ALG 1.2
Asymptotics and Recurrence Equations

(a) Computational Complexity of a Program
(b) Worst Case and Expected Bounds
(c) Solution of Recurrence Notation
(d) Definition of Asymptotic Equations

Main Reading Selections:
CLR, Chapter 2, 3, 4
Handout: "Counting and Estimating"

Auxiliary Reading Selections:
AHU-Design, Chapter 2
AHU-Data, Chapter 9
BB-Chapter 2

Asymptotics:

goal
is to estimate and compare
growth rates of functions

ignore
costant factors of growth
"f(n) is asymptotically equal to g(n)"

\[ f(n) = g(n) \quad \text{if} \quad \lim_{n \to \infty} \frac{f(n)}{g(n)} = 1 \]

"f(n) is little-o of g(n)"

f(n) is \( o(g(n)) \) if

\[ \lim_{n \to \infty} \frac{f(n)}{g(n)} = 0 \]
"f(n) is $O(g(n))$"

f(n) is $O(g(n))$ if

$$\lim_{n \to \infty} \frac{|f(n)|}{g(n)} \leq c$$

ie, 

$$\exists \ c, n_0 > 0 \ \text{s.t.} \ f(n) \leq c \cdot g(n)$$

for all $n \geq n_0$
"f(n) is order at least g(n)"

\[ f(n) \text{ is } \Omega(g(n)) \text{ if } \liminf_{n \to \infty} \frac{f(n)}{g(n)} \geq c \]

\[ \exists n_0, c > 0 \text{ s.t. } f(n) \geq c g(n) \text{ for all } n \geq n_0 \]
"f(n) is order tight with g(n)" if

f(n) is $O(g(n))$ and also $\Omega(g(n))$

i.e.

$\exists n_0, c, c' \text{ s.t. } c \cdot g(n) \leq f(n) \leq c' \cdot g(n)$

Suppose my algorithm runs in time $O(n)$

*don't say: "his runs in time $O(n^2)$ so is worse"

*but prove: "his runs in time $\Omega(n^2)$ so is worse"

- must find a worst case input of length $n$ for which his algorithm takes time $\geq cn^2$ for all $n \geq n_0$
**Notation**

\[ n \text{ is } O(n^2) \]

sometimes written

\[ n = O(n^2) \]

but \( n^2 \) is not \( O(n) \)

so can't use identities!

The two sides of the equality do not play a symmetric role

**write**

"\( f(n) - g(n) \) is \( o(h(n)) \)"

**as**

\[ f(n) = g(n) + o(h(n)) \]

**example**

\[
\frac{n}{n-1} = 1 + \frac{1}{n} + O\left(\frac{1}{n^2}\right)
\]

\[
= 1 + \frac{1}{n} + o\left(\frac{1}{n}\right)
\]

\[
= 1 + o(1) \quad \text{as } n \to \infty
\]
Convergent Power Sum
\[ \sum_{i=0}^{\infty} x^i = \frac{1}{1-x} \leq O(1) \]
for \(0 < x < 1\)

A polynomial is asymptotically equal to its leading term as \(x \to \infty\)

\[ \sum_{i=0}^{d} a_i x^i = \Theta \left( x^d \right) \]
\[ \sum_{i=0}^{d} a_i x^i = o \left( x^{d+1} \right) \]
\[ \sum_{i=0}^{d} a_i x^i \sim a_d x^d \]

Sums of Powers:
for \(n \to \infty\)

\[ \sum_{i=1}^{n} i^d \sim \frac{1}{d+1} n^{d+1} \]

or equivalently

\[ \sum_{i=1}^{n} i^d = \frac{1}{d+1} n^{d+1} + o \left( n^{d+1} \right) \]
examples

\[
\sum_{i=1}^{n} i \sim \frac{n^2}{2}
\]
\[
\sum_{i=1}^{n} i^2 \sim \frac{n^3}{3}
\]

2nd order asymptotic expansion

\[
\sum_{i=1}^{n} i^d = \frac{1}{d+1} n^{d+1} + \frac{1}{2} n^d + O\left( n^{d-1} \right)
\]

Asymptotic Expansion of \( f(n) \) as \( n \to n_0 \)

\[
f(n) \sim \sum_{i=1}^{\infty} c_i g_i (n)
\]

if

(1) \( g_{i+1} (n) = o(g_i (n)) \) for all \( i \geq 1 \)

and

(2) \( f(n) = \sum_{i=1}^{k} c_i g_i (n) + o(g_k (n)) \) for all \( k \geq 1 \)
Bounding Sums by Integrals

\[ \sum_{k=1}^{n} f(k) \leq \int_{1}^{n+1} f(x) \, dx \leq \sum_{k=1}^{n} f(k+1) \]

so \[ \int_{1}^{n+1} f(x) \, dx - f(n+1) + f(1) \leq \sum_{k=1}^{n} f(k) \leq \int_{1}^{n+1} f(x) \, dx \]

**example** if \( f(x) = \ln(x) \) then \[ \int_{1}^{n+1} \ln(x) \, dx = x\ln(x) - x \]

so \[ \sum_{k=1}^{n} \ln k = (n+1) \ln(n+1) - n + \Theta(\ln(n)) \]

since \( \log(n) = \frac{\ln n}{\ln 2} \)

so \[ \sum_{k=1}^{n} \log k = (n+1) \log(n+1) - \frac{n}{\ln 2} + \Theta(\log(n)) \]

Other Approximations Derived from Integrals

\[ \sum_{k=1}^{n} k \log k = \frac{(n+1)^2}{2} \log(n+1) - \frac{(n+1)^2}{4 \ln 2} + \Theta(n \log n) \]

Harmonic Numbers

\[ H_n = \sum_{k=1}^{n} \frac{1}{k} \]

\[ H_n = \ln(n) + \gamma + O \left( \frac{1}{n} \right) \]

Euler's constant \( \gamma \approx 0.577 \ldots \)
factorial \( n! = 1 \cdot 2 \cdot 3 \cdots n \)

**Stirling's Approximation for Factorial**

\[
n! \sim \sqrt{2\pi n} \left(\frac{n}{e}\right)^n \text{ as } n \to \infty
\]

So,

\[
\log(n!) = n \log n - n \log e + \frac{1}{2} \log(2\pi n) + \Theta(1) \\
= n \log n - \Theta(n)
\]
**Fibonacci Sequence**

**example:**

\[ F_n = F_{n-1} + F_{n-2} \]

\[ F_0 = 0, \ F_1 = 1 \]

\[ n \geq 2 \]

**Solution of Fibonacci Sequence**

\[ r_1 = \frac{1}{2} \left( 1 + \sqrt{5} \right) = 1.618 \ldots \text{ "golden ratio"} \]

\[ r_2 = \frac{1}{2} \left( 1 - \sqrt{5} \right) \]

\[ F_n = c_1 r_1^n + c_2 r_2^n \]

where \( F_0 = c_1 + c_2 = 0 \)

\[ F_1 = c_1 r_1 + c_2 r_2 = 1 \]

\[ \ldots \]

\[ F_n = \frac{1}{\sqrt{5}} \left[ \left( \frac{1 + \sqrt{5}}{2} \right)^n - \left( \frac{1 - \sqrt{5}}{2} \right)^n \right] \]

\[ \Rightarrow F_n \sim \frac{1}{\sqrt{5}} \left( \frac{1 + \sqrt{5}}{2} \right)^n \]
**Homogeneous Recurrence Relations**

(no constant additive term)

**Solve:** \[ x_n = a_1 x_{n-1} + a_2 x_{n-2} + \ldots + a_d x_{n-d} \]

try \[ x_n = r^n \]

multiply by \[ \frac{r^d}{r^n} \]

get characteristic equation:

\[ r^d - a_1 r^{d-1} - a_2 r^{d-2} - \ldots - a_d = 0 \]

---

**Case**

**Distinct Roots**

\[ r_1 \neq r_2 \neq r_3 \neq \ldots \neq r_d \]

\[ x_n = \sum_{i=1}^{d} c_i r_i^n \]

where \( r_i \) is dominant root \[ |r_i| > |r_j| \ (\forall \ j \neq i) \]

**Case**

Roots are not distinct

\[ r_1 = r_2 = r_3 \]

Then solutions not independent, so additional terms:

\[ x_n = c_1 r_1^n + c_2 n r_1^n + c_3 n^2 r_1^n + \sum_{i=4}^{d} c_i r_i^n \]
Inhomogeneous Recurrence Equation

\[ x_n = a_1 x_{n-1} + a_2 x_{n-2} + \ldots + a_d x_{n-d} + a_0 \]

nonzero constant term  \( a_0 \neq 0 \)

Solution Method

(1) Solve homogeneous equation

\[ Y_n = a_1 Y_{n-1} + a_2 Y_{n-2} + \ldots + a_n Y_{n-d} \]

(2) Case

\[ \Sigma a_i \neq 1, \text{ add particular solution} \]

\[ x_n = c = \frac{a_0}{1 - \Sigma a_i} \]

Case

\[ \Sigma a_i = 1, \text{ add particular solution} \]

\[ x_n = cn = \left( \frac{a_0}{\Sigma ia_i} \right)^n \]

(3) add particular and homogeneous solutions, and solve for constants
This is all we usually need!!

**A Useful Theorem**

\[ c > 0 \quad d > 0, \]

\[
\begin{align*}
\text{if } T(n) &= \begin{cases} 
    c_0 & n = 1 \\
    a T\left( \frac{n}{b} \right) + c n^d & n > 1 
\end{cases} \\
\text{then } T(n) &= \begin{cases} 
    \theta \left( n^{\log_b a} \right) & a > b^d \\
    \theta \left( n^d \log_b n \right) & a = b^d \\
    \theta \left( n^d \right) & a < b^d 
\end{cases}
\end{align*}
\]

Proof

\[
T(n) = c n^d g(n) + a^{\log_b n} d
\]

is solution

\[
g(n) = 1 + \frac{a}{b^d} + \left( \frac{a}{b^d} \right)^2 + \cdots + \left( \frac{a}{b^d} \right)^{\log_b n - 1}
\]

**Cases**

1. \( a > b^d \Rightarrow g(n) \sim \left( \frac{a}{b^d} \right)^{\log_b n - 1} \)
   
   is last term so
   
   \[
   T(n) = \theta \left( n^{\log_b a} d \right) = \theta \left( n^{\log_b a} \right)
   \]

2. \( a = b^d \Rightarrow g(n) = \log_b n \)
   
   so \( T(n) = \theta \left( n^d \log_b n \right) \)

3. \( a < b^d \Rightarrow g(n) \) upper bound by \( O(1) \)
   
   so \( T(n) = \theta \left( n^d \right) \)
Example

mergesort

input list L of length N

if N=1 then return L

else do

let \( L_1 \) be the first \( \left\lfloor \frac{N}{2} \right\rfloor \) elements of L

let \( L_2 \) be the last \( \left\lceil \frac{N}{2} \right\rceil \) elements of L

M_1 \leftarrow \text{Mergesort} (L_1)

M_2 \leftarrow \text{Mergesort} (L_2)

return Merge (M_1 , M_2)

Time Bound

Initial Value \( T(1) = c_1 \)

for \( N>1 \)

\[
T(N) \leq T\left( \frac{N}{2} \right) + T\left( \frac{N}{2} \right) + c_2 N
\]

for some constants \( c_1 , c_2 \geq 1 \)

\( N>1 \)

\[
T(N) \leq 2T\left( \frac{N}{2} \right) + c_2 N
\]

guess \( T(N) \leq a N \log N + b \)

\[
\leq 2 \left( a \frac{N}{2} \log \left( \frac{N}{2} \right) + b \right) + c_2 N
\]

holds if \( a = c_1 + c_2 , \; \; b = c_1 \)

Solution \( T(N) \leq (c_1 + c_2) N \log N + c_1 \)
Transform Variables

\[ \begin{align*}
n &= \log N, \quad N = 2^n \\
n - 1 &= \log N - \log 2 = \log \left( \frac{N}{2} \right)
\end{align*} \]

Recurrence equation:

\[ X_n = T \left( 2^n \right) = 2 X_{n-1} + c_2 2^n \]

\[ X_0 = T \left( 2^0 \right) = T (1) = c_1 \]

Solve by usual methods for recurrence equations

\[ X_n = O(n2^n) \]

so \( T(N) = O(N \log N) \)