Tutorial-7

Greedy algorithm paradigm - It builds up a solution piece by piece, always choosing the next piece that offers most obvious and immediate benefit. So the problems were choosing locally optimal also leads to global solution are best fit for greedy.

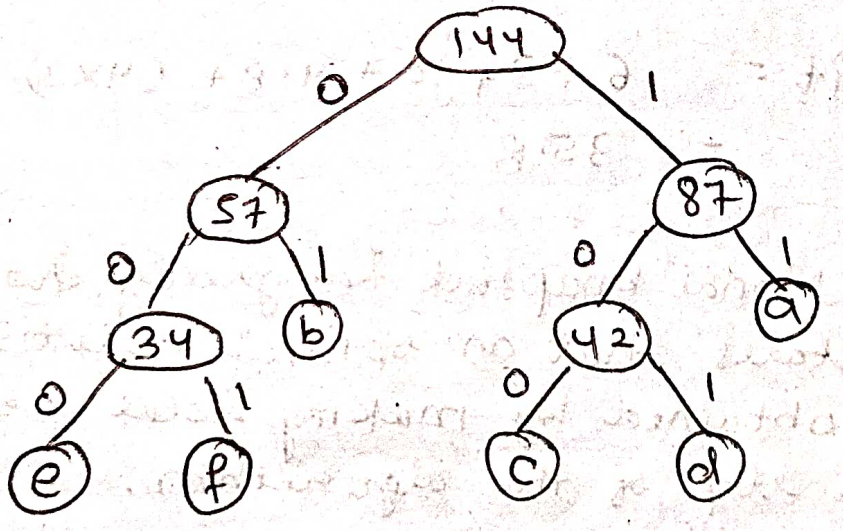
eg. Fractional knapsack

	Time Complexity	Space complexity
Activity Selection	O(n)	O(1)
Job sequencing	$O(n^2)$	O(n)
Fractional knapsack	O(n log n)	O(n)
Huffman encoding	O(n log n)	O(n)

a	b	c	d	e	f
45	23	22	20	19	15

= 144

a → 45
b → 23
c → 22
d → 20
e → 19
f → 15
ef → 34
cd → 42
efb → 57
cda → 87



a = 11 ⇒ 2×45 = 90
b = 01 ⇒ 2×23 = 46
c = 100 ⇒ 3×22 = 66
d = 101 ⇒ 3×20 = 60
e = 000 ⇒ 3×19 = 57
f = 011 ⇒ 3×15 = 45

Total = 364

Average = $\frac{364}{144}$

= 2.54

④ Full binary tree is used while implementing Huffman Encoding.

Application of Huffman Encoding

1) They are used for transmitting fax and text.
2) They are used by conventional compression formats like PKZIP, GZIP etc.

⑤

index	v	w	v/w
1	10	2	5
2	5	3	1.6
3	15	5	3
4	7	7	1
5	6	1	6
6	18	4	4.5
7	3	1	3

sorted

w	v	v/w
1	6	6
1	3	3
2	10	5
3	5	1.6
4	18	4.5
5	15	3
7	7	1

$K = 15$, $(15-1-1-2-3-4-4) = 0$

Profit = $6 + 3 + 10 + 4.8 + (4 \times 3)$

$= 35.8$

⑥ Fractional knapsack has greedy choice property which states that an optimal solution to a problem can be obtained by making local best choices at each step of the algorithm.

Now my proof assumes that there's an optimal solution to the fractional knapsack problem that does not include a greedy choice and then tries to reach a contradiction.

Proof: Assume there's an optimal solution $A = \{a_1, a_2, \ldots a_n\}$ to the problem (E) that does not include item i, with greatest value per weight (V/W) ratio of all initial items.
Suppose a_1 is the item in the solution A with the greatest value per weight ratio,

$$\frac{V_i}{w_i} \geq \frac{V_{a_1}}{W_{a_1}}$$

Now suppose we remove a_1 from A and we obtain a soln A' to sub-problem F'

$$A' = A - a_1$$

If we combine soln A' with greedy choice i, we will obtain a greater or equal valuable soln B, since

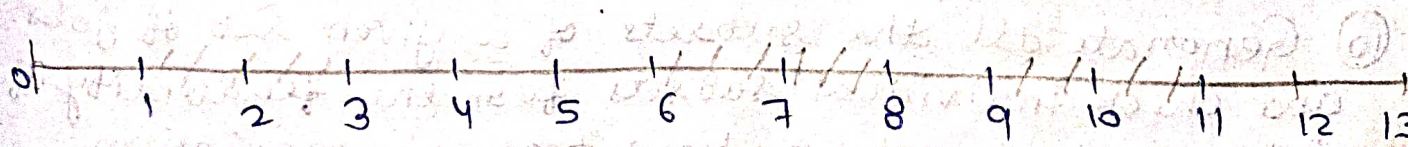
$$\frac{V_i}{w_i} \geq \frac{V_{a_1}}{W_{a_1}}$$

If B is greater than A, then this is a contradiction and thus i must be included; if B=A, then we have shown that the greedy choice is included anyway, since that would mean that

$$\frac{V_i}{w_i} = \frac{V_{a_1}}{W_{a_1}}$$

(7)

Start	1	2	0	6	9	10	
End	3	5	7	8	11	12	



(7)

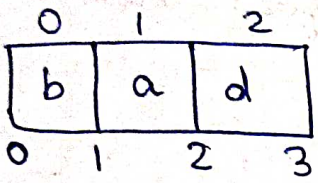
Start Time	1	2	0	6	9	10
End Time	3	5	7	8	11	12

(8)

JOB ID	Profit	Deadline
a	20	2
b	15	2
c	10	1
d	5	3
e	1	3

JOB ID	Profit	Deadline
a	20	2
b	15	2
c	10	1
d	5	3
e	1	3

X



20+15+5 = 40
Profit = 40

(9) In Dijkastra's algorithm greedy approach doesn't works in graphs with negative edges
Similarly we can't break objects in knapsack problem (0-1), the solution that we obtain when using greedy can be pretty bad. We can always make algorithm fail badly.
In travelling salesman problem, we can greedily approach the problem by always going to the nearest possible city. We select any of the cities as the first one and apply that strategy.

(10) Generate all the subsets of a given set of jobs and check individual subsets for the feasibility of jobs in that subset. Keep track of maximum profit among all feasible subsets.

Algorithm

1) Sort all jobs in decreasing order of profit.
2) Iterate on jobs in decreasing order of profit for each job do, the following.
   a) Find a time slot i, such that slot is empty and i < deadline and i is greatest. Put the job in the slot and mark this slot filled.
   b) If no such job exists, then ignore job.