Asymptotic Analysis

High-level idea: supress constant factors and lower order terms

example: merge-sort $6N \cdot \log_2 N + 6N$ simplify to $N \cdot \log_2 N$

Terminology: Running time is $O(N \cdot \log_2 N)$

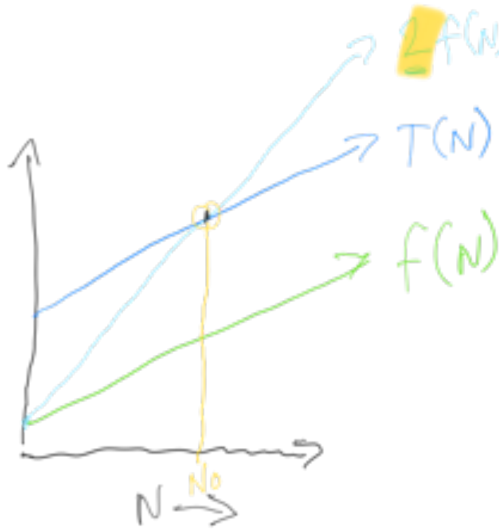
N = input size (e.g. array length)

English definition

Big Oh Let $T(N)$ = function on $N = 1, 2, 3, \ldots$

When $T(N) = O(f(N))$?

If for all sufficiently large values of N, $T(N)$ is bounded above by a constant multiple of $f(N)$



$T(N) = O(f(N))$

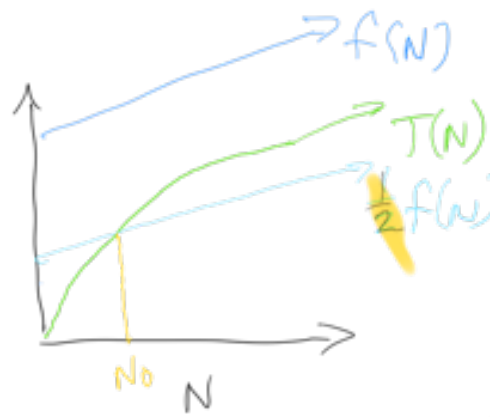
Formal definition

$T(N) = O(f(N))$
if and only if there exists constants c, $N_0 > 0$.
such that
$T(N) \leq c \cdot f(N)$
for all $N \geq N_0$

warning c, N_0 cannot depend on N.

Omega Notation

Definition: $T(N) = \Omega(f(N))$
if and only if $\exists$ constants
c, N_0 such that
$T(N) \geq c \cdot f(N)$
for all $N \geq N_0$



Theta Notation

Definition: $T(N) = \Theta(f(N))$ if and only if
$T(N) = O(f(N))$ and $T(N) = \Omega(f(N))$
Equivalently $\exists$ constants c_1, c_2 and N_0 such that
$c_1 f(N) \leq T(N) \leq c_2 f(N)$
for all $N \geq N_0$

Claim: $2^{N+10} = O(2^N)$

proof: We need to pick 2 constants c, N_0 such that the inequality holds $\forall N \geq N_0$

$$2^{N+10} \leq c\, 2^N$$

$$2^{N+10} = 2^{10} \cdot 2^N = 1024 \cdot 2^N$$

so if choose $c = 1024$, $N_0 = 1$ then inequality holds $\forall N \geq N_0$. QED

Claim: 2^{10N} is <u>NOT</u> $O(2^N)$

proof by contradiction. If $2^{10N} = O(2^N)$ then $\exists$ constants $c, N_0 > 0$ such that

$$2^{10N} \leq c \cdot 2^N \qquad \forall N \geq N_0$$

But then (cancelling 2^N):

$$2^{9N} \leq c \qquad \forall N \geq N_0$$

But this inequality is false since c is a fixed constant and N can go to ∞. Therefore 2^{9N} will surpass C. QED