Recurrences

Analyzing run time of a recursive algorithm

MergeSort

$T(n)$ time to mergesort a list of n numbers

$$T(0) = \Theta(1)$$
$$T(1) = \Theta(1)$$

$n > 1$:

$$T(n) = T\left(\left\lfloor \tfrac{n}{2} \right\rfloor\right) + T\left(\left\lceil \tfrac{n}{2} \right\rceil\right) + \Theta(n)$$

$$T(1) = \Theta(1)$$
$$T(n) = T\left(\left\lfloor \tfrac{n}{2} \right\rfloor\right) + T\left(\left\lceil \tfrac{n}{2} \right\rceil\right) + \Theta(n)$$

→ $T(1) = d$ (constant)

→ $T(n) \underset{\geq}{\leq} T\left(\left\lfloor \tfrac{n}{2} \right\rfloor\right) + T\left(\left\lceil \tfrac{n}{2} \right\rceil\right) + cn$

(c, d constants)

Base case is always, "If n is $\leq \underline{n_0}$ constant then $T(n)$ is bounded by a <u>constant</u>."

Asymptotics of T are independent of these constants, so we will usually omit the base case.

$$T(n) = T\left(\left\lceil \tfrac{n}{2} \right\rceil\right) + T\left(\left\lfloor \tfrac{n}{2} \right\rfloor\right) + cn$$

(often set $c := 1$) so asymptotics unaffected

$$T(n) = T\left(\left\lfloor \tfrac{n}{2} \right\rfloor\right) + T\left(\left\lceil \tfrac{n}{2} \right\rceil\right) + n$$

Note: $n = \left\lfloor \tfrac{n}{2} \right\rfloor + \left\lceil \tfrac{n}{2} \right\rceil$ always (n integer)

At some point, we will generally ignore ceilings & floors. They usually don't matter

$$T(n) = 2T\left(\left\lfloor \tfrac{n}{2} \right\rfloor\right) + n$$

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

Digression: Quiz 2 solution

$$f(n) = \overset{o}{\cancel{\Theta}}(n)$$
$$f(n) = \omega(n^{1-\varepsilon}) \quad \text{for all } \varepsilon > 0$$

$f(n) = n$ ✓

$f(n) = \frac{n}{\lg n}$ ✓

Solving recurrences

3 methods:

1. substitution method
   rigorous proof by induction
2. tree method
   unwind the recursion completely
3. master method
   plug & chug

Sep 8-12:45 PM

Substitution method

Proof by induction about T.

$T(0) := 1$

$n>0$: $T(n) := 7T\left(\lfloor \frac{n}{3} \rfloor\right) + n^2$

Prove, using subst method, that $T(n) = O(n^2)$.

That is, there exist constants c, n_0 such that

(*) $\boxed{T(n) \leq cn^2}$ for all $n \geq n_0$.

Fix n large enough.
Assume that (*) for all large enough $m < n$, that is, $T(m) \leq cm^2$

Now:

$$T(n) = 7T\Big(\underbrace{\lfloor \tfrac{n}{3} \rfloor}_{<n}\Big) + n^2$$

$$\leq 7c\lfloor \tfrac{n}{3} \rfloor^2 + n^2 \qquad (m := \lfloor \tfrac{n}{3} \rfloor)$$

inductive hypothesis

$$\leq 7c\left(\tfrac{n}{3}\right)^2 + n^2$$

$$= \tfrac{7}{9}cn^2 + n^2$$

$$= \left(\tfrac{7}{9}c + 1\right)n^2$$

$$\overset{?}{\leq} cn^2 \leftarrow \text{goal}$$

provided $\left(\frac{7}{9}c + 1\right) \leq c$

$$1 \leq \tfrac{2}{9}c$$

$$\boxed{c \geq \tfrac{9}{2}}$$

$$T(0) = 1 \leq \underbrace{c0^2}_{0}$$

So can't choose n_0 to be 0.
How about $n_0 := 1$?

$T(0) = 1$

$T(1) = 7T\Big(\underbrace{\lfloor \tfrac{1}{3} \rfloor}_{0}\Big) + 1^2$

$= 7T(0) + 1 = 8$

$T(2) = 7T\left(\lfloor \tfrac{2}{3} \rfloor\right) + 2^2$

$= 7 \cdot 1 + 4 = 11$

$T(1) = 8 \leq c1^2 = c$

$T(2) = 11 \leq 8 \cdot 2^2$ ✓

for $n \in \{1, 2\}$,
$T(n) \leq cn^2$ for all $c \geq 8$.

Take any $c \geq 8$

$T(1) \leq c1^2$

$T(2) \leq c2^2$

$T(3) \leq c3^2$

because

$T(3) = 7T\Big(\underbrace{\lfloor \tfrac{3}{3} \rfloor}_{1}\Big) + 3^2$

$= 7\underbrace{T(1)}_{\substack{n=1 \text{ works} \\ \leq c1^2}} + 3^2$

So $T(n) \leq 8n^2$ for all $n \geq 1$. //

Sep 8-1:06 PM

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

Show that $T(n) = O(n)$.

Inductive step: $O(n)$

prove $\underline{T(n) \le cn}$

assuming $T(m) \le cm$

for all $m < n$.

$T(n) = \underline{T(\lfloor \frac{n}{2} \rfloor)} + \underline{T(\lfloor \frac{n}{3} \rfloor)} + n$

$\le c\lfloor \frac{n}{2} \rfloor + c\lfloor \frac{n}{3} \rfloor + n$

$\le c(\frac{n}{2}) + c(\frac{n}{3}) + n$ $\boxed{\text{goal: } T(n) \le cn}$

$= n(\frac{c}{2} + \frac{c}{3} + 1)$

$= n(\underline{\frac{5}{6}c + 1})$

$\le cn$

provided $\frac{5}{6}c + 1 \le c$

i.e., $c \ge 6$.

$T(n) = T(n-1) + 1$

"prove" that $T(n) = O(1)$.

[false]

Assume

$T(m) = O(1)$ for $m < n$.

$T(n) = T(n-1) + 1$

$= \underline{O(1)} + 1$

$= \underline{O(1)}$. $\Rightarrow\Leftarrow$

Proof that $T(n) \le c$.

$T(n) = \underline{T(n-1)} + 1$

$\le c + 1$

$\not\le c$

no c works.

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

prove that $T(n) = O(n)$?

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

$\le 2c(\frac{2n}{3}) + n$

$= n(\frac{4}{3}c + 1)$

$\overset{?}{\le} cn$ fail

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

Show $T(n) = \Omega(n)$

Prove by induction that

$\rightarrow \underline{T(n) \ge cn}$ (some $c > 0$ constant)

$T(n) = 2\underbrace{T(\frac{n}{2})}_{\ge c(\frac{n}{2})} + n$

$\ge 2c(\frac{n}{2}) + n$

$= cn + n$

$= (c+1)n$

$\ge cn$

true for any $c > 0$.

Tight bound on T.

$T(n) = \Theta(\underline{n \lg n})$.

Sep 8-1:27 PM