Chapter 3:

"Algorithm"

Algorithm:

A step by step seq procedure to solve a problem is called algorithm.

e.g;

Algorithm sum of two numbers (integers)

1. Take two integers
2. Add them
3. Stop when get answer.

Pseudocode:

An artifical and informal lang. that helps to develop algorithms is called pseudocode.

Properties of Algorithm:

- Input
- Output
- Definiteness
- Correctness
- Finiteness
- Effectiveness
- Generality.

Searching Algorithm:

A algorithm used to search or find one or more than one element from data set is called searching algorithm.

linear search algorithm:

A sequence type of search to compare every step is called linear search algorithm.

e.g: 1, 2, 3, ... 15

procedure linearsearch (x : $\overline{1, 2, 3, \ldots 15}^{a_i}$)

$i := 1$

while ($i \leq 15$ & $x \neq a_i$)

$i := i + 1$

if $i \leq n$ then location $:= i$

else location $:= 0$

return

Binary search algorithm:

An efficient algorithm that searches a sorted list for a desired, or target, element.

Scanned with CamScanner

Explanation

1. Algorithm

"Finding max no. in a finite sequence"

1. let x is a max no.
2. Compare other integer with x
3. If x is greater then ~~*~~ max = x else max = ~~temp~~ other integer
4. Repeat until there is no integer left
5. terminate

"Pseudocode"

Procedure max$(a_1, a_2, \ldots a_n)$

max $:= a_1$

For $i = 2$ to n Or $i = 2$ & $i \le n$

if max $< a_i$, then max $:= a_i$

else max $:= a_1$

return max {max is the largest integer}

Sequential Search:

1. let r first value is max
2. Compare max with other value
3. If max < other value, then other value will be the max value.
4. Repeat until there is no integer left
5. Return max.

Scanned with CamScanner

Procedure linear search ($a_1, a_2, \ldots a_n$)

max = a_1

for ($i = 2$; $i <= n$)

if max < a_i Then max = a_i
$i := i + 1$

else

max := a_1

return max

Binary Search (Finite set)

1, ... 10 Find 5

1. Divide the list into two
2. Compare 5 with both list
3. If 5 > than first list than check the other list else check First list
4. ~~Compar~~ Compare it with center digit & check is it greater or not the leave the other part
5. Repeat until you find it or no number left
6. return search number

Procedure boolean search ($a_1, a_2, \ldots a_n$)

number = 5 $i = 1$ // Start

~~no := a_1~~ $j = n$ // End

~~no~~ = while $i < j$

no = $[(i+j)/2]$

if no > 5 then $i := m + 1$

else $j := m$

if $5 = m$ then loc = 1

return loc

Scanned with CamScanner

Scanned with CamScanner

Sorting:

Putting elements in an order (increasing or decreasing) is called sorting.

two way to sort element

- Bubble sort
- Insertion sort

Bubble sort:

Comparing every element to place it an order is called bubble sorting.

3 4 5 1 2

temp=

1. let x is in order.
2. compare it with other value
3. If x is smaller than other then no swap else swap 4 temp = x+1
5. Repeat until every value is arranged
6. Return order

Procedure bubblesort ($a_1, a_2, \dots a_n$) $n \geq 2$

For $i := 1$ to $n-1$

For $j := 1$ to $n-1$

if $a_j > a_{j+1}$ the interchange a_j with a_{j+1}

$\{a_1, \dots a_n\}$ is in accessing order.

Scanned with CamScanner

Insertion sort :

In each pass of an insertion sort, one or more pieces of data are inserted into their correct location of ordered list.

e.g 3,2,4,1,5

1. Compare the second element with the first one & check is is less than or not
2. If yes swap then & ~~med~~ move to e next element
3. If not move to the element
4. Repeat until every element is sorted.
5. Return sorted in ascending order

Procedure insertionsort $\{a_1, a_2, \ldots a_n\}$ $n \geq 2$

For $j := 2$ to n

$i := 1$

while $(a_j > a_n)$

$i := i + 1$

$m := a_j$

For $k := 0$ to $j-1-1$

$a_{j-k} := a_{j-k-1}$

$a_i := m$

Greedy Algorithm:

Algorithms that make what seems to be the "best" choice at each step is called greedy algorithm.

$n = 5$

a	a_1	a_2	a_3	a_4	a_5
	1	2	3	4	5

Algorithm greedy (a,n)
{

for i=1 to n
{
x = select (a);

if feasible (x) then
{
Solution = Solution + x;
}

}

$n = 6$

a	1	2	3	4	5	6
	i = 1	2	3	4	5	6

(x) α must be integer & less than 4

{ for i=1 to n

x = select(a)

if (a_i % 2 ≠ 0) then
{ x = a_i;
}

if x < 4 then
{ sol = x;
}

Scanned with CamScanner

inged

ces"

cal lin

Big-O notation :

"The notation used to express the upper bound of an algorithm in runnig time is called big-O notation"

⇒ It represent the worst case of an algorithm time complexity.

i-e " The largest amount of time an algorithm can possible take to complete.

" A function $f(n) = O g(n)$, if there exist a value of positive integer n & n_0 of positive constant C; such that

$$f(n) \le C.g(n) \quad \text{for all } n \ge n_0$$

Hence, function $g(n)$ is an upper bound for function $f(n)$, as $g(n)$ grows faster than $f(n)$

e.g: $f(n) = 2n^2 + 5n + 1$

$g(n) = n^2, C = 8$

$f(n) \le c.g(n)$

$2n^2 + 5n + 1 \le 8n^2$

$n=1$, $2+5+1 \le 8$ ✓

$n=2$, true for every n

let $c = 4$

$f(n) \le c.g(n)$

$2(1)^2 + 5(1) + 1 \le \cancel{8}\ 4(1)^2$ ✗

$2(2) + 5(2) + 1 \le 4(2)^2$ ✗

$2(3) + 5(3) + 1 \le 4(3)$ ✓

$f(n) = O.g(n) \ \forall n \ge 3$

for

time complexity:

Determine the approximate number of operations required to solve a problem of size n.

tractable:

A problem the is solvable by a poly-nomial-time algorithm.

intractable:

A problem that cannot solvable by a polynomial-time algorithm. The lowerband is exponential.

x———x———x

"Matrices"

Matrices:

"A rectangular array of elements arranged in rows & columns is called matrices"

Horizontal line shows rows while vertical lines shows columns.

e.g:

$$A = \begin{bmatrix} a_1 & a_3 & b_1 \\ a_2 & a_4 & b_2 \end{bmatrix}$$

"2" rows

"3" columns

Zero-One matrices:

A matrix whose all entries are either 0 or 1 is called zero-one matrices.

e.g $A = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix}$

transpose matrices:

A matrix whose rows become column & column becomes rows is called transpose matrix.

$$A = \begin{bmatrix} 1 & 3 \\ 2 & 4 \end{bmatrix} \Rightarrow A^t = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

Symmetric:

A square matrix is called symmetric if $A = A^t$.

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \Rightarrow A^t = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Bollean product:

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$$

$$A \odot B = \begin{bmatrix} (1 \wedge 1) \vee (0 \wedge 1) & (1 \wedge 1) \vee (0 \wedge 0) \\ (0 \wedge 1) \vee (1 \wedge 1) & (0 \wedge 1) \vee (1 \wedge 0) \end{bmatrix}$$

$$= \begin{bmatrix} 1 \vee 0 & 1 \vee 0 \\ 0 \vee 1 & 0 \vee 0 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$$

Join & meet:

$$A = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 0 \end{bmatrix}$$

Join $A \vee B = \begin{bmatrix} 1\vee 0 & 0\vee 1 & 1\vee 0 \\ 0\vee 1 & 1\vee 1 & 0\vee 0 \end{bmatrix}$

$$= \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix}$$

Meet $A \wedge B = \begin{bmatrix} 1\wedge 0 & 0\wedge 1 & 1\wedge 0 \\ 0\wedge 1 & 1\wedge 1 & 0\wedge 0 \end{bmatrix}$

$$= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

Power:

$A = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$ Find A^3

$$A^2 = A \odot A = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \odot \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} (1\wedge 1)\vee(0\wedge 0) & (1\wedge 0)\vee(0\wedge 0) \\ (0\wedge 1)\vee(0\wedge 0) & (0\wedge 0)\vee(0\wedge 0) \end{bmatrix}$$

$$= \begin{bmatrix} 1\vee 0 & 0\vee 0 \\ 0\vee 0 & 0\vee 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$$

$$A^2 \circ A = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \odot \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} (1\wedge 1)\vee(0\wedge 0) & (1\wedge 0)\vee(0\wedge 0) \\ (0\wedge 1)\vee(0\wedge 0) & (0\wedge 0)\vee(0\wedge 0) \end{bmatrix}$$

$$= \begin{bmatrix} 1\vee 0 & 0\vee 0 \\ 0\vee 0 & 0\vee 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$$

Scanned with CamScanner