Chapter 2: Algorithm Analysis

Yangtao Ge

June 19, 2019

Abstract

This section discusses about:

- how to estimate the time required for a program
- how to reduce running time of a program
- The result of careless use of recursion
- very efficient algorithms to raise a number to a power
- compute GCD

1 Mathematical Background

Four definitions of the framework:

Definition 1.1 (Upper bound)**.** $T(N) = O(f(N))$ if there are positive *constants* c and n_0 such that $T(N) \leq cf(N)$ when $N \geq n_0$

Definition 1.2 (Lower bound)**.** $T(N) = \Omega(g(N))$ if there are positive *constants* c and n_0 such that $T(N) \geq cg(N)$ when $N \geq n_0$

Definition 1.3 (Envelope)**.** $T(N) = \Theta(h(N))$ iff $T(N) - O(h(N))$ and $T(N) = \Omega(h(N))$

Definition 1.4. $T(N) = o(p(N))$ if for ***all*** positive constants c there exists an n_0 such that $T(N) < cp(N)$ when $N > n_0$.

Ref: p.30 for detail theorem

Typical growth rates:

Function	Name
c	Constant
$\log N$	Logarithmic
$\log^2 N$	Log-Squared
N	Linear
$N \log N$	
N^2	Quadratic
N^3	Cubic
2^N	Exponential

Some Theorem from the definition:

Theorem 1.1. If $T_1(N) = O(f(N))$ and $T_2(N) = O(g(N))$, then:

1. $T_1(N) + T_2(N) = O(f(N) + g(N))$
2. $T_1(N) * T_2(N) = O(f(N) * g(N))$

Theorem 1.2. If $T(N)$ is a polynomial of degree k, then $T(N) = \Theta(N^k)$

Theorem 1.3. $\log^k N = O(N)$ for any constant k, which tell us 'Logarithms grow very slowly

For big-O notation answers: Lower-order terms can generally be ignored e.g. $f(N) = 2N^2 + N$ then its big-O notation is $T(N) = O(N^2)$

For *relative growth rates* of two Function, we using '*L'Hopitals's rule*' to determine it:

Theorem 1.4 (L'Hopitals's rule)**.** If $\lim_{N\to\infty} f(N) = \infty$ and $\lim_{N\to\infty} g(N) = \infty$, then $\lim_{N\to\infty} \frac{f(N)}{g(N)} = \lim_{N\to\infty} \frac{f'(N)}{g'(N)}$

Four possible results:

- Limit is 0: $f(N) = 0(g(N))$
- Limit is $c \neq 0$: $f(N) = \Theta(g(N))$
- Limit is ∞: $g(N) = o(f(N))$
- Limit does not exist: No relations

2 Model

Basically a normal computer which execute instructions sequentially. And it has inifinite memory

3 What to Analyze