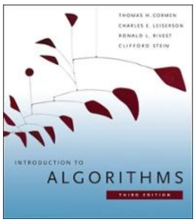
CS146 Data Structures and Algorithms



Chapter 4: Divide-and-Conquer

L4.1

Recurrence Relations

- Equation or an inequality that characterizes a function by its values on smaller inputs.
- Recurrence relations **arise when we analyze the running time of iterative or recursive algorithms**.
  - **Ex:** Divide and Conquer.

    $T(n) = \Theta(1)$ if $n \le c$

    $T(n) = a\,T(n/b) + D(n) + C(n)$ otherwise

L4.2

Recurrences

- The expression:

$$T(n) = \begin{cases} c & n = 1 \\ 2T\left(\frac{n}{2}\right) + cn & n > 1 \end{cases}$$

is a *recurrence*.

  - Recurrence: an equation that describes a function in terms of its value on smaller functions

L4.3

Recurrence Examples

$$s(n) = \begin{cases} 0 & n = 0 \\ c + s(n-1) & n > 0 \end{cases}$$

$$s(n) = \begin{cases} 0 & n = 0 \\ n + s(n-1) & n > 0 \end{cases}$$

$$T(n) = \begin{cases} c & n = 1 \\ 2T\left(\frac{n}{2}\right) + c & n > 1 \end{cases}$$

$$T(n) = \begin{cases} c & n = 1 \\ aT\left(\frac{n}{b}\right) + cn & n > 1 \end{cases}$$

L4.4

Recurrences

- A recurrence is an equation or inequality that describes a function in term of its value on smaller inputs.

  ex1. $T(n) = 1$ if $n = 1$,
  $= T(n-1) + 1$ if $n > 1$.
  Solution: $T(n) = n$.

  ex2. $T(n) = aT(n/b) + f(n)$
- Methods for solving recurrences
  (Obtaining asymptotic bounds on the solution, Θ or O)
  - Substitution method
  - Recursion-tree method
  - Master method

L4.5

Technicalities

- We neglect certain technical details when we state and solve recurrences.
  1. The assumption of integer arguments to functions.
  2. Boundary conditions is ignored.
  3. Ignore floors, ceilings.
- Exact vs. Asymptotic functions.
  - In algorithm analysis, both the recurrence and its solution **are expressed using asymptotic notation.**
  - Ex: Recurrence with exact function
    $T(n) = 1$ if $n = 1$
    $T(n) = 2T(n/2) + n$ if $n > 1$
    Solution: $T(n) = n \lg n + n$
    - Recurrence with asymptotics (BEWARE!)
      $T(n) = \Theta(1)$ if $n = 1$
      $T(n) = 2T(n/2) + \Theta(n)$ if $n > 1$
    Solution: $T(n) = \Theta(n \lg n)$

L4.6

4.3 Substitution method

- The substitution method for solving recurrence entails two steps:
  1. Guess the form of the solution.
  2. Use mathematical induction to find the constants and show that the solution works.

L4.7

Example

$$\begin{cases} T(n) = 2T(\lfloor n/2 \rfloor) + n \\ T(1) = 1 \end{cases}$$

(We may omit the initial condition later.)

1.Guess $T(n) = O(n \lg n)$ ***i.e.*** $T(n) \le c\, n \lg n$

2. Induction.
   Inductive hypothesis is that $T(k) = k \lg k$ for $k < n$.
   Assume this bound holds for $\lfloor n/2 \rfloor$

$$T(\lfloor n/2 \rfloor) \le c\lfloor n/2 \rfloor \lg \lfloor n/2 \rfloor$$

L4.8

$$T(n) \le 2(c\lfloor n/2 \rfloor \lg \lfloor n/2 \rfloor) + n \le cn \lg \frac{n}{2} + n$$
$$= cn \lg n - cn \lg 2 + n \le cn \lg n \quad \textbf{\textit{(if}} \;\; c \ge 1.)$$

Initial condition $1 = T(1) < cn \lg 1 = 0 (\rightarrow\leftarrow)$

However, $4 = T(2) \le c2 \lg 2 \quad (\textbf{\textit{if}} \;\; c \ge 2)$

$5 = T(3) \le c3 \lg 3 \quad (\textbf{\textit{if}} \;\; c \ge 2)$

lg2=1, lg3= 1.585

L4.9