Date : / /
Page No.

DAA = Design & Analysis of Algorithum

Algorithum :—

Algorithum is a set of steps to complete a task
[These step should be finite]

For eg :— To make a cup of Tea

* Algorithum :—

step 1 :— Add water & milk to kettle

step 2 :— Boil, add tea leaves, add sugar and serve it in a cup

Technology Used Today :—

GPS [Global Positioning System]

Algorithum	Program
(i) We use Algorithum in "Design phase"	(i) We use program in "Implementation Phase"
(ii) We should have the knowlege about Natural Language	(ii) We should have the knowledge about the language which we are using [Programming language]
(iii) We always Analysis the Algorithum	(iii) We always test the Program [Testing]
(iv) If we are creating Algorithum we should have the domain knowledge	(iv) Programes should have respective Programming language

Date : / /
Page No.

characteristics of an Algorithum :—

(i) Must take an Input
(ii) Must Give some output
(iii) Finiteness :— (Finite No. of steps)
Algorithums terminate after a finite num of steps
(iv) Definiteness :— Each instruction is clear and Unambiguous (Not confusing easy to unde
(v) Effectiveness :— Every instruction must be ba i.e. simple Instruction.

Expectation from an Algorithum :—

(i) Correctness :— Algorithum must produce corres result

(ii) Produced an Incorrect Answer :— Even it 's fail to give an Incorrect answer result all t time still, there is a control on how often it give a wrong result.
Eg Robbin miler primarily test use in RSA Algorithum it does not give correct answe all the time. one out of 2^{50} time it give Incorrect result

(iii) Approximation Algorithum :— Exact solution is not found but near optimal solution can be found out. (Apply to Approximation Proble

(iv) Less Resources Use :— Algorithum should use les Resource (Time & space). Here time is consider

Date : / /
Page No.

to be the primary measure of an efficiency, we are also concerned with how much the respective algorithm involves the computer memory

a. The Actual Running time depends on variety of bagrounds like speed of computer or processor
b. The language in which the algorithm is complemented
c. The compiler Interpretor
d. The skill of a programmer.

Analysis of an Algorithm :—

Algorithm Analysis is an Important part of computational complexity theory which provides theoretical estimation for the required resource of an algorithm to solve a specific computational problem.

It is the determination of the Amount of time and space resource required to execute it.

Why Analysis of Algorithm is Important

i) To Predict the Behaviour of an Algorithm

To Predict the behaviour of the Algorithm without implementing it on specific computer. It is much more convinient to have

Date : / /
Page No.

Simple measures for the efficiency of an Algorithm then to Implement the Algorithm and test the efficiency every time a certain parameter of the underline computer system changes

∴ It is Impossible to predict the exact behaviour of an Algorithm. There are too many influencing factor. The Analysis is thus only are approximation.

Types of Algorithm Analysis :-

1. Best Case :- It define the input that take less time or minimum time. In the Best case we calculate lower Bound of an Algorithm. [minimum running time]

Eg In linear search when the search data is present at the first location of large data then the best case occurs.

2. Worst Case :- It define the Input for which Algorithm take long time or maximum time In worst case we calculate Upper Bound of an Algorithm. [max. running time]

Eg In linear search when search data is not present at all then the worst case occurs.

Date : 12/02/2024
Page No.

Unit = 2

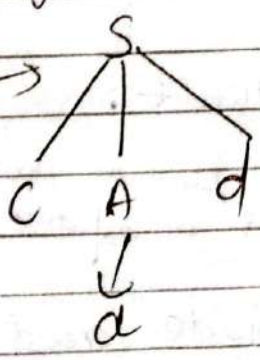
fundamental Algorithmic strategies.

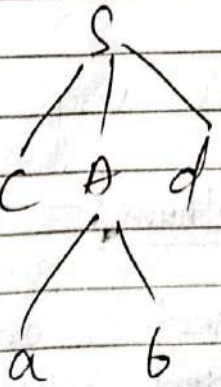
Not Algo.

 # Brute force approach :— Techniques.

- It is an approach that find out all possible solution to find a satisfactory solution to a given problem.
- The brute force approach tries out all the possibilities till a satisfactory solution is not form.
- Whenever a non terminal is expanding first time, then go with the first alternative then compare the input stream. If doesn't match go for the second alternative and compare with input string If doesn't match go with third alternative & continue with each & every alternative.

correct →





S → CAd
A → ab/a

[w = cad]

Date : / /
Page No.

- If the matching occurs for atleast one alternative then parsing is successful

* Advantage :-

1. It findout all possible solutions and it also gurantees that it finds the correct solution to a problem
2. It can be applicable to wide range of domain.
3. It is mainly use for solving simple & smaller problem

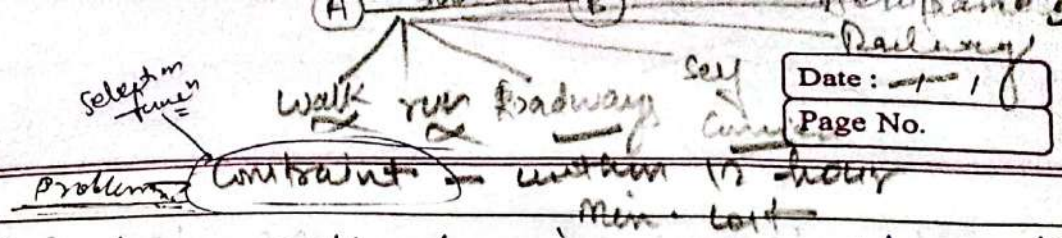
* Disadvantage :-

1. Brut forces required a lot of backtracking & takes Big $O(2^n)$.
2. Backtracking is very costly and reduce the performance of parsed (Break of problem)
3. Debugging is very difficult.

Greedy Algorithum :- It also based on present situation not on future decision

(i) It is based on Divide and Conquer
(ii) It is used for solving optimisation problem (find max. val or min. val)

It depend upon present situation



Greedy method is one of the strategy like dividing conquer use to solve this problem. It is use for solving optimisation problem

* optimisation problem :- It is a problem that demands either max. or minimum value or result.

- This technique find feasible solution that may or may not obtain.

* feasible solution :- Any subset that satisfy the condition

* optimal solution :- Best and most favourable solution.

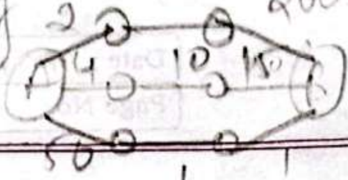
* Characteristic of Greedy Algorithm :-

- To construct the solution in an optimal way this Algorithm maintain two sets
  A - one contain Choosen item
  B - one contain rejected item

- Greedy Algorithm makes good local choices

(i) An optimal solution (ii) feasible solution

Greedy only see first node



Dynamic That see all node to move first

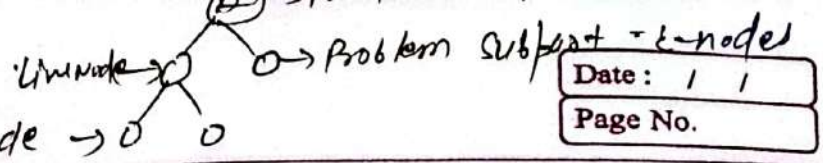
*** Components of Greedy Algorithm :-**

- A Candidate set :- A solution is created from this set
- Selection function :- It is used to choose the best candidate to be added solution.
- A feasibility ~~solution~~ function used for determine whether candidate can be used to contribute to the solution
- An objective function is used to assign a value to solution or a partial solution
- A solution function :- It is used to indicate whether a complete solution has been reached or not.

*** Application for greedy**

1. **Areas of Application :-**

- finding shortest path
- Minimum spanning tree
- Job Scheduling with deadline



Date : / /
Page No.

Basic components of Branch & Bound :-

Generation of a state space tree :-

Branch & Bound generate a state space tree to efficiently search the solution space of a given problem in instant.

In Branch & Bound all childern of an E-node in a state space tree are produced before any live nodes get converted in an E-node. Thus the E-node remains an E-node until it becomes a dead node

Evaluation of a candidate solution :-

Branching Bound needs additional factors to evaluate a candidate solution

A way to assign a best value of the given criterian func^n to each node in a state space tree. it is produced by the addition of further components to the partial solution given by that node.

The best value of given criterian func^n obtained so far it describes

Date : / /
Page No.

the upper bound for maximization problem or lower bound for minimization problem.

(iii) feasible solution :-

- A feasible solution is defined by a problem state that satisfies all the given constraints.

(iv) Optimal solution :-

It is a feasible solution which gives the best value of given objective func^n

(v) Bounding func^n :-

It optimize the search for a solution factor in a solution space of a given problem instant.

(vi) Heuristics func^n :-

It is a heuristics func^n that evaluate a upper & lower bound of a possible solution at each node

Application of Greedy

Date : / /
Page No.

Job Sequencing with deadline :—

Let us take an job $J_1, J_2, J_3 \dots J_N$

deadline — $d_1, d_2, d_3 \dots d_n$

Profits — $P_1, P_2, P_3 \dots P_N$

⇒ If Job complet before deadline the profit is earn.

⇒ An objective is to earn maximum profit when only one job can be schedule and process in any given time.

Index	1	2	3	4
Jobs	A_1	A_2	A_3	A_4
deadlines	2	1	3	1
Profit	40	20	100	50

Index	1	2	3	4
Jobs	A_3	A_4	A_1	A_2
deadline	3	1	2	1
Profit	100	50	40	20

Step-1 Profit Arrange in descending order.

Step-2 $d_{max.} = 3$ (analyzing index number), $i = 1$ → Job = A_3

$K = \min(d_{max}, \text{deadline}[i])$

$= \min(3, 3)$

$K = 3$ ⇒ $K >= 1$

Date : / /
Page No.

Step-3) $i=2$

job $= A_4$

$k = \min(d_{max}, \text{deadline})$

$= \min(3,1)$

$k = 1$

$k >= 1 \rightarrow$ True.

Time slot is empty - Yes.

step-4) $i=3$

job $= A_1$

$k = \min(3,2)$

$k = 2$

$k >= 1$

$2 >= 1 \rightarrow$ True

Time slot is empty → Yes.

3 is because of Max. No. of deadline is 3.

Index	1	2	3
Job	A_3	A_4	A_1
	empty	empty	empty
	A_3	A_4	A_1

$A_3 \rightarrow A_4 \rightarrow A_1$

$A_3 \rightarrow A_4 \rightarrow A_1$

k = Max Profit

$= 100 + 50 + 40$

$= 190.$

Date : / /
Page No.

Step-4 $i = 4$

$Job = A_3$

$k = (3, 3)$

$k = 3$ → True

Step-5 $i = 5$

$Job = A_5$

$k = (3, 4)$

$k = 4$

Knapsack problem :—

(i) we are given an objects and a knapsack or bag.

(ii) object i has ~~weight~~ weight (w_i) & a knapsack has a capacity m.

(iii) If a fraction x_i $0 \le x_i \le 1$ of a object, is placed into an knapsack. then a profit of $p_i x_i$ is earned the objective is to obtained a max. profit

Q1

object	obj 1	ob2	ob3
weight	18	15	10
Profit	25	24	15

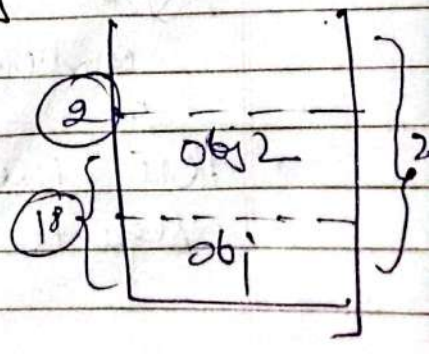
(1) Greedy about Profit :—

see only profit

$25 + \frac{24}{15} \times 2$

$25 + 3.2$

$= 28.2$

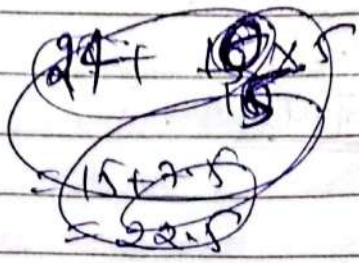


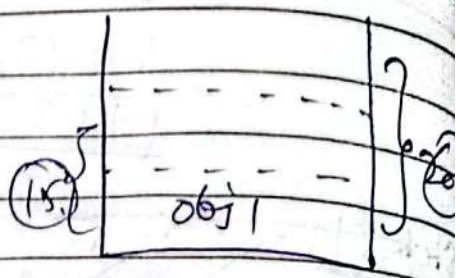
Date : / /
Page No.

see only weight

Greedy about weight

24 + 15×5/10
24 + 2.5
= 32.5





Greedy about profit by weight ratio.

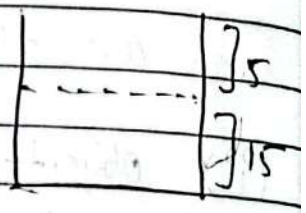
P/W 25/18 : 24/15 : 15/10

= 1.38 : 1.6 : 1.5

= 24 + 5×15/10

= 24 + 7.5

= 32.5



Huffman coding :-

(or) It is data compression technique

- Reduces the size of data file mem
- To the transfer the data once network data is compress so that the size of the data reduces an crosseng cost reduces
- Here we consider the data to be sequence of characters.

Date : / /
Page No.

- Haffmann greedy Algorithm used a table of frequency of the operands of the characters to built up an optical way of representation each character as the binary slip..

- Number of bit required to encode a file $\sum_{c \to} f_i d_i$ where f is frequency count of character d are distance from root to character i.

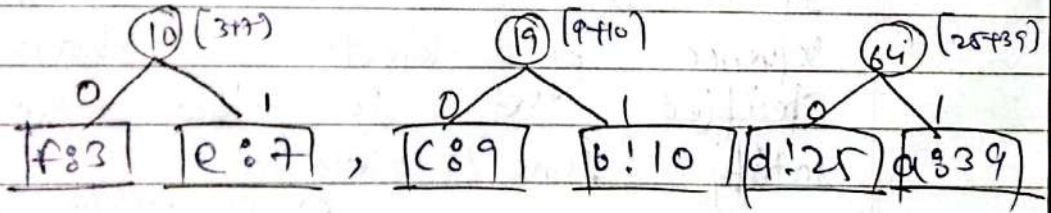
Ex ⇒ a:39, b:10, c:9, d:25, e:7, f:3.
characters change in binary form.

Soln|

Step 1| Change in Ascending form

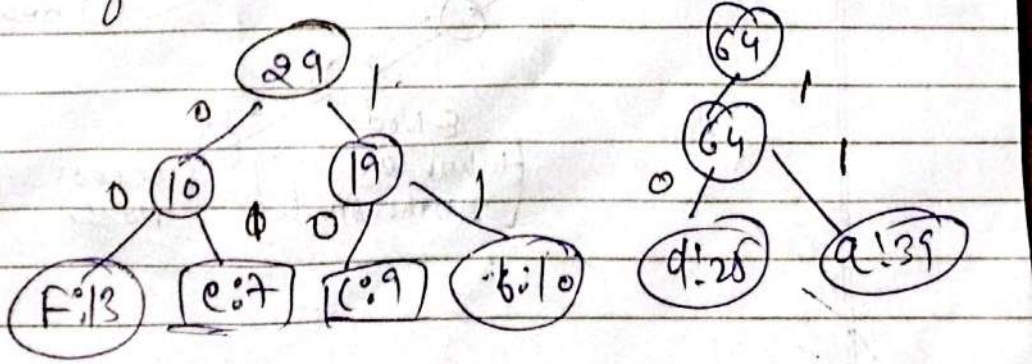
[f:3], [e:7], [c:9], [b:10], [d:25], [a:39]

Step 2|



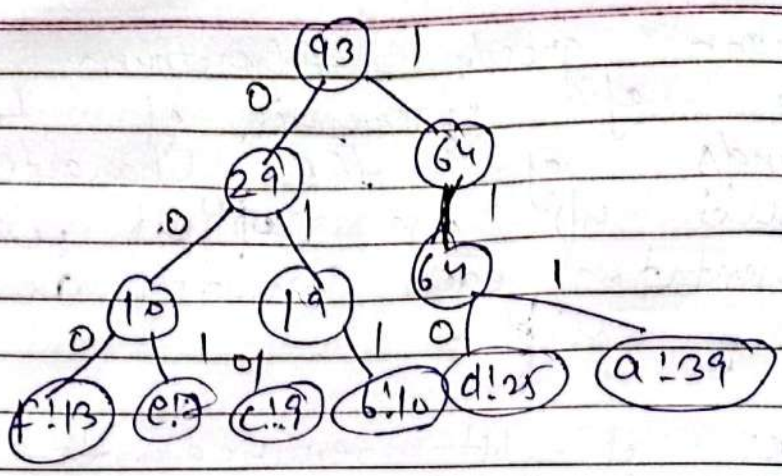
show left hand side with the zero
" right " " " 1.

step 3|



Date : / /
Page No.

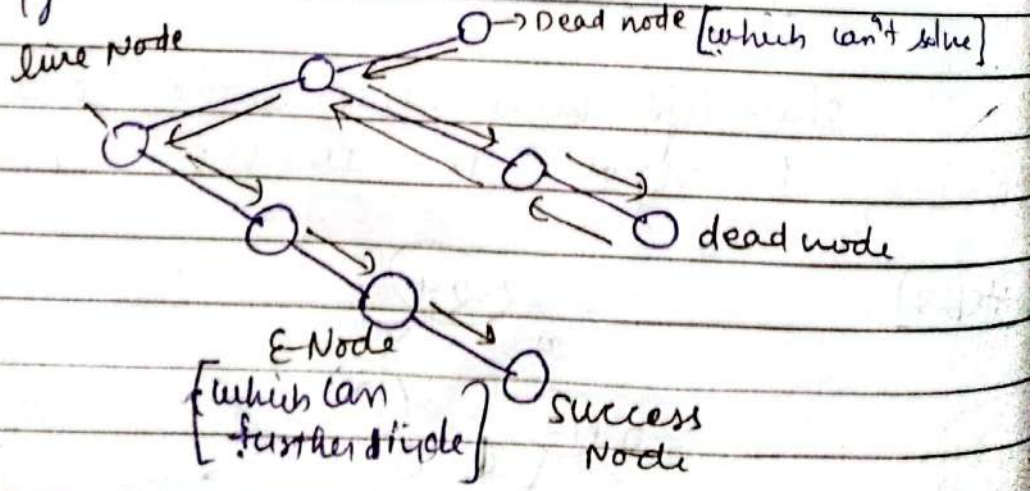
Step-4



a - 111
b - 011
c - 010
d - 110
e - 001
f - 000

Backtracking :-

It is use to solve problem in ~~this~~ the sequence of object is choosen for specified set so that the sequence satisfy some criteria.



Date : / /
Page No.

When to use backtracking :-

- when we have different choices then we use backtracking
- A sufficient information is not available on best choice
- Each Decision ~~leads~~ leads to new set of choice

How to use backtracking :-

Backtracking is a systematic method of trying out various sequence of decision until you find out that work.

2 Types of Constraints/criteria in Back tracking :-

1. Implicit criteria :-
   It is a rule in which how each element is related

2. Explicit criteria :-
   It is a rule that restrict each element to be choosen from the given set

Application of Backtracking :-

- ~~N-queen~~ N-queen problem
- Subset problem
- Hamitonial cycle

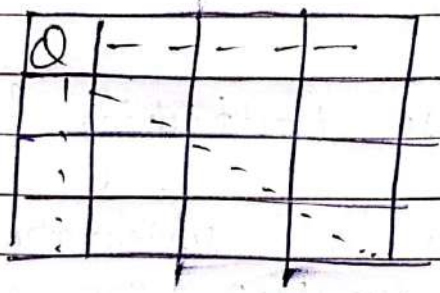
Date : / /
Page No.

1. N-queen Problem :-

- N-queen Problem of placing an chess queen on a N x N chess board so that no two queen attack each other.

- N x N 4 x 4

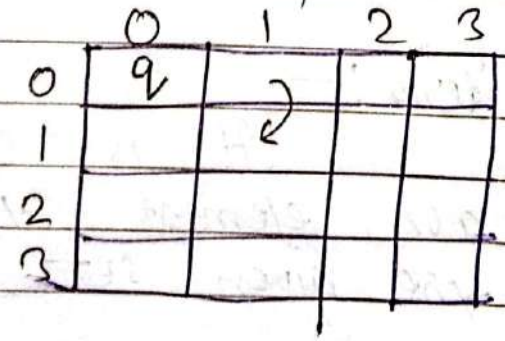
It can be 4x4
8x8



- We have to check that No two queen are place in same row, column and diagonal

* 4-queen Problem :-

Let us take a 4x4 chess board



board = [R] [C]
Row column.