The above problem can be solved by "Huffman's Algorithm.

Step 1- Initialize n one-~~node~~ tree, label them with the characters of the alphabet. Record the frequency of each character in its tree's root to indicate the tree's weight.

2. Repete the following operation until a single tree is obtained. Find two tree with the smallest weight. Make them the left and right subtree of a new tree and record the sum of their weights in the root of the new tree as its weight.

The tree constructed by the above algorithm is called a "Huffman tree".

9

Char	A	B	C	D	—
Prob/freq.	0.35	0.1	0.2	0.2	0.15

A	C	D	—	B
0.35	0.2	0.2	0.15	0.1

Final code

Char	A	B	C	D	—
Prob/freq	0.35	0.1	0.2	0.2	0.15
Code	11	100	00	01	101

Interval Scheduling Proof

S: array of start time

f: array of finish time

n: # of activities.

A solution set

k:- index of last taken activity in A

Greedy (S, f, n)

Sort activities in both arrays by finish time in ascending order. (n log n)

Add a[0] to A

k = 0

$\Theta(n)$

for i = 1 to n-1

if S[i] $\geq$ f[k]

Add a[i] to A

k = i

Body of the loop is constant

Total = $\Theta(n \log n) + \Theta(n)$

$= \Theta(n \log n)$

10

Given a set of activities $S = \{a_1, a_2, a_3 \cdots a_n\}$

$S_{opt} = \{O_1, O_2, O_3 -- O_m\}$

Assume that the activities in both S and S_{opt} ordered by finish time in increasing order

$\boxed{O_1 \text{ and } a_1 \text{ are not necessarily equal}}$

→ If $O_1 == a_1$, we are done

→ if $O_1 \neq a_1$, we can replace O_1 with a_1 and still get a valid solution, why?

i	1	2 (O_1)	3	4	5	6	7	8	9
s_i	2	2	4	1	5	8	9	11	13
f_i	3	5	7	8	9	10	11	14	16
d_i	1	3	3	7	4	2	2	3	3
duration	✓	✗	✓	✗	✗	✓	✗	✓	✗

Let suppose we have a solution starts with $\{a_2, a_5, a_7, a_8\}$

---

arer get $S_{opt}' = \{a_1, O_2, O_3 -- O_m\}$

we know that start time of $S(O_i) \geq finish(O_1)$

for $i = 2, 3, -- m$

Sience $f_{a_1} \leq f_{o_1}$ $s_{o_i} \geq f_{a_1}$ for all $i = 2, 3, -- m$

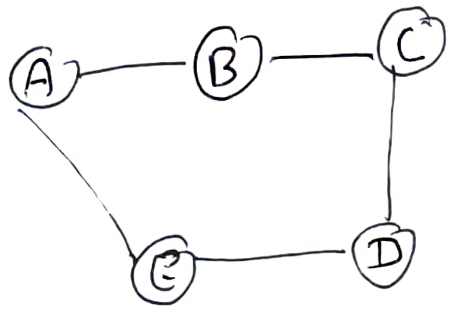
~~This implies that~~

This proves that there always exists an optimal solution that starts with the greedy choice (activity with min finish time)

(Greedy choice property)

1. Greedy choice property → There exists at least one optimal solution that starts with greedy choice.
2. optimal substructure
&
we can keep repeating

Logest path



L.P (B, D) = <B, A, E, D>
≠
L.P (B, A) = <B, C, D, E>
+
L.P (A, D) = <A, B, C, D>

This does not satisfy the optimal substructure subproblem.

$O_1 \quad O_2 \quad O_3 \quad - - - \quad O_m$

$a_1 \quad a_2 \quad a_3 \quad a_4 \quad - - \quad - - - \quad a_n$

"cut & paste"