Overview of Part 4

Theme: Relating

Small problems $\rightsquigarrow$ Big problems

* proof techniques using

  Small $\rightsquigarrow$ Big relationships

  + proof by smallest counter example
  + Proof by induction

* Problem solving techniques using

  Small $\rightsquigarrow$ Big relationships

  + Recursion
  + Divide & Conquer

* Running time of those techniques

  + Recurrence

Overview of Part 4

L10 :

* Smallest Counter Example
* Induction

L11 :

* Recursion
* Recurrence

L12 :

* Divide & Conquer
* Recurrence

L13 :

* Advanced Induction

Strong Induction Example

$p(n)$: n is power of prime number <u>or</u>

product of powers of prime numbe

prove: $\forall n \in N^+ \; p(n)$ $(\bigstar)$

---

proof: <u>Base case</u>: $1 = 2^0$

$p(1)$ true

<u>Induction Hypothesis</u>:

$p(k)$ true for all $k \in N^+, k < n$ $(*)$

<u>Induction</u>: consider n

① $n = p = p^1$ true

② $n = r \cdot s$ $r, s \in N^+, < n$

$(*) \Rightarrow p(r), p(s)$ true

$\Rightarrow p(n)$ true.

$(\bigstar)$ is true.

Recap of L11

* Recursion:

  Reduce problem of size n into subproblems of size $n-1, \ldots$

* Example: Towers of Hanoi

---

* Recurrences for running time of recursive programs

$$M(n) = \begin{cases} 1 & n=1 \\ 2M(n-1)+1 & n>1 \end{cases} = 2^n - 1$$

$$S(n) = \begin{cases} 0 & n=0 \\ 2S(n-1) & n>0 \end{cases} = 2^n$$

$$T(n) = \begin{cases} b & n=0 \\ rT(n-1)+a & n>0 \end{cases} = r^n b + a\frac{1-r^n}{1-r}$$

$$T(n) = \begin{cases} a & n=0 \\ rT(n-1)+g(n) & n>0 \end{cases} = r^n a + \sum_{i=1}^{n} r^{n-i} g(i)$$

L12 overview

* Divide & conquer

  Reduce problem of size n

  into subproblems of size n/m

* Binary Search : Example

* Recurrences for Divide & Conquer

  programs running time of

$$T(n) = \begin{cases} 1 & n=1 \\ T\left(\frac{n}{2}\right)+1 & n>1 \end{cases} = 1+\log_2 n$$

$$T(n) = \begin{cases} 1 & n=1 \\ 2T\left(\frac{n}{2}\right)+n & n>1 \end{cases} = n+\log_2 n$$

$$T(n) = \begin{cases} 1 & n<3 \\ 3T\left(\frac{n}{3}\right)+n & n\geq 3 \end{cases} = n+n\log_3 n$$

$$T(n) = \begin{cases} 1 & n=1 \\ 4T\left(\frac{n}{2}\right)+n & n>1 \end{cases} = 2n^2-n$$

* Big-O Notation : $O(n)$, $\Theta(n)$

* Three cases for halfing problems

$$T(n) = \begin{cases} > 0 & n = 1 \\ aT(\frac{n}{2}) + n & n > 1 \end{cases}$$

$$T(n) = \begin{cases} \Theta(n) & a < 2 \\ \Theta(n \log n) & a = 2 \\ \Theta(n^{\log_2 a}) & a > 2 \end{cases}$$

* Lemma 4.3

Solving Recurrence for Binary Search

$$T(n) = \begin{cases} 1 & n=1 \\ T(\frac{n}{2})+1 & n>0 \end{cases}$$

Assume: $n = 2^j$ for some j

$$T(n) = T(\tfrac{n}{2})+1$$

$$= T(\tfrac{n}{2^2})+2$$

$$T(\tfrac{n}{2}) = T(\tfrac{n}{2^2})+1$$

$$T(\tfrac{n}{2^2}) = T(\tfrac{n}{2^3})+1$$

$$= T(\tfrac{n}{2^3})+3$$

$\vdots$ step i

$$= T(\tfrac{n}{2^i})+i$$

$\vdots$ Final step

$$= T(\tfrac{n}{2^j})+j$$

$$= T(1)+j$$

$$= 1+\log_2 n$$

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

$$T(n) = 2T(\tfrac{n}{2}) + n$$

$$= 2(2T(\tfrac{n}{2^2}) + \tfrac{n}{2}) + n \qquad T(\tfrac{n}{2}) = 2T(\tfrac{n}{4}) + \tfrac{n}{2}$$

$$= 2^2 T(\tfrac{n}{2^2}) + 2n$$

$$= 2^2(2T(\tfrac{n}{2^3}) + \tfrac{n}{2^2}) + 2n \qquad T(\tfrac{n}{2^2}) = 2T(\tfrac{n}{2^3}) + \tfrac{n}{2^2}$$

$$= 2^3 T(\tfrac{n}{2^3}) + 3n$$

$$\vdots$$

$$= 2^i T(\tfrac{n}{2^i}) + in$$

After i steps

$$\vdots$$

Assume $n = 2^j$

$$= 2^j T(\tfrac{n}{2^j}) + jn$$

$$= n\,T(1) + n\log_2 n$$

$$= n + n\log_2 n$$

$$T(n) = \begin{cases} 1 & n < 3 \\ 3T\left(\frac{n}{3}\right) + n & n \geq 3 \end{cases}$$

$$T(n) = 3T\left(\frac{n}{3}\right) + n$$

$$= 3\left(3T\left(\frac{n}{3^2}\right) + \frac{n}{3}\right) + n \qquad T\left(\frac{n}{3}\right) = 3T\left(\frac{n}{3^2}\right) + \frac{n}{3}$$

$$= 3^2 T\left(\frac{n}{3^2}\right) + 2n$$

$$= 3^2\left(3T\left(\frac{n}{3^3}\right) + \frac{n}{3^2}\right) + 2n \qquad T\left(\frac{n}{3^2}\right) = 3T\left(\frac{n}{3^3}\right) + \frac{n}{3^2}$$

$$= 3^3 T\left(\frac{n}{3^3}\right) + 3n$$

$$\vdots$$

After i steps

$$= 3^i T\left(\frac{n}{3^i}\right) + in$$

$$\vdots$$

Assume $n = 3^j$

$$= 3^j T\left(\frac{n}{3^j}\right) + jn$$

$$= n\,T(1) + n\log_3 n$$

$$= n + n\log_3 n$$

~~L13-T4~~
L13-6

$$T(n) = \begin{cases} 1 & n = 1 \\ 4T(\frac{n}{2}) + n & n > 1 \end{cases}$$

$$T(n) = 4T(\frac{n}{2}) + n$$

$$T(\frac{n}{2}) = 4T(\frac{n}{2^2}) + \frac{n}{2}$$

$$= 4^2 T(\frac{n}{2^2}) + \frac{4}{2}n + n$$

$$T(\frac{n}{2^2}) = 4T(\frac{n}{2^3}) + \frac{n}{2^2}$$

$$= 4^3 T(\frac{n}{2^3}) + (\frac{4}{2})^2 n + \frac{4}{2}n + n$$

$$\vdots$$

Assume $n = 2^j$

$$= 4^j T(\frac{n}{2^j}) + (\frac{4}{2})^{j-1} n + \cdots + \frac{4}{2}n + n$$

$$= 2^{2j} T(1) + n \sum_{i=0}^{j-1} (\frac{4}{2})^i$$

$$= n^2 + n \frac{2^j - 1}{2 - 1}$$

$$= n^2 + n(n-1) = 2n^2 - n$$

L13-8

Big-O Examples

$g(n) = n^2$

$f(n) = 4n^2$

$$f(n) \le 4g(n), \quad n \ge 0 \Rightarrow f = \Theta(g)$$

$f(n) = 8n^2 + 2n - 3$

$$\le 10g(n), \quad n > 10 \Rightarrow f = O(g)$$

$f(n) = n^2/5 + \sqrt{n} + 10\log n$

$$\le g(n), \quad n > 10 \Rightarrow f = O(g)$$

In all cases, we can say: the running time is $O(n^2)$.

Convey essential info, & Simple.

use of big-O, Θ

* $n^2 \le 3n^2 + n \le 4n^2, \quad n \geq 10$

  If program has running time

  $$3n^2 + n,$$

  we can simply say : running time is $\Theta(n$

* $n \log n \le \frac{n}{5} + 10 n \log n \le 100 n \log n, \quad n > 10$

  Running time : $\frac{n}{5} + 10 n \log n$

  $$\Rightarrow \Theta(n \log n)$$

* $\frac{n^2}{5} \le \frac{n^2}{5} + 10 n \log n \le n^2, \quad n > 10$

  Running time : $\frac{n^2}{5} + 10 n \log n$

  $$\Rightarrow \Theta(n^2)$$

$\boxed{\text{Report Dominating term only}}$

Lemm 4.3: $\sum_{i=0}^{n-1} r^i = o(t(n))$

Claim: $t(n) = O(\sum_{i=1}^{n-1} r^i)$ (why?

$\Rightarrow$ Theorem 4.4:

$$\sum_{i=0}^{n-1} r^i = o(t(n))$$

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

$$T(n) = T(\frac{n}{2}) + n$$

$$T(\frac{n}{2}) = T(\frac{n}{2^2}) + \frac{n}{2}$$

$$= T(\frac{n}{2^2}) + \frac{n}{2} + n$$

$$T(\frac{n}{2^2}) = T(\frac{n}{2^3}) + \frac{n}{2^2}$$

$$= T(\frac{n}{2^3}) + \frac{n}{2^2} + \frac{n}{2} + n$$

... Assume $n = 2^j$

$$= T(\frac{n}{2^j}) + \frac{n}{2^{j-1}} + \cdots + \frac{n}{2} + n$$

$$= 1 + n\left(1 + \frac{1}{2} + \left(\frac{1}{2}\right)^2 + \cdots + \left(\frac{1}{2}\right)^{j-1}\right)$$

$$= 1 + O(n) \qquad \text{Theorem 4.4}$$

$$= O(n)$$

Three recurrences

$T(n) = T\left(\frac{n}{2}\right) + n \quad = \qquad\qquad \Theta(n)$

$T(n) = 2T\left(\frac{n}{2}\right) + n = n + n\log_2 n = \Theta($

$T(n) = 4\,T\left(\frac{n}{2}\right) + n = 2n^2 - n \quad = \Theta(\qquad)$

Lemma 4.7, Case 3

Assume $n = 2^j$.

Follow L13-7 :

$$T(n) = a\,T\left(\frac{n}{2}\right) + n$$

$$= a^j\,T(1) + n \sum_{i=0}^{j-1} \left(\frac{a}{2}\right)^i$$

$$n \sum_{i=1}^{j-1} \left(\frac{a}{2}\right)^j = \Theta\left(n \left(\frac{a}{2}\right)^{j-1}\right)$$

$$= \Theta\left(n \frac{a^j}{2^j} \frac{2}{a}\right) = \Theta\left(n \frac{a^{\log_2 n}}{2^{\log_2 n}}\right)$$

$$= \Theta\left(a^{\log_2 n}\right)$$

$$T(n) = a^{\log_2 n} + \Theta\left(a^{\log_2 n}\right)$$

$$= \Theta\left(a^{\log_2 n}\right)$$