CSE 201 1-28-20

Sketch of Proof of Case(1) of Master Theorem.

$$T(n) = f(n) + aT\left(\frac{n}{b}\right)$$

$$= f(n) + a\left(f\left(\frac{n}{b}\right) + aT\left(\frac{n/b}{b}\right)\right)$$

$$= f(n) + af\left(\frac{n}{b}\right) + a^2 T\left(\frac{n}{b^2}\right)$$

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

$$\vdots$$

$$= \sum_{i=0}^{k-1} a^i f\left(\frac{n}{b^i}\right) + a^k T\left(\frac{n}{b^k}\right)$$

Stop recurring when

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

$$\therefore \quad n = b^K$$

$$\therefore \quad \boxed{\log_b n = K}$$

Then $T\left(\frac{n}{b^K}\right) = T(1)$ and

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

Thus

$$T(n) = \sum_{i=0}^{K-1} a^i f\left(\frac{n}{b^i}\right) + T(1) \cdot n^{\log_b a}$$

we see $T(n) \geq \text{Const.}\, n^{\log_b a} = \Omega(n^{\log_b a})$

Hence $T(n) = \Omega(n^{\log_b a})$

Also $T(n) = aT(\frac{n}{b}) + f(n) \geq f(n)$, so

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

specialize to case (1): $\exists\, \varepsilon > 0$ s.t.

$$f(n) = O(n^{\log_b a - \varepsilon})$$

so there are pos. c, n_0 s.t.

$$(*) \qquad 0 \leq f(n) \leq c n^{\log_b a - \varepsilon}$$

for all $n \geq n_0$. we re-define $f(n)$ on a finite number of initial terms $n \in \{1, 2, \ldots, n_0 - 1\}$, so that (*) is true for all n.

claim: this does not affect the asymptotic growth rate of $T(n)$ (Proof omitted.)

Thus

$$f\left(\frac{n}{b^i}\right) \leq c\left(\frac{n}{b^i}\right)^{\log_b a - \varepsilon}$$

for all $i \geq 0$.

$$\therefore \quad a^i f\left(\frac{n}{b^i}\right) \leq c a^i \left(\frac{n}{b^i}\right)^{\log_b a - \varepsilon}$$

$$\therefore \quad \sum_{i=0}^{k-1} a^i f\left(\frac{n}{b^i}\right) \leq \sum_{i=0}^{k-1} c \cdot a^i \left(\frac{n}{b^i}\right)^{\log_b a - \varepsilon}$$

$$= c\, n^{\log_b a - \varepsilon} \cdot \sum_{i=0}^{k-1} a^i \left(\frac{1}{b^i}\right)^{\log_b a - \varepsilon}$$

$$= c n^{\log_b a - \varepsilon} \cdot \sum_{i=0}^{k-1} a^i \cdot \frac{(b^i)^\varepsilon}{(b^i)^{\log_b a}}$$

$$= c n^{\log_b a - \varepsilon} \cdot \sum_{i=0}^{k-1} \cancel{a^i} \frac{(b^\varepsilon)^i}{\cancel{(b^{\log_b (a)})^i}}$$

$$= c n^{\log_b a - \varepsilon} \cdot \sum_{i=0}^{k-1} (b^\varepsilon)^i$$

$$= \frac{c \cdot n^{\log_b a}}{n^\varepsilon} \cdot \left(\frac{(b^\varepsilon)^k - 1}{b^\varepsilon - 1}\right)$$

$$= \left(\frac{c}{b^\varepsilon - 1}\right) \cdot \frac{n^{\log_b a}}{n^\varepsilon} \cdot \left((b^\varepsilon)^k - 1\right)$$

$$\leq \text{const} \cdot \frac{n^{\log_b a}}{n^{\varepsilon}} \cdot (b^k)^{\varepsilon}$$

$$= \text{const} \cdot \frac{n^{\log_b a}}{\cancel{n^{\varepsilon}}} \cdot \cancel{\left(b^{\log_b n}\right)^{\varepsilon}}$$

$$= O(n^{\log_b a})$$

$$\therefore \quad \sum_{i=0}^{k-1} a^i f\left(\frac{n}{b^i}\right) = O(n^{\log_b a})$$

$$\therefore \quad T(n) = O(n^{\log_b a})$$

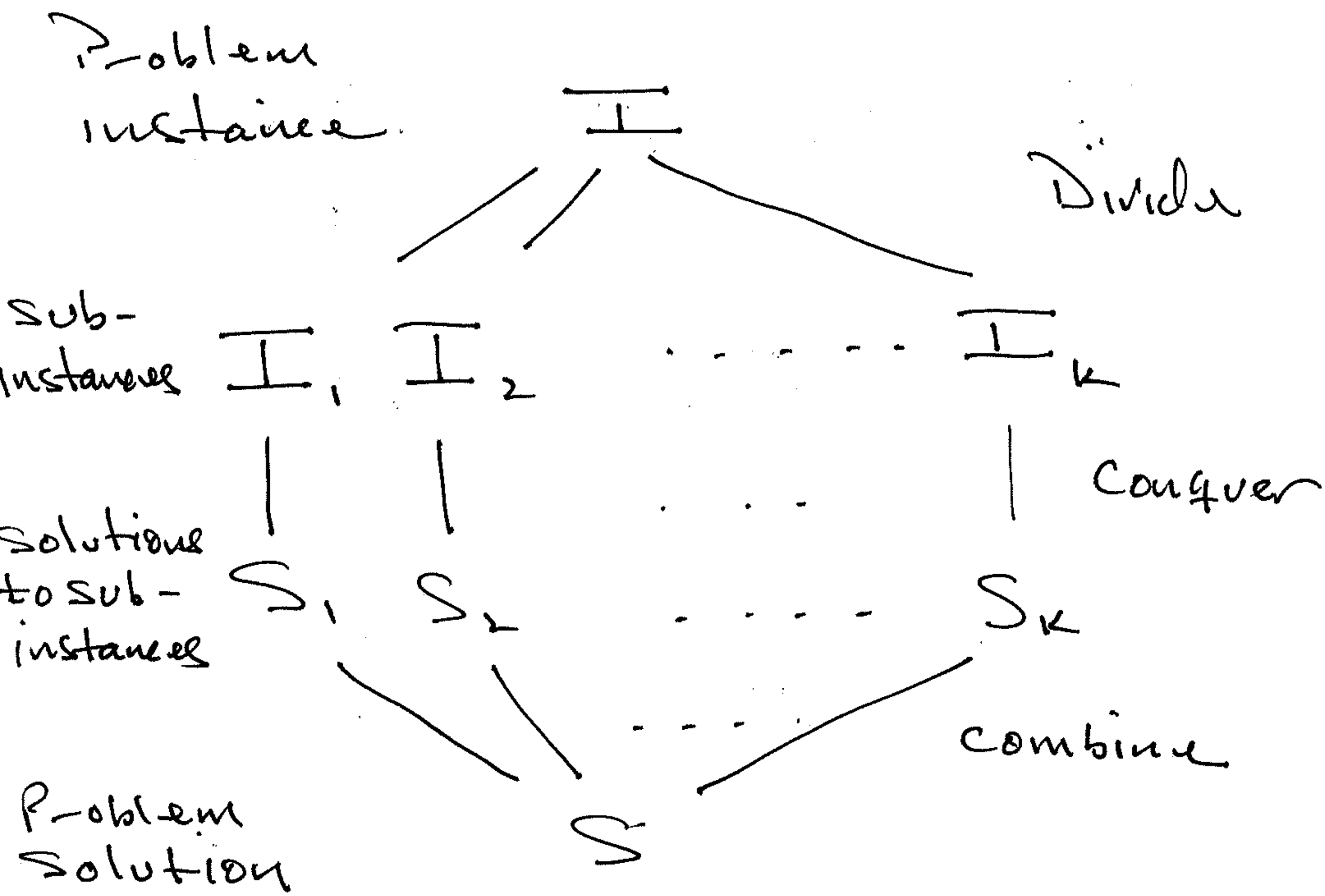
Therefore

$$\boxed{T(n) = \Theta(n^{\log_b a})}$$

Exercise

sketch proofs of cases (2) and (3).

Divide & Conquer Algorithms:



Ex Mergesort

$A[p \cdots\cdots r]$

divide

$q = \left\lfloor \frac{p+r}{2} \right\rfloor$

$A[p \cdots q]$ $\quad$ $A[q+1 \cdots r]$

sort

$A[p \cdots q]$ $\quad$ $A[q+1 \cdots r]$

merge

$A[p \cdots\cdots r]$

Sec. 2.3
p. 31 of
CLRS

MergeSort(A, p, r)

		cost
1.	if $p < r$	0
2.	$q = \left\lfloor \frac{p+r}{2} \right\rfloor$	0
3.	MergeSort(A, p, q)	$T(\lceil \frac{n}{2} \rceil)$
4.	MergeSort(A, q+1, r)	$T(\lfloor \frac{n}{2} \rfloor)$
5.	Merge(A, p, q, r)	$n-1$

<u>Theorem</u>

If Merge(A, p, q, r) is correct, then after MergeSort(A, p, r), the sub-array $A[p \cdots r]$ is sorted.

Proof

use induction on

$$m = \text{length}(A[p \ldots r]) = r - p + 1$$

I. if $m = 1$, then $r - p + 1 = 1$, so $r = p$. The cond. on line 1 is false, so M.S. returns with no changes to $A[\ldots]$. Indeed, $A[p \ldots r]$ has length 1, so is already sorted.

II. let $n > 1$. Assume any call to MergeSort on a sub-array of length k with $1 \leq k < n$, causes that sub-array to be sorted. must show $A[p \ldots r]$ is sorted when algorithm is complete.

Now $n > 1 \Rightarrow r - p + 1 > 1 \Rightarrow r > p$, so lines 2-5 are executed.

line 2 sets $q = \lfloor \frac{p+r}{2} \rfloor$.

This implies that $p \leq q < r$

<u>why?</u>

$$p < r \Rightarrow 2p < p + r \Rightarrow p < \frac{p+r}{2} \Rightarrow p \leq \lfloor \frac{p+r}{2} \rfloor = q$$

$$p < r \Rightarrow p + r < 2r \Rightarrow \frac{p+r}{2} < r \Rightarrow \lfloor \frac{p+r}{2} \rfloor < r$$

Thus

$$\text{length}(A[p \ldots q]) = q - p + 1 < r - p + 1 = m$$

and

$$\text{length}(A[q+1 \ldots r]) = r - (q+1) + 1 = r - q \leq r - p < r - p + 1 = m$$

By the induction hypothesis, Both $A[p \ldots q]$ and $A[q+1 \ldots r]$ are sorted after lines 3 & 4.

By the correctness of Merge(A, p, q, r), we have $A[p \ldots r]$ sorted after line 5.

Run time analysis: $T(n)$ = worst case # of array comparisons on $A[1 \ldots\ldots n]$

$$T(n) = \begin{cases} 0 & n=1 \\ T(\lceil \frac{n}{2} \rceil) + T(\lfloor \frac{n}{2} \rfloor) + (n-1) & n \geq 2 \end{cases}$$

check: if $p=1$, $r=n$, $q = \lfloor \frac{p+r}{2} \rfloor$

then $q-p+1 = \lceil \frac{n}{2} \rceil$ and $r-q = \lfloor \frac{n}{2} \rfloor$.

To apply master thm, first simplify:

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

compare n to $n^{\log_2 2} = n$

$\therefore$ By case 2 we have

$$T(n) = \Theta(n \log n)$$

3-way Merge sort!

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

comp. n to $n^{\log_3 3} = n$.

$\therefore$ $T(n) = \Theta(n \log n)$

by case 2.

Problem: Count inversions in $A[1 \ldots n]$, i.e. pairs (i, j) s.t. $i < j$ and $A[i] > A[j]$

Inversions(A, p, r)

```
if p < r
    q = ⌊(p+r)/2⌋
    a = Inversions(A, p, q)
    b = Inversions(A, q+1, r)
    c = Compare(A, p, q, r)
    return a+b+c
else
    return 0
```

Compare(A, p, q, r)

```
count = 0
for i = p to q
    for j = q+1 to r
        if A[i] > A[j]
            count++
return count
```

$$\text{cost} = \left\lceil \frac{n}{2} \right\rceil \cdot \left\lfloor \frac{n}{2} \right\rfloor = \Theta(n^2)$$

so

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