Digital Logic Design: a rigorous approach ©

Chapter 7: Asymptotics

part 2: recurrence eqs

Guy Even Moti Medina

School of Electrical Engineering Tel-Aviv Univ.

April 6, 2020

Book Homepage:
http://www.eng.tau.ac.il/~guy/Even-Medina

In this section we deal with the problem of solving or bounding the rate of growth of functions $f : \mathbb{N}^+ \to \mathbb{R}$ that are defined recursively. We consider the typical cases that we will encounter later.

Recurrence 1

Consider the recurrence

$$f(n) \triangleq \begin{cases} 1 & \text{if } n = 1 \\ n + f(\lfloor \frac{n}{2} \rfloor) & \text{if } n > 1. \end{cases}$$

- Why is $f(n)$ interesting?
- What is the rate of growth of $f(n)$?

$$\hat{f}(n) \triangleq \begin{cases} 0 & \text{if } n = 1 \\ \frac{n}{2} + \hat{f}(\lfloor \frac{n}{2} \rfloor) & \text{if } n > 1. \end{cases} \qquad [f(n) = 1 + 2\hat{f}(n) = \Theta(\hat{f}(n))]$$

exercise

Algorithm 1 $MAX(x_1, \ldots, x_n)$ - assume n is a power of 2

1. Base Case: If $n = 1$ then return x_1.
2. Reduction Rule:
   1. For $i = 1$ to $\frac{n}{2}$ Do: $y_i \leftarrow \max\{x_{2i-1}, x_{2i}\}$
   2. Return $MAX(y_1, \ldots, y_{n/2})$

$x_1 \; x_2 \quad x_3 \; x_4 \; \cdots$

$y_1 \qquad y_2$

Claim

Number of comparisons in $MAX(x_1, \ldots, x_n)$ equals $\hat{f}(n)$.

Recurrence 1 - analysis

$$f(n) \triangleq \begin{cases} 1 & \text{if } n = 1 \\ n + f\left(\left\lfloor \frac{n}{2} \right\rfloor\right) & \text{if } n > 1. \end{cases}$$

Lemma

The rate of growth of the function $f(n)$ is $\Theta(n)$.

- start by proving for powers of 2.
- what ~~of~~ if n is not a power of 2?
- what about $f(n) = n + f(\lceil \frac{n}{2} \rceil)$?

$n = 2^k$

$\Rightarrow \lfloor \frac{n}{2} \rfloor = \frac{n}{2}$

claim: $f(2^k) = 2 \cdot 2^k - 1$

proof 1: by ind. on k

basis: $k=0$ $f(2^0) = 1$

$2 \cdot 2^0 - 1 = \underline{1}$

hyp: claim holds for k.

step: $f(2^{k+1}) = 2^{k+1} + f(2^k)$

$= 2^{k+1} + 2 \cdot 2^k - 1$

$= 2 \cdot 2^{k+1} - 1$

easy if you guess a solution! $\boxtimes$

$$f(2^k) = \underbrace{2^k}_{\text{penalty}} + \underbrace{f(2^{k-1})}_{\text{recursive value}}$$

$$f(2^0) = \underline{1}$$

$$f(2^k) = \text{sum penalties}$$
$$= 2^k + 2^{k-1} + 2^{k-2} + \dots + 2^0$$
$$= \boxed{2 \cdot 2^k - 1}$$

penalties

$$\begin{tikzcd}
\boxed{f(2^k)} & 2^k \arrow[l] \\
\boxed{f(2^{k-1})} \arrow[u] & 2^{k-1} \arrow[l] \\
\boxed{f(2^{k-2})} \arrow[u] & 2^{k-2} \arrow[l] \\
\vdots \arrow[u] & \\
\boxed{f(2^0)} \arrow[u] & 1
\end{tikzcd}$$

2nd proof of $f(2^k) = 2 \cdot 2^k - 1$

build a recursion tree

$$f(2^k) = 2^k + f(2^{k-1})$$

$$= 2^k + 2^{k-1} + f(2^{k-2})$$

$$\vdots$$

$$= 2^k + 2^{k-1} + \cdots + 1$$

"penalties"

$\boxed{f(2^k)}$ 2^k

$\boxed{f(2^{k-1})}$ 2^{k-1}

$\vdots$

$\boxed{f(2^0)}$ 2^0

$f(2^k)$ = sum of penalties $= 2^{k+1} - 1$

Is it enough to solve for powers of 2?

Lemma

Assume that:

1. *The functions $f(n)$ and $g(n)$ are both monotonically nondecreasing.*
2. *The constant ρ satisfies, for every $k \in \mathbb{N}$,*

$$\frac{g(2^{k+1})}{g(2^k)} \leq \rho \, .$$

Then,

1. *If $f(2^k) = O(g(2^k))$, then $f(n) = O(g(n))$.*
2. *If $f(2^k) = \Omega(g(2^k))$, then $f(n) = \Omega(g(n))$.*

exercise!

claim: $f, g \nearrow$, $\frac{g(2^{k+1})}{g(2^k)} \leq \rho$, $f(2^k) = O(g(2^k))$

$\Rightarrow \quad f(n) = O(g(n))$

proof $\exists c \, \exists K \quad \forall k > K : f(2^k) \leq c \cdot g(2^k)$

$\Rightarrow \forall n > 2^K$: sandwich: $2^k \leq n < 2 \cdot 2^k$

$$\begin{aligned}
f(n) &\leq f(2^{k+1}) && (f \nearrow) \\
&\leq c \cdot g(2^{k+1}) && (f(2^k) = O(g(2^k))) \\
&\leq c \cdot \rho \cdot g(2^k) && \left(\frac{g(2^{k+1})}{g(2^k)} \leq \rho\right) \\
&\leq \underbrace{c \cdot \rho}_{\text{new constant}} \cdot g(n) && (g \nearrow)
\end{aligned}$$

$\boxtimes$

back to rec. #1 : $f(n) = \begin{cases} 1 & \text{if } n=1 \\ n + f(\lfloor \frac{n}{2} \rfloor) & \text{if } n>1 \end{cases}$

what do we know?

1) $f \nearrow$ (exercise)

2) $f(2^k) \leq 2 \cdot 2^k = O(g(2^k))$, $g(n) \doteq n$

3) $g \nearrow$

4) $$\frac{g(2^{k+1})}{g(2^k)} = 2$$

$$\Longrightarrow \quad f(n) = O(g(n)) \qquad [f(n) = \Omega(g(n))]$$

! no need to "worry" about floor func.

Recurrence 2.

Consider the recurrence

$$f(n) \triangleq \begin{cases} 1 & \text{if } n = 1 \\ n + 2 \cdot f(\lfloor \frac{n}{2} \rfloor) & \text{if } n > 1. \end{cases}$$

- Why is $f(n)$ interesting?
- What is the rate of growth of $f(n)$?

Recurrence 2 - motivation

$$\hat{f}(n) \triangleq \begin{cases} 0 & \text{if } n = 1 \\ (n-1) + 2\hat{f}(\lfloor \frac{n}{2} \rfloor) & \text{if } n > 1. \end{cases}$$

exercise

$$[f(n) = \hat{f}(n) + 2n - 1]$$

Algorithm 2 *SORT*$(x_1, \ldots, x_n)$ - assume n is a power of 2

Merge Sort

1. Base Case: If $n = 1$ then return x_1.
2. Reduction Rule:
   1. $(y_1, \ldots, y_{n/2}) \leftarrow SORT(x_1, \ldots, x_{n/2})$
   2. $(y_{n/2+1}, \ldots, y_n) \leftarrow SORT(x_{n/2+1}, \ldots, x_n)$
   3. Return $MERGE((y_1, \ldots, y_{n/2}), (y_{n/2+1}, \ldots, y_n))$

Claim

Number of comparisons in $SORT(x_1, \ldots, x_n)$ *equals* $\hat{f}(n)$.

$$f(n) \triangleq \begin{cases} 1 & \text{if } n = 1 \\ n + 2 \cdot f(\lfloor \frac{n}{2} \rfloor) & \text{if } n > 1. \end{cases}$$

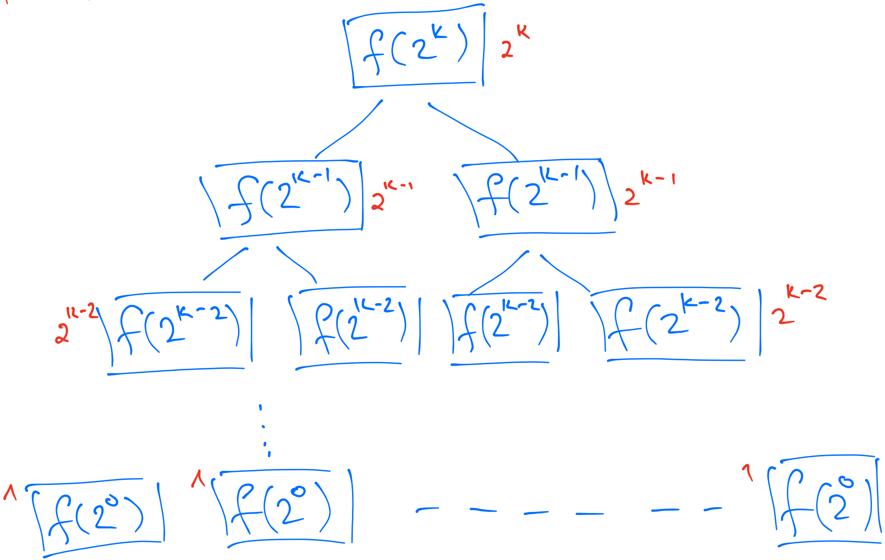
Lemma

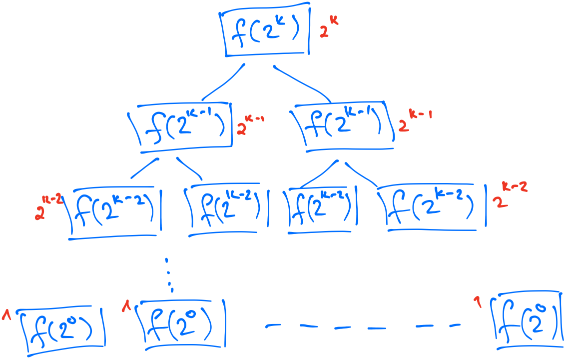
The rate of growth of the function $f(n)$ is $\Theta(n \log n)$.

- prove for powers of 2
- extend to arbitrary n

claim: $f(2^k) = (k+1) \cdot 2^k$

proof: recursion tree





penalties

$$1 \cdot 2^k = 2^k$$

$$2 \cdot 2^{k-1} = 2^k$$

$$4 \cdot 2^{k-2} = 2^k$$

$$2^k \cdot 1 = 2^k$$

$$(k+1) \cdot 2^k$$

$\boxtimes$

what about $n \neq 2^k$?

we know $f(2^k) = (k+1) \cdot 2^k$.

$g(n) \triangleq n \cdot \log_2 n \quad \left(f(2^k) = \Theta(g(2^k)) \right)$

now: $f \uparrow$, $g \uparrow$, and

$$\frac{g(2^{k+1})}{g(2^k)} \leq \frac{2^{k+1} \cdot (k+1)}{2^k \cdot k} \leq 4$$

So: $f(n) = \Theta(n \cdot \log n)$

$\boxtimes$

Consider the recurrence

$$f(n) \triangleq \begin{cases} 1 & \text{if } n = 1 \\ n + 3 \cdot f\left(\left\lfloor \frac{n}{2} \right\rfloor\right) & \text{if } n > 1. \end{cases}$$

Lemma

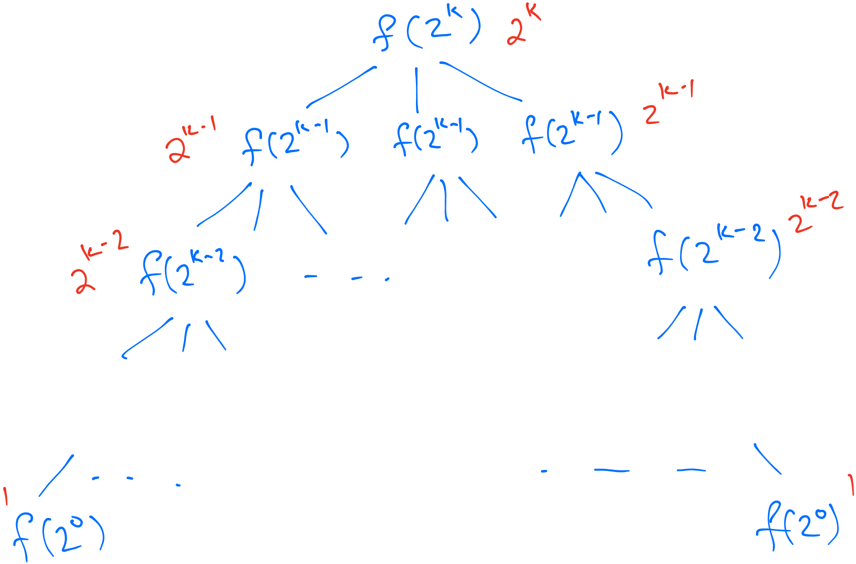
The rate of growth of the function $f(n)$ is $\Theta(n^{\log_2 3})$.

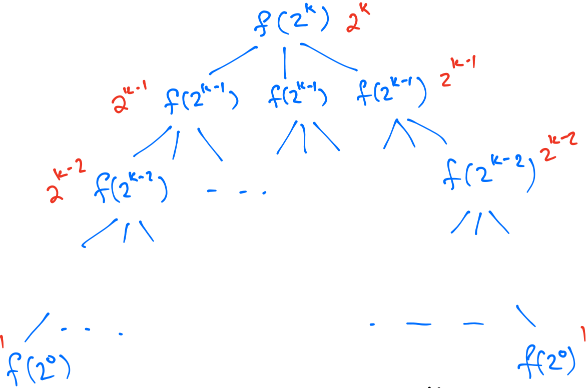
hint: $f(2^k) = 3^{k+1} - 2^{k+1}$.

$\sim \Theta(n^{1.6})$

claim: $f(2^k) = 3^{k+1} - 2^{k-1}$

proof: recursion tree





Sum of penalties

$3^0 \cdot 2^k$

$3^1 \cdot 2^{k-1}$

$3^2 \cdot 2^{k-2}$

$\vdots$

$3^k \cdot 2^0$

$$\sum_{i=0}^{k} 2^i \cdot 3^{k-i} = 3^k \cdot \sum_{i=0}^{k} \left(\frac{2}{3}\right)^i = 3^k \cdot \frac{1-\left(2/3\right)^{k+1}}{1-2/3}$$

$$= 3^{k+1} - 2^{k+1} \qquad \boxtimes$$

what about $n \neq 2^k$?

define $g(2^k) = 3^k$

$$g(n) = 3^{\log_2 n} = n^{\overbrace{\log_2 3}^{\approx 1.6}}$$

know: $f(2^k) = \Theta(g(2^k))$

$f \uparrow$, $g \uparrow$ exercise

$$\frac{g(2^{k+1})}{g(2^k)} = 3$$

$$\Longrightarrow \quad f = \Theta(g).$$

Example - 1

Consider the recurrence

$$f(n) \triangleq \begin{cases} c & \text{if } n = 1 \\ a \cdot n + b + f\left(\left\lfloor \frac{n}{2} \right\rfloor\right) & \text{if } n > 1, \end{cases}$$

non-neg.

where a, b, c are constants. $a \neq 0$

Lemma

The rate of growth of the function $f(n)$ is $\Theta(n)$.

proof: $f(2^k) = 2a \cdot 2^k + b \cdot k + c - 2a ...$

$f(n) = an + b + f(n/2) \quad n > 1 \qquad f(1) = c$

Solve $f(2^k)$:

	penalty
$f(2^k)$	$a \cdot 2^k + b$
$f(2^{k-1})$	$a \cdot 2^{k-1} + b$
$\vdots$	
$f(2^0)$	c

$$\sum_{i=1}^{k} (a \cdot 2^i + b) + c = a \cdot (2^{k+1} - 2) + bk + c$$

$$= \Theta(2^k)$$

Example -2

Consider the recurrence

$$f(n) \triangleq \begin{cases} c & \text{if } n = 1 \\ a \cdot n + b + 2 \cdot f(\lfloor \frac{n}{2} \rfloor) & \text{if } n > 1, \end{cases}$$

where $a, b, c = O(1)$. $a \neq 0$

Lemma

The rate of growth of the function $f(n)$ is $\Theta(n \log n)$.

proof: We claim that $f(2^k) = a \cdot k2^k + (b + c) \cdot 2^k - b \ldots$

try to sum penalties in recursion tree to find $f(2^k)$

Example - 3

Consider the recurrence

$$F(k) \triangleq \begin{cases} 1 & \text{if } k = 0 \\ 2^k + 2 \cdot F(k-1) & \text{if } k > 0, \end{cases}$$

Lemma

$F(k) = (k+1) \cdot 2^k.$ $= \Theta(2^k \cdot k)$

Proof: Define $f(n) \triangleq F(\lceil \log_2 n \rceil)$. Observe that $f(2^x) \triangleq F(x)$....

$F(k) = 2^k + 2 \cdot F(k-1)$ if $k > 0$, $F(0) = 1$

* recursion applied to $k-1$ not $k/2$

use substitution!

$f(2^k) \stackrel{\Delta}{=} F(k)$.

Then $f(2^k) = \begin{cases} F(0) = 1 & \text{if } k = 0 \\ 2^k + 2 \cdot F(k-1) & \text{if } k > 0 \\ = 2^k + 2 \cdot f(2^{k-1}) & \end{cases}$

f corresponds to rec #2!

we already know:

$$f(2^k) = \Theta(k \cdot 2^k)$$

$$\Rightarrow F(k) = \Theta(k \cdot 2^k)$$

$\boxtimes$

Examples with floor and ceiling

* how many times can you fold a rod of length n till it reaches 1/2 unit length?

1. $$f(n) \triangleq \begin{cases} 1 & \text{if } n = 1 \\ 1 + f(\lfloor \frac{n}{2} \rfloor) & \text{if } n > 1, \end{cases}$$

constant penalty

* SORT when n is odd.

2. $$f(n) \triangleq \begin{cases} 1 & \text{if } n = 1 \\ n + \underbrace{f(\lfloor \frac{n}{2} \rfloor) + f(\lceil \frac{n}{2} \rceil)} & \text{if } n > 1, \end{cases}$$

in $n = 2^k$, all ceil & floor funcs can be ignored!

instead of $2 \cdot f(\lceil \frac{n}{2} \rceil)$ or $2 \cdot f(\lfloor \frac{n}{2} \rfloor)$