Welcome back!

Lec 7

Recursive Correctness

2025-06-02

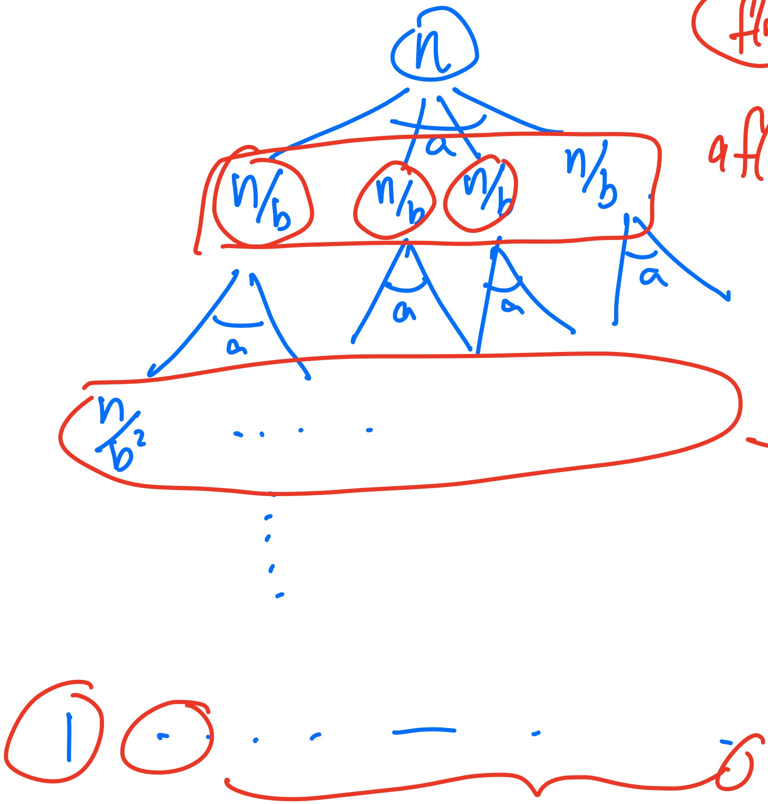
Review

Recursion Trees

$$T(n) = aT(n/b) + f(n)$$

$$\frac{n}{b^h} = 1$$

$$h = \log_b n.$$



$f(n)$

$a f(n/b)$

h

$a^2 f(n/b^2)$

$\vdots$

$a^h f(n/b^h).$

h

$$a^h = a^{\log_b n} = n^{\log_b a}$$

Total : $\displaystyle\sum_{h=0}^{\log_b n} a^h f(n/b^h)$

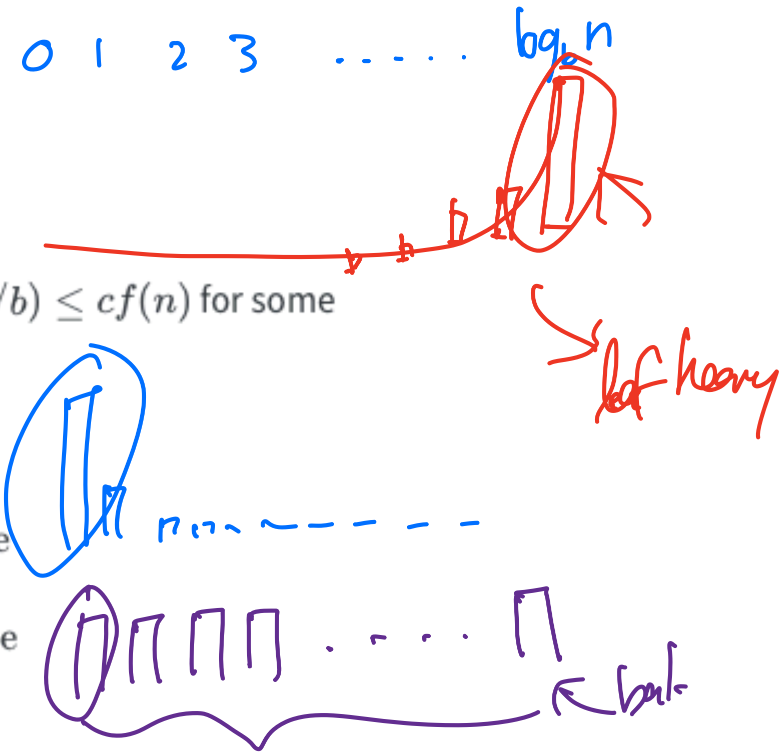
Master Theorem

Let $T(n) = aT(n/b) + f(n)$. Define the following cases based on how the root work compares with the leaf work.

1. **Leaf heavy**. $f(n) = O(n^{\log_b(a)-\epsilon})$ for some constant $\epsilon > 0$.
2. **Balanced**. $f(n) = \Theta(n^{\log_b(a)})$
3. **Root heavy**. $f(n) = \Omega(n^{\log_b(a)+\epsilon})$ for some constant $\epsilon > 0$, and $af(n/b) \leq cf(n)$ for some constant $c < 1$ for all sufficiently large n.

Then,

$$T(n) = \begin{cases} \Theta(n^{\log_b(a)}) & \text{Leaf heavy case} \\ \Theta(f(n)\log(n)) & \text{Balanced case} \\ \Theta(f(n)) & \text{Root heavy case} \end{cases}$$



New Stuff

Correctness

- For any algorithm/function/program, define a precondition and a postcondition.

- The **precondition** is an assertion about the inputs to a program.
- The **postcondition** is an assertion about the end of a program.

- An algorithm is **correct** if the precondition implies the postcondition.

- *“If I gave you valid inputs, your algorithm should give me the expected outputs.”*

Examples

→ Binary Search:

Inputs: $l \in List[\mathbb{N}]$, $t \in \mathbb{N}$,

Pre: $t \in l$, l is sorted

Post: index of t in l.

non example: ["a", {}, (1,1)]

→ Merge sort:

Input: l, a list.

Pre: l has elements that you can compare.: $\forall i,j$: $l[i]$ can be compared to $l[j]$

Post: a sorted version of l.

BFS

input: $G = (V,E)$, s, t

precondition: $s \in V$, $t \in V$

post: returns a path from s to t.

Correctness of Mergesort

Assuming merge is correct

```python
def merge_sort(l):
    n = len(l)
    if n <= 1:
        return l
    else:
        left = merge_sort(l[:n//2])
        right = merge_sort(l[n//2:])
        return merge(left, right)
```

For now, suppose all lists are lists of integers.

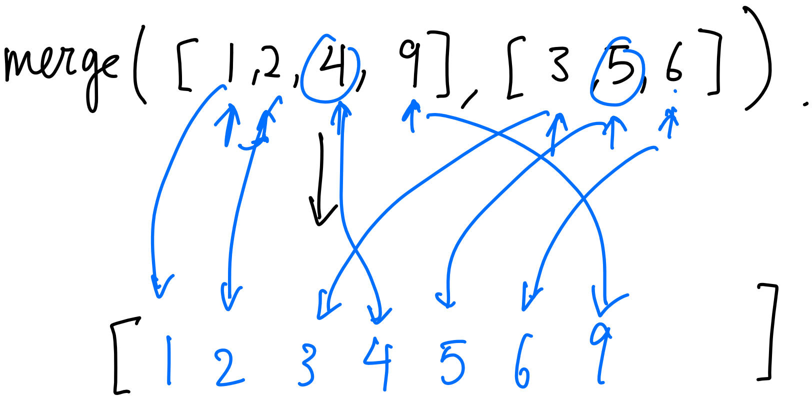
merge

inputs: left, right

pre: left, right are sorted.

post: a sorted list containing all the elements of left and right.

By induction on the length of the list.

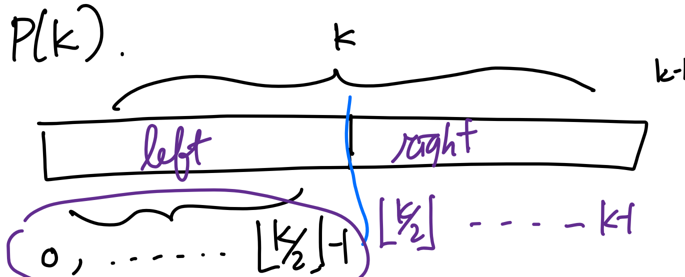


$P(n)$: $\forall$ lists, $l \in List[\mathbb{N}]$ s.t. $len(l) = n$,
merge sort is correct on l.

WTS: $\forall n \in \mathbb{N}, P(n)$

Base case: $n=0$, $n=1$ } falls under the base case of the function which just returns l. Any list of size 0 or 1 is already sorted, so this is correct.

Inductive step: let $k \geq 2$, and suppose $P(i)$ for all $i < k$. } IH.
WTS $P(k)$.



1) show that merge-sort($l[:\lfloor k/2 \rfloor]$) and mergesort($l[\lfloor k/2 \rfloor :]$) are covered by the IH.

$len(l[:\lfloor k/2 \rfloor]) = \lfloor k/2 \rfloor$ since $k \geq 2$, $\lfloor k/2 \rfloor < k$, so good!

$len(l[\lfloor k/2 \rfloor :]) = \lceil k/2 \rceil$ since $k \geq 2$: $\lceil k/2 \rceil < k$. so good again.

2) by IH, left, right are sorted versions of $l[:\lfloor k/2 \rfloor]$, $l[\lfloor k/2 \rfloor :]$
3) by correctness of merge: I return a sorted version of l.

Multiplication

Grade School

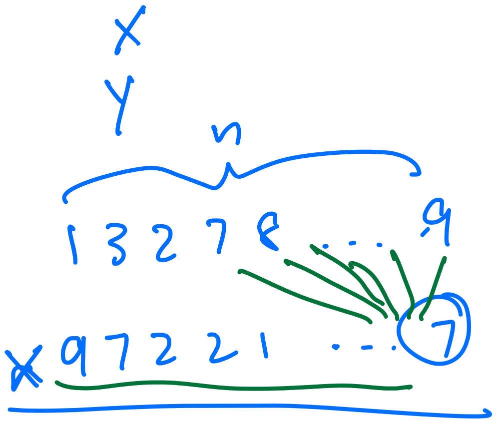
Multiplication :

input : $x \in \mathbb{N}$
$y \in \mathbb{N}$,

precondition : —

post : returns xy.

$$
\begin{array}{r}
{\scriptstyle 5\,1\;\;} \\
36 \\
\times \quad 9\cancel{3} \\
\hline
108 \\
+\; 3240 \\
\hline
3348
\end{array}
$$



Multiplication

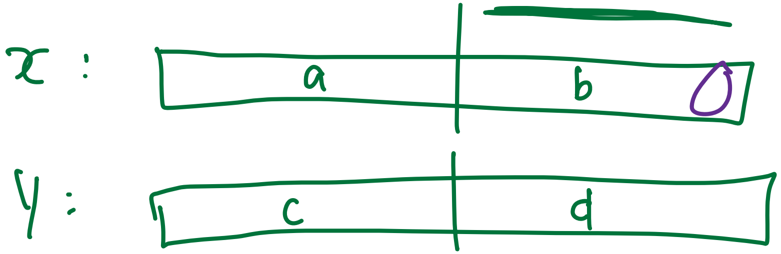
Divide and Conquer

$$(a+b)\cdot(c+d) = ac+ad+bc+bd.$$

$$\text{left } m = \lfloor n/2 \rfloor$$

$$x = a\cdot 10^m + b$$

$$y = c\cdot 10^m + d.$$



$$xy = (a10^m + b)(c10^m + d) = ac10^{2m} + \boxed{ad\cdot 10^m + bc\cdot 10^m} + bd.$$

$$T(n) = 4T(n/2) + n$$

leaf work: $n^{\log_2 4} = n^2 \gg n.$

so leaf heavy and

$$T(n) = \Theta(n^2).$$

$36 \cdot 93$:

$x = 3 \cdot 10 + 6$

$y = 9 \cdot 10 + 3$.

$a=3$
$b=6$
$c=9$
$d=3$
$n=2$
$m=1$

$ac = 27$
$ad = 9$
$bc = 54$
$bd = 18$

$2700 + 90 + 540 + 18$.

$= 3348$

---

goal : compute $xy = ac \, 10^{2m} + (ad+bc) \cdot 10^m + bd$

$\underbrace{(ad+bc)}$ ④ $= Z_1 \cdot 10^{2m} + (Z_3 - Z_1 - Z_2) 10^m + Z_2$

① compute $ac = Z_1$
② compute $bd = Z_2$
③ compute $(a+b) \cdot (c+d) = Z_3$.

$Z_1, Z_2, Z_3 \le n+2$ digits.
$\le 2n$

$Z_3 = (a+b)(c+d) = ac + ac + bc + bd$.

thus: $ad + bc = Z_3 - Z_2 - Z_1$

$T(n) = 3T(n/2) + n \rightarrow$ leaf: $n^{\log_2 3} \approx n^{1.58\ldots}$

$n^{\log_2 5}$

$\Rightarrow T(n) = \Theta(n^{\log_2 3})$. $\ll n^2$.

36, 93

$Z_1 = ac = 27$
$Z_2 = bd = 18$
$Z_3 = (a+b) \cdot (c+d) = 108$
$\quad 9 \quad 12$

$27 \cdot 100 + (\underbrace{108 - 27 - 18}_{35}) \cdot 10 + 18$

$2700 + 630 + 18$

$$\begin{array}{r} 2700 \\ 630 \\ 18 \\ \hline 3348 \end{array}$$

$\boxed{\text{best} : n \log(n)}$

Karatsuba

```
def karatsuba(x, y):
    if x < 10 and y < 10:
        return x * y
    n = max(len(str(x)), len(str(y)))
    m = n//2
 b  x_l = x // 10**m
 a  x_h = x % 10**m
 d  y_l = y // 10**m
 c  y_h = y % 10**m
    z_0 = karatsuba(x_l, y_l)
 z2 z_1 = karatsuba(x_l + x_h, y_l + y_h)
 z1 z_2 = karatsuba(x_u, y_h)
    return (z_2 * 10**(2*m)) + ((z_1 - z_2 - z_0) * 10**m) + z_0
```

Summary

Correctness of Recursive Algs:

1. By induction on the size of the inputs.
2. Base of the induction is the Base case of the recursive function.
3. Inductive step correspond to the recursive calls & how the results of those calls are combined.

Binary Search

```
def bin_search(l, t, a, b):
    if b == a:
        return None
    else:
        m = (a + b)//2
        if l[m] == t:
            return m
        elif l[m] < t:
            return bin_search(l, t, m+1, b)
        elif l[m] > t:
            return bin_search(l, t, a, m)
```