Lecture1: Asymptotic Notations, Euclid's Algorithm

Scriber: Xingyu Chen

September 2017

1 Asymptotic Notation

The Asymptotic Notation is to roughly measure the running time or memory for algorithm. So we will only keep the most weighted term. E.g. In Asymptotic Notation, $3n^2 + 2n \approx n^2 \not\approx 2^n$.

1.1 Definition

Definition 1. *$f(n) = O(g(n))$, if there is constants $C > 0$, $n_0 > 0$ such that for all $n \geq n_0$, $f(n) \leq C \cdot g(n)$*

Definition 2. *$f(n) = \Omega(g(n))$, if there is constants $C > 0$, $n_0 > 0$ such that for all $n \geq n_0$, $f(n) \geq C \cdot g(n)$*

Definition 3. *$f(n) = \Theta(g(n))$, if $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$*

Definition 4. *$f(n) = o(g(n))$, if $g(n) \neq O(f(n))$. In other words, f(n) becomes insignificant relative to g(n) as n approaches infinity, $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$*

Definition 5. *$f(n) = \omega(g(n))$, if $f(n) \neq O(g(n))$. In other words, $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$*

1.2 Example

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

Proof.

$$3n^2 + 2n \leq 3n^2 + 2n^2 = 5n^2$$

In the definition, we can choose $C = 5$, $n_0 = 1$. From there we obtain

$$3n^2 + 2n \leq C \cdot n^2$$

So $3n^2 + 2n = O(n^2)$ $\square$

(b) $n^2 \neq O(n)$, $(n^2 = \omega(n))$

Proof. Use contradiction to prove, assume $n^2 = O(n)$, there is $C > 0$, $n_0 > 0$ s.t. when $n \geq n_0$, $n^2 \leq C \cdot n$.
Pick $n > \max\{n_0, C\}$. Then we have $n^2 = n \cdot n > C \cdot n$. Contradiction! Therefore $n^2 \neq O(n)$ $\square$

There's a useful inequality useful in asymptotic notation

Remark 1.

$$\log n < \sqrt{n} < n < n\log n < n^2 < 2^n < 3^n < n!$$

2 The Importance of Asymptotic Notation

For this section, let's consider a well-known sorting algorithm Bubble Sort to take a close look at the usage and importance of asymptotic notation.

2.1 Algorithm

Bubble Sort is a sort algorithm which compares each element to its neighbor and swap if not in order. The pseudocode is as below:

```
for i = n downto 1
    for j = 1 to i-1
        if a[j]>a[j+1] then swap
```

2.2 Analysis of running time

Each round, the inner loop will run $i-1$ times in each outer loop round while i goes from n to 1. So

$$T = (n-1) + (n-2) + ... + 1 = \frac{n(n-1)}{2}$$

2.3 Analysis of running time II

Now we consider a problem that if there is an algorithm that calls Bubble Sort on an array of size $1, 2, 3..., n$. What can we say about the running time?
Of course there are still ways to calculate the exact expression for running times as below:

$$\begin{aligned}
T &= \frac{1\cdot 0}{2} + \frac{2\cdot 1}{2} + \frac{3\cdot 2}{2} + ... + \frac{n\cdot(n-1)}{2} \\
&= \frac{2\cdot 1\cdot 0 - 1\cdot 0\cdot -1}{6} + \frac{3\cdot 2\cdot 1 - 2\cdot 1\cdot 0}{6} + \frac{4\cdot 3\cdot 2 - 3\cdot 2\cdot 1}{6} + ... + \frac{(n+1)\cdot n\cdot(n-1) - n\cdot(n-1)\cdot(n-2)}{6} \\
&= \frac{(n+1)n(n-1)}{6}
\end{aligned}$$

However, there's not always such a clever way to obtain an accurate polynomial. By using asymptotic notation, we can obtain a bound for running time more easily and for more occasions.

Claim 1. $T = \Theta(n^3)$

Proof.

$$T = \sum_{i=1}^{n} \frac{i(i-1)}{2} \leq n \cdot \frac{n(n-1)}{2} = O(n^3)$$

On the other hand,

$$T = \sum_{i=1}^{n} \frac{i(i-1)}{2} > \sum_{i=\frac{n}{2}+1}^{n} \frac{i(i-1)}{2} \geq \frac{n}{2} \cdot \frac{(\frac{n}{2})^2}{2} = \frac{n^3}{16} = \Omega(n^3)$$

In conclusion, $T(n) = \Theta(n^3)$ □

3 Euclid's Algorithm

The Euclid's Algorithm aims to compute greatest common divisor (g.c.d) of 2 integers. For example,

$$gcd(15,9) = 3$$

3.1 Algorithm

Algorithm 1 Euclid's Algorithm

```
if b == 0 then
    return a
else
    return gcd(b, a mod b)
```

Example Run:

$$gcd(15,9) \rightarrow gcd(9,6) \rightarrow gcd(6,3) \rightarrow gcd(3,0) \rightarrow 3$$

3.2 Proof of Correctness

We use induction to prove.
Base Case:
if $b = 0$, $gcd(a,0) = a$. a is indeed the greatest common divisor of a and 0. Base case is correct.

Induction:
We want to prove the following claim.

Claim 2.

$$gcd(a,b) = gcd(b, a\ mod\ b)$$

Proof. Assume $gcd(b, a \bmod b)$ is the greatest common divisor of b and $(a \bmod b)$.
1. If $c|a$, $c|b$, then

$$\frac{a \bmod b}{c} = \frac{a - k \cdot b}{c} = \frac{a}{c} - k \cdot \frac{b}{c}$$

Since $c|a$ and $c|b$, then k, $\frac{a}{c}$ and $\frac{b}{c}$ are all integers. So $\frac{a \bmod b}{c}$ must be an integer as well. In other words, $c|(a \bmod b)$.
2. If $c|a$, $c|a \bmod b$, then

$$\frac{a}{c} = \frac{(a - k \cdot b) + k \cdot b}{c} = \frac{a - k \cdot b}{c} + k \cdot \frac{b}{c}$$

Since $c|a$ and $c|a \bmod b$, then k, $\frac{a-kb}{c}$ and $\frac{b}{c}$ are all integers. So $\frac{a}{c}$ must be an integer as well. In other words, $c|a$.
From the above two parts, we know that for any arbitrary integer c, if it divides a and b, it divides $(a \bmod b)$. If it divides b and $a \bmod b$, it divides a as well. So the set of common divisors for (a,b) and $(b, a \bmod b)$ are the same.
By induction hypothesis, $gcd(b, a \bmod b)$ is correct(i.e. $gcd(b, a \bmod b)$ is the greatest common divisor of b and $(a \bmod b)$). Since the set of common divisors are the same for (a,b) pair and $(b, a \bmod b)$ pair, $gcd(a,b)$ must as well be the greatest common divisor of a and b. $\square$