CMPS 201 –

OUTLINE:

I. MATHEMATICAL PRELIMINARIES
- ASYMPTOTIC GROWTH RATES (CLRS 3)
- INDUCTION PROOFS (HANDOUT)
- RECURRENCE RELATIONS (CLRS 4)

II. DIVIDE & CONQUER ALGORITHMS
- SEARCHING
- SORTING (CLRS 7,8)
- SELECTION (CLRS 9)
- NON-COMPARISON SORTS (CLRS 8)

III. LOWER BOUNDS & COMPUTATIONAL COMPLEXITY
- DECISION TREES
- ADVERSARY ARGUMENTS

IV. DYNAMIC PROGRAMMING (CLRS 15)
- COIN CHANGING PROBLEM
- 0-1 KNAPSACK
- MATRIX CHAIN MULTIPLICATION, APSP

V. GREEDY ALGORITHMS (CLRS 16)
- CONTINUOUS KNAPSACK
- MINIMUM WEIGHT SPANNING TREES.
- MATROIDS

VI. AMORTIZED ANALYSIS. (CLRS 17)
- FIBONACCI HEAPS (CLRS 20)
- DISJOINT SETS. (CLRS 21)

VII. NP-COMPLETENESS (CLRS 34)

- Asymptotic growth of functions
  - All
- Induction Proofs
  - Examples 2, 3, 5
  - Examples 4, 8 (induction trap)
- Recurrences
  - Substitution 1st example only
  - Iteration (no recursion trees) p. 5 Ex.
  - Master Thm, Examples, Proof
- Sec 3.2

3.2 Some Common Functions

Floors & Ceilings

Given $x \in \mathbb{R}$, $\lfloor x \rfloor$ and $\lceil x \rceil$ denote the unique integers satisfying

$$x - 1 < \lfloor x \rfloor \leq x \leq \lceil x \rceil < x + 1$$

Equivalently if $x \in \mathbb{R}$ and $N \in \mathbb{Z}$:

(1) $N = \lfloor x \rfloor$ iff $N \leq x < N + 1$

(2) $N = \lceil x \rceil$ iff $N - 1 < x \leq N$

i.e. $\lfloor x \rfloor$ is the greatest integer less than or equal to x, and $\lceil x \rceil$ is the least integer greater than or equal to x.

Lemma 1

Let $x \in \mathbb{R}$, $a, b \in \mathbb{Z}$. Then

(1) $a \leq x < b$ iff $a \leq \lfloor x \rfloor < b$

(2) $a < x \leq b$ iff $a < \lceil x \rceil \leq b$.

Proof of (1)

(i) $a \leq x \Rightarrow a \leq \lfloor x \rfloor$ since amongst all integers which are less than or equal to x, $\lfloor x \rfloor$ is the greatest.

(ii) $x < b \Rightarrow \lfloor x \rfloor < b$ since $\lfloor x \rfloor \leq x$.

(iii) $a \leq \lfloor x \rfloor \Rightarrow a \leq x$ since $\lfloor x \rfloor \leq x$.

(iv) $\lfloor x \rfloor < b \Rightarrow x < b$ since $b \leq x \Rightarrow b \leq \lfloor x \rfloor$ by (i). ///

LEMMA 2

LET $x \in \mathbb{R}$, $m \in \mathbb{Z}^+$. THEN

(1) $\lfloor \lfloor x \rfloor / m \rfloor = \lfloor x/m \rfloor$

(2) $\lceil \lceil x \rceil / m \rceil = \lceil x/m \rceil$

PROOF OF (1)

LET $N = \lfloor \lfloor x \rfloor / m \rfloor$. THEN

$$N \leq \frac{\lfloor x \rfloor}{m} < N+1$$

$\therefore mN \leq \lfloor x \rfloor < m(N+1)$

$\therefore mN \leq x < m(N+1)$ (BY LEMMA 1.)

$\therefore N \leq x/m < N+1$

$\therefore N = \lfloor x/m \rfloor$. ///

LEMMA 3

LET $a, b, n \in \mathbb{Z}^+$. THEN

(1) $\lfloor \lfloor n/a \rfloor / b \rfloor = \lfloor n/ab \rfloor$

(2) $\lceil \lceil n/a \rceil / b \rceil = \lceil n/ab \rceil$

PROOF OF (1).

SET $x = n/a$ AND $m = b$ IN LEMMA 2. ///

EXERCISE
PROVE PARTS (2) OF LEMMAS 1, 2, & 3.

EXERCISE.
PROVE IF $n \in \mathbb{Z}$, THEN $\lfloor \frac{n}{2} \rfloor + \lceil \frac{n}{2} \rceil = n$.
ALSO SHOW $\lceil n/2 \rceil = \lfloor \frac{n+1}{2} \rfloor$ AND $\lfloor \frac{n}{2} \rfloor = \lceil \frac{n-1}{2} \rceil$.

LOGARITHMS
LET $x, a, b \in \mathbb{R}$, $x > 0$, $a > 1$, $b > 1$. THEN $\log_a(x)$ DENOTES THE EXPONENT ON a WHICH GIVES x. THUS

$$x = a^{\log_a(x)} = \left(b^{\log_b(a)}\right)^{\log_a(x)} = b^{\log_b(a) \cdot \log_a(x)}$$

(*) $\therefore \log_b(x) = \log_b(a) \cdot \log_a(x)$

$\therefore \log_b(n) = \text{CONST} \cdot \log_a(n)$

I.E. ANY TWO log FUNCTIONS DIFFER BY A CONSTANT MULTIPLE, WHENCE

$$\boxed{\log_b(n) = \Theta(\log_a(n))}$$

EQUATION (*) GIVES

$$\log_a(x) = \frac{\log_b(x)}{\log_b(a)} \qquad \therefore \lg(x) = \frac{\ln(x)}{\ln(2)}$$

WHICH SHOWS HOW TO CONVERT FROM ONE log TO ANOTHER.

EQUATION (*) ALSO IMPLIES

$$a^{\log_b(x)} = a^{\log_a(x)\cdot\log_b(a)} = \left(a^{\log_a(x)}\right)^{\log_b(a)} = x^{\log_b(a)}$$

$$\therefore \boxed{a^{\log_b(x)} = x^{\log_b(a)}}$$

STIRLING'S FORMULA

LET $n \in \mathbb{Z}^+$. THEN

$$\boxed{n! = \sqrt{2\pi n}\cdot\left(\frac{n}{e}\right)^n\cdot\left(1+\Theta\left(\frac{1}{n}\right)\right)}$$

COROLLARY

(1) $n! = o(n^n)$

(2) $n! = \omega(2^n)$

(3) $\log(n!) = \Theta(n\log n)$

PROOF OF (1)

$$\frac{n!}{n^n} = \frac{\sqrt{2\pi n}\cdot\left(\frac{n}{e}\right)^n\cdot\left(1+\Theta\left(\frac{1}{n}\right)\right)}{n^n}$$

$$= \frac{\sqrt{2\pi n} \cdot \left(1 + \Theta\left(\frac{1}{n}\right)\right)}{e^n} \to 0 \quad \text{AS } n \to \infty.$$

$\therefore n! = o(n^n)$, AS CLAIMED. ///

PROOF OF (3)

$$\log(n!) = \log\sqrt{2\pi n} + \log\left(\frac{n}{e}\right)^n + \log\left(1 + \Theta\left(\frac{1}{n}\right)\right)$$

$$= \frac{1}{2}\log 2\pi + \frac{1}{2}\log n + n\log n - n\log e + \log\left(1 + \Theta\left(\frac{1}{n}\right)\right)$$

$$\therefore \frac{\log(n!)}{n\log n} = 1 + \left(\begin{array}{l}\text{STUFF WHICH} \to 0 \\ \text{AS } n \to \infty\end{array}\right)$$

$\therefore \log(n!) = \Theta(n \log n)$. ///

EXERCISE: PROVE (2)

EXERCISE: PROVE THAT

$$\binom{2n}{n} = \Theta\left(\frac{4^n}{\sqrt{n}}\right)$$

READ INDUCTION HANDOUT.

READ RECURRENCE HANDOUT

EX. BINARY SEARCH

LET A BE AN ARRAY OF INTEGERS (SAY) WITH $n = \text{length}[A]$. WE WILL ADOPT THE CONVENTION THAT ARRAY INDICES BEGIN AT 1. THUS

$$A[1 \cdots n] = (A[1], \ldots, A[n]).$$

LET $A[p \cdots r]$ DENOTE THE SUBARRAY

$$A[p \cdots r] = (A[p], \ldots, A[r]).$$

IF $p > r$ WE UNDERSTAND THIS TO BE AN EMPTY ARRAY.

ASSUME $A[1 \cdots n]$ IS SORTED IN INCREASING ORDER (WITH POSSIBLE REPEATED ELEMENTS.)

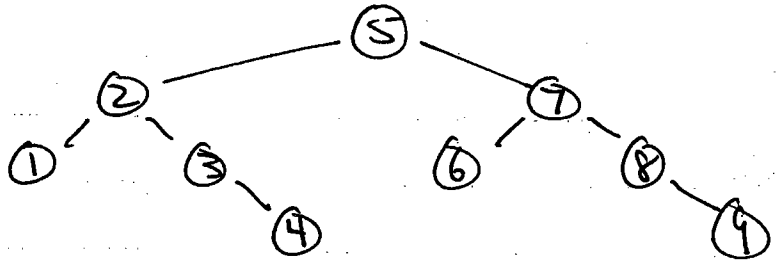
BINARY SEARCH IS A D&C ALGORITHM WHICH LOCATES A GIVEN TARGET t IN THE SUB-ARRAY $A[p..r]$. IF AN INDEX i IS FOUND SUCH THAT $A[i] = t$, THEN i IS RETURNED, OTHERWISE 0 IS RETURNED.

BinSearch (A, p, r, t) (PRE: $A[p..r]$ SORTED.)

1.) if $p > r$
2.) return 0
3.) $q \leftarrow \lfloor \frac{p+r}{2} \rfloor$
4.) if $A[q] = t$
5.) return q
6.) if $A[q] < t$
7.) return BinSearch$(A, q+1, r, t)$
8.) return BinSearch$(A, p, q-1, t)$

Ex. $A = (1, 2, 3, 4, 5, 6, 7, 8, 9)$

1 2 3 4 5 6 7 8 9



THIS BINARY SEARCH TREE REPRESENTS THE ORDER IN WHICH THE TARGET t IS COMPARED TO ELEMENTS OF A.

THE CALL TO BinSearch ON THE FULL ARRAY $A[1..n]$ is JUST

BinSearch $(A, 1, n, t)$

THEOREM (CORRECTNESS OF BinSearch)
BinSearch RETURNS EITHER AN INDEX j SUCH THAT $A[j] = t$, OR RETURNS 0 IF NO SUCH j EXISTS.

PROOF
USE INDUCTION ON $m = r - p + 1 = length[A[p \cdots r]]$.

I. BASE
If $m = 0$ THE SUBARRAY DOES NOT CONTAIN TARGET t. ALSO $m = 0$ IMPLIES $r = p - 1 < p$, SO THAT 0 IS RETURNED ON LINE 2.

II. (STRONG) INDUCTION
LET $m > 0$ AND ASSUME THAT BinSearch RETURNS THE CORRECT INDEX ON ANY SUBARRAY OF LENGTH LESS THAN m.

NOW $m > 0$ IMPLIES $r > p - 1$ WHENCE $r \geq p$. THUS $q = \lfloor \frac{p+r}{2} \rfloor$ (LINE 3) IMPLIES $p \leq q \leq r$.

IF $A[q] = t$ THEN OBVIOUSLY BinSearch RETURNS CORRECT INDEX ON LINE 5.

IF $A[q] < t$ THEN $A[(q+1) \cdots r]$ IS A SUBARRAY OF length

$$r - (q+1) - 1 = r - q \leq r - p < r - p + 1 = m.$$

THE INDUCTION HYPOTHESIS GUARANTEES THAT A CORRECT VALUE IS RETURNED ON LINE 7.

IF ON THE OTHER HAND, $A[q] > t$ THEN $A[p \ldots (q-1)]$ IS OF LENGTH

$$(q-1)-p+1 = q-p \le r-p < r-p+1 = m,$$

SO THE INDUCTION HYPOTHESIS INSURES THAT A CORRECT VALUE IS RETURNED ON LINE 8.

IN ALL CASES A CORRECT VALUE IS RETURNED, AND THE PROOF IS COMPLETE. ///

THE (WORST CASE) ANALYSIS OF BINARY SEARCH IS VERY SIMPLE

$$T(n) = \begin{cases} \Theta(1) & n = 0 \\ 1 \cdot T\left(\frac{n}{2}\right) + \Theta(1) & n \ge 1 \end{cases}$$

$n^{\log_2 1} = n^0 = 1 = \Theta(1)$, SO CASE 2 OF THE MASTER THEOREM SAYS

$$T(n) = \Theta(\lg n).$$

Exact soln to BinSearch

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

$$T(n) = (a+b) + b\lfloor \lg n \rfloor = \Theta(\lg n).$$

Exact soln to MergeSort where $n = 2^k$.

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

$$T(n) = dn \lg n + cn = \Theta(n \lg n).$$

Ex Merge Sort

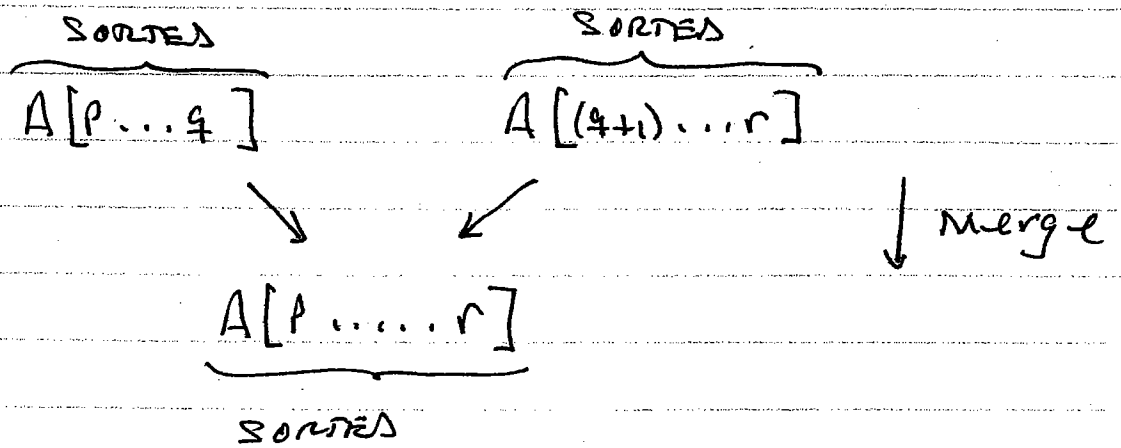
As before A is an array (of numbers) with $n = length[A]$.

MergeSort recursively sorts a subarray $A[p..r]$ of $A[1...n]$ where $1 \le p \le r \le n$.

MergeSort (A, p, r)

1.) if $p < r$
2.) $\quad q \leftarrow \lfloor \frac{p+r}{2} \rfloor$
3.) $\quad$ MergeSort (A, p, q)
4.) $\quad$ MergeSort $(A, q+1, r)$
5.) $\quad$ Merge (A, p, q, r)

Merge (A, p, q, r) is a sub-algorithm which combines the sorted subarrays $A[p..q]$ and $A[(q+1)..r]$ into one sorted subarray $A[p..r]$.



TO SORT THE ENTIRE ARRAY $A[1 \cdots n]$ WE CALL MergeSort$(A, 1, n)$.

THEOREM.
AFTER MergeSort(A, p, r) is CALLED, THE SUB-ARRAY $A[p..r]$ is SORTED.

PROOF.
WE USE INDUCTION ON $m = r - p + 1 = length[A[p..r]]$.

I. IF $m = 1$ THEN $r = p$ AND $A[p..r]$ is ALREADY SORTED. INDEED, THE ALGORITHM DOES NOTHING IN THIS CASE.

II. LET $m > 1$ AND ASSUME THAT ANY CALL TO MergeSort ON A SUBARRAY OF LENGTH LESS THAN m RESULTS IN THAT SUBARRAY BEING SORTED.

NOW $m > 1$ IMPLIES $r > p$, SO LINE 2 is EXECUTED SETTING $q = \lfloor \frac{r+p}{2} \rfloor$. THIS IMPLIES $p \le q < r$:

$$p < r \Rightarrow 2p < p + r \Rightarrow p < \frac{p+r}{2} \Rightarrow p \le \left\lfloor \frac{p+r}{2} \right\rfloor \Rightarrow p \le q < r$$

$$p < r \Rightarrow p + r < 2r \Rightarrow \frac{p+r}{2} < r \Rightarrow \left\lfloor \frac{p+r}{2} \right\rfloor < r \Rightarrow p \le q < r$$

THUS

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

AND

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

THE INDUCTION HYPOTHESIS THEREFORE INSURES THAT $A[p..q]$ AND $A[(q+1)..r]$ ARE SORTED AFTER LINES 3 AND 4 ARE EXECUTED. HENCE $A[p..r]$ IS SORTED AFTER THE EXECUTION OF LINE 5.

THIS COMPLETES THE PROOF. ///

Analysis (worst case)

$$T(n) = \begin{cases} \Theta(1) \\ T(\lfloor \tfrac{n}{2} \rfloor) + T(\lceil \tfrac{n}{2} \rceil) + \Theta(n) \end{cases}$$

THIS CAN BE SIMPLIFIED TO $T(n) = 2T(\tfrac{n}{2}) + \Theta(n)$, AND THE MASTER THEOREM (CASE 2) GIVES

$$T(n) = \Theta(n \lg n).$$

Exact solution to MergeSort when $n = 2^k$

$$T(n) = \begin{cases} c & n = 1 \\ 2T\left(\frac{n}{2}\right) + dn & n \geq 2 \end{cases}$$

$$\begin{aligned} T(n) &= dn + 2T\left(\frac{n}{2}\right) \\ &= dn + 2\left(d\left(\frac{n}{2}\right) + 2T\left(\frac{n/2}{2}\right)\right) \\ &= dn + dn + 2^2 T\left(\frac{n}{2^2}\right) \\ &\vdots \\ &= \sum_{i=1}^{k} dn + 2^k T\left(\frac{n}{2^k}\right) \end{aligned}$$

Pick k s.t. $\frac{n}{2^k} = 1$ (AND $\therefore T\left(\frac{n}{2^k}\right) = c$).

i.e. $n = 2^k$, $k = \lg n$.

THEN $T(n) = dnk - d\sum_{i=1}^{k} i + nc$

$$= dnk - d\frac{k(k+1)}{2} + cn$$

$$= dn\lg n - \frac{1}{2}dn(\lg n + 1) + cn$$

$$\therefore T(n) = dn\lg n + cn - \frac{1}{2}dn\lg n - \frac{1}{2}dn$$

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

Ex. Quicksort. (7.1 - 7.4)

AGAIN $A[p \cdots r]$ IS SORTED RECURSIVELY.

Quicksort(A, p, r)

```
1.) if p < r
2.)     q ← Partition(A, p, r)
3.)     Quicksort(A, p, q-1)
4.)     Quicksort(A, q+1, r)
```

Partition(A, p, r)

```
1.) i ← p-1
2.) for j ← p to (r-1)
3.)     if A[j] ≤ A[r]
4.)         i ← i+1
5.)         A[i] ↔ A[j]
6.) A[i+1] ↔ A[r]
7.) return (i+1)
```

THE SUBROUTINE Partition(A, p, r) RE-ARRANGES $A[p..r]$ INTO TWO (POSSIBLY EMPTY) SUBARRAYS $A[p..(q-1)]$ AND $A[(q+1)..r]$ SUCH THAT

$$\underbrace{A[p \cdots (q-1)]}_{\text{NOT SORTED}} \leq A[q] \leq \underbrace{A[(q+1) \cdots r]}_{\text{NOT SORTED}}$$

$A[q]$: PIVOT

Ex. Partition

i j=p r

8 6 1 3 7 2 5 (4) ← Pivot

i j

8 6 1 3 7 2 5 (4)

i j

8 6 1 3 7 2 5 (4)

i j

1 6 8 3 7 2 5 (4)

i j

1 6 8 3 7 2 5 (4)

i j

1 3 8 6 7 2 5 (4)

i j

1 3 8 6 7 2 5 (4)

i j

1 3 8 6 7 2 5 (4)

i j

1 3 2 6 7 8 5 (4)

i j

1 3 2 6 7 8 5 (4)

i j

1 3 2 6 7 8 5 (4)

(1 3 2) (4) (7 8 5 6)

A[p..(q-1)] ↑ Pivot A[(q+1)..r]