Chapter 2

Algorithm Complexity Analysis

Profilers

← tool

→ Comparing 2 algorithms in idea level

→ How my algorithm behaves when the input grows

Counting Instruction

```
int max = mArray[0];                     → 2
for (int i=0 ; i<n ; i++) {              → 3   | 1+n+n |
    if (mArray[i] > max) {               → 2
    max = mArray[i];
}
```

$\boxed{O(n)}$

1. Assignment
2. Look up into array
3. Comparison
4. Incrementation
5. Arithmatic op

$$2 + 1 + n + n + 2$$
$$= 5 + 2n$$

Cost function, $T(n) = \boxed{5} + 2n$

Constants
constant multipliers $\Big\} \to X$

Asymptotic behavior

$f(n) = n$

1. $T(n) = 3n + 60$

$f(n) = n$

2. $T(n) = 5n^2 + 20n$

$f(n) = n^2$

3. $T(n) = 3033$

$f(n) = 1$

$$\boxed{2^x = 1024}$$

$2^{10} = 1024$

$\log_2(1024) = 10$

Binary Search

n data

0^{th} iteration : $n/2^0$

1^{st} " : $n/2^1$

2^{nd} " : $n/4$ 2^2

$\vdots$ $\vdots$

i^{th} " : $n/2^i$

$\hookrightarrow$ last iteration

1 data

$$1 = \frac{n}{2^i}$$

$$\Rightarrow 2^i = n$$

$$\Rightarrow i = \boxed{\log_2(n)}$$

$\log_2(32) = 5$

$$\begin{array}{cccccc} 32 & \rightarrow 16 & \rightarrow 8 & \rightarrow 4 & \rightarrow 2 & \rightarrow 1 \\ 0 & 1 & 2 & 3 & 4 & 5 \end{array}$$

Tight Upper Bound

$\boxed{\text{Big - Oh} \quad O}$

Upper Bound (not - tight)

Small - oh

o

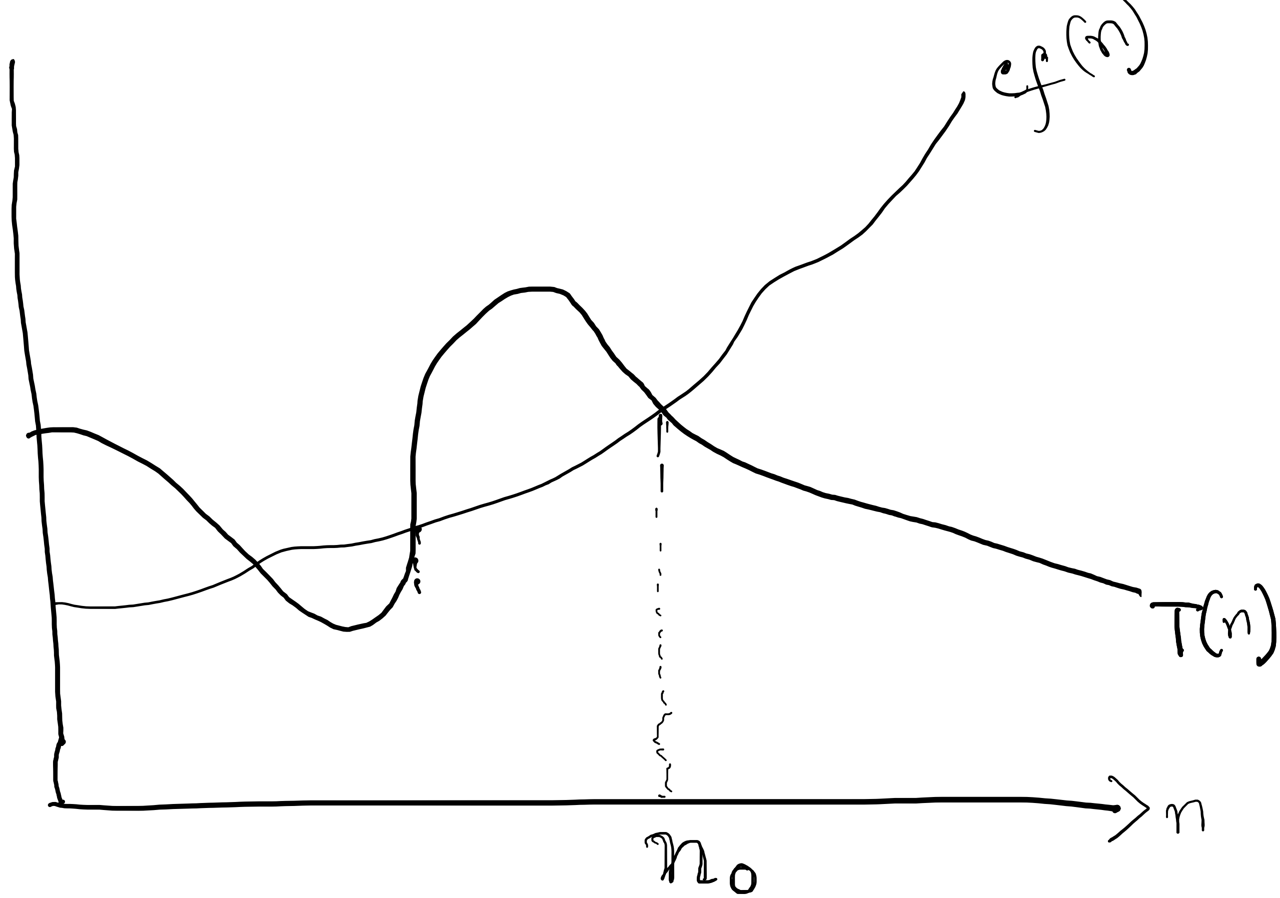
Big-oh : [upper bound, tight]

$T(n) = O(f(n))$ if

there are positive constants

c and n_0 such that

$T(n) \leq cf(n)$ where $n \geq n_0$



Example:

Cost function, $T(n) = 3n + 12$

$f(n) = n$

If we can find constants n_0 and c such that

$$T(n) \le c * f(n)$$

$$\Rightarrow 3n + 12 \le c * n$$

$\boxed{c = 4}$, $n_0 = 12$

$3n + 12 \le 4 * n$ for all $n \geq 12$

Example 2.

$$T(n) = 2n^2 + 10n + 6$$

$$f(n) = n^2$$

$c = 3$, $n_0 = 11$

$3n^2 \geq 2n^2 + 10n + 6$ for all $n \geq 11$

$$O(1) < O(\log n) < O(n) < O(n \log n) < O(n^2) < O(n^3) \ldots < O(n^k) < O(k^n)$$