$: real numbers with $n \geq 2$)
for $j := 2$ **to** n
  $i := 1$
  while $a_j > a_i$
    $i := i + 1$
  $m := a_j$
  for $k := 0$ **to** $j - i - 1$
    $a_{j-k} := a_{j-k-1}$
  $a_i := m$
$\{a_1, \ldots, a_n$ is in increasing order$\}$

perform at most j comparsions

$$\text{Number of comparsions} \leq \sum_{j=2}^{n} j = \frac{n(n+1)}{2} - 1$$

which is $\Theta(n^2)$

Worst-case complexity of bubble sort and insertion sort are $\Theta(n^2)$.

Worst-case complexity of merge sort is $\Theta(n \log n)$

(see section 5.4 and 8.3 of [1])