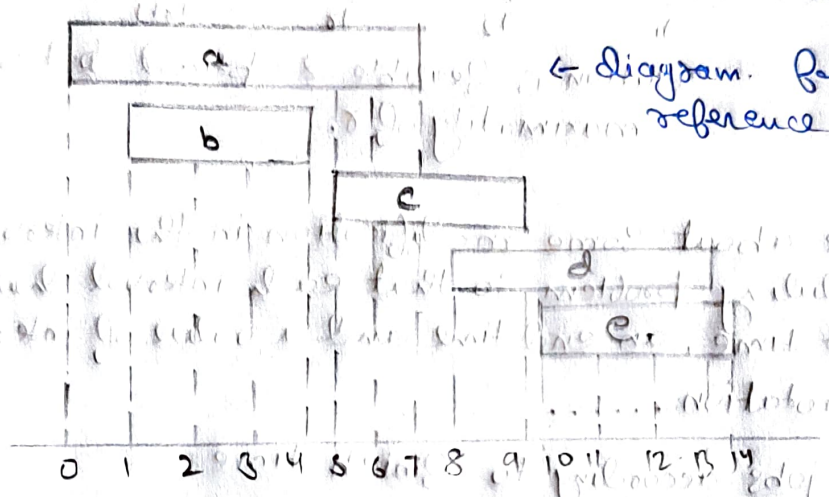
→ Aditya Prakash Swain
→ 21410194 19. → CSIT-E
→ Algorithm Design.

Assignment-4

(1) In interval scheduling problem, we always choose the earliest finishing interval a(s,f), where 's' is starting time and 'f' is finish time. Assume that instead of always.....

Sol!



← diagram for reference.

here we choose the last interval to start that is compatible with every interval that we have already chosen. So basically, we are doing scheduling from right to left and taking starting times as finishing times.

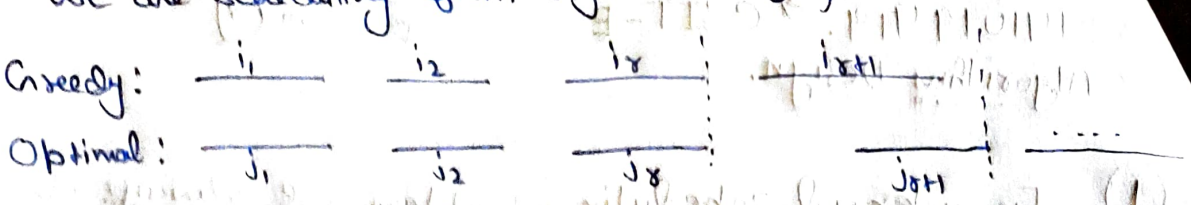
Greedy algorithm: Consider jobs in decreasing order of start time. Take each job provided it's compatible with ones already taken.

To show: Greedy algorithm is optimal.

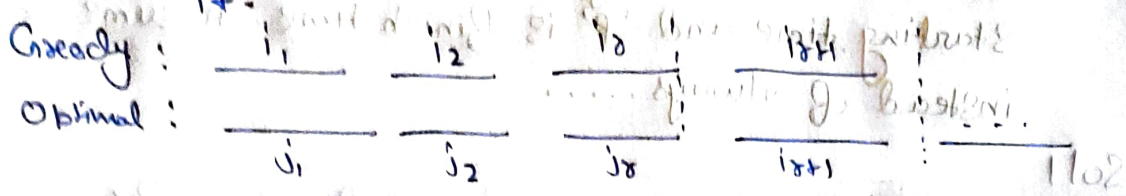
Proof: (by contradiction)

- Assume greedy is not optimal, & let's see what happens.
- Let $i_1, i_2, \ldots i_k$ denotes set of jobs selected by greedy.
- Let $j_1, j_2, \ldots j_m$ denotes set of jobs in optimal soln (with $j_1 = i_1, j_2 = j_2, \ldots i_r = j_r$ for the largest possible value of r.

(here we took ~~from~~ starting times as finish times as we are scheduling from right to left).



i_{r+1} finishes before j_{r+1} so replace j_{r+1} with it.



Solution is feasible & optimal but contradicts maximality of r.

(2) Think about some modification in the interval scheduling problem so that each interval has a start time, an end time and a value of interval. In notation.....

Sol?

- Sort jobs according to finish time.
- Find the latest job before the current job (in sorted array) that doesn't conflict with current job 'arr[n-1]'. Once we find such a job, we recur for all jobs till that job & add profit ~~of profit~~ of current job to result.
- We do above ~~for~~ process for all the jobs. arr[n-1] to arr[0]. and take the maximum result obtained.

Time-complexity $\rightarrow O(n^2)$

(3) Let a cache with a capacity of storing 5 data items containing (a, b, c, d) initially. If memory requests are coming in order a, c, f, d, e, c, g, b, h, f, a, d, c, b, e How many cache miss will occur.....

(i) Farthest-in-Future Scheduling. (FIF)

(ii) Last In First Out. (LIFO)

(iii) Least recently used. (LRU)

Sol3

(i)

a	b	c	d	—
a	b	c	d	f
a	b	c	e	f
a	b	c	g	f
a	b	c	h	f
a	b	c	d	f
a	b	c	d	e

There are 6 misses in Farthest-in-Future Scheduling.

(ii)

a	b	c	d	—
a	b	c	d	f
a	b	c	d	e
a	b	c	d	g
a	b	c	d	h
a	b	c	d	f
a	b	c	d	e

There are 6 misses in Last In First Out.

(iii)

a	b	c	d	—
a	b	c	d	f
a	e	c	d	f
g	e	c	d	f
g	e	f	d	b
g	e	c	h	b
g	f	c	h	b
g	f	a	h	b
d	f	a	h	b
d	f	a	h	c
d	f	a	b	c
d	e	a	b	c

There are 11 misses in Least Recently used.

(4) The time required, to access the cache is 100 nano seconds & that of main memory is 1 micro second. Let the cache has a capacity of 3 & contain (a, b, c) initially. The memory.....

order - a, d, c, f, d, b, g, a, e, c, b, f, a, d, g.

4.

Optimal Schedule.

d	b	c
d	b	f
g	b	f
a	b	f
e	b	f
c	b	f
c	b	a
c	d	a
g	d	a

9 misses

LIFO

a	b	d
a	b	c
a	b	f
a	b	d
a	b	g
a	b	e
a	b	c
a	b	f
a	b	d
a	b	g

10 misses

least recent

a	d	c
f	d	c
f	d	b
g	d	b
g	a	b
g	a	e
c	a	e
c	b	e
c	b	f
a	b	f
a	d	f
a	d	g.

12 misses

(5) Let $A = \{a, b, c, d, e\}$ be a set of independent letters with their probabilities. $p(a) = 1/16$, $p(b) = 1/2$, $p(c) = 1/16$, $p(d) = 1/8$, $p(e) = 1/4$. Give a huffman code for these letters.

letters	a	b	c	d	e
frequency	1	8	1	2	4

a = 1110
b = 0
c = 1111
d = 110
e = 10

(6) Optimal Caching : Suppose cache size $K = 3$ & Sequences of request are {b, a, c, a, d, c, f, a, c, d, g, f, e, a, c, b}. Let assume cache already contain only {a, c} request.

cache initially → a c

By farthest-in-future,

a	c	b
a	c	d
a	c	f
a	d	f
a	g	f
a	e	f
a	e	c
a	b	c

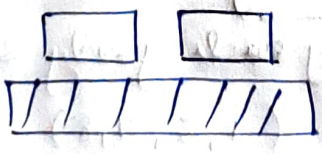
we got 8 misses by using "farthest-in-future" method.

(7) Give complete mathematical formulation of Interval Scheduling problem to maximize the number of mutually compatible jobs.....

- Job j starts at s_j and finishes at f_j
- Two jobs are compatible if they don't overlap.
- Goal: find maximum subset of mutually compatible jobs.

Greedy choices →

(i) [earliest-start-time] - consider jobs in ascending order of s_j.



← (counter-example)

hence, fails.

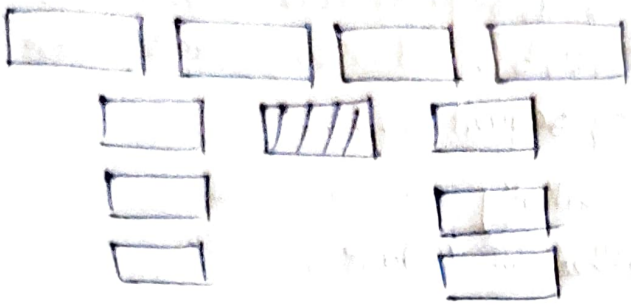
(ii) [Shortest-interval] - consider jobs in ascending order of $f_j - s_j$.

← (counter-example)

hence, fails

(iii) [Fewest-conflicts] - for each job j, count no. of

conflicting jobs c_j. Scheduling in ascending order c_j.



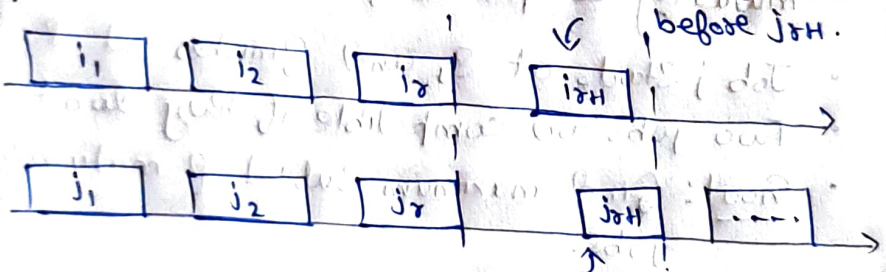
← (Counter example) hence, fails — o

(iv) [earliest-finish-time] - Considers the jobs in ascending order of f_j.

Checking optimality using contradictions. -

- Assume greedy is not optimal, & lets see whats happen.
- Let $i_1, i_2, i_3, \ldots i_k$ denote set of jobs selected by greedy.
- Let $j_1, j_2, j_3, \ldots j_m$ denote the set of jobs selected in optimal solution with $i_1 = j_1$, $i_2 = j_2$, ... $i_r = j_r$ for largest possible value of r.

Greedy:



OPT:

why not replace job j_{r+1} with job i_{r+1}

Then, we will obtain a feasible & optimal solution but it will contradict our maximality of r. Hence, greedy is optimal.

— o

No. of compatible jobs by each greedy strategies -

- [earliest-start-time] - 4 (1, 5, 7, 8)
- [Shortest-interval] - 4 (4, 2, 8, 7)
- [fewest-conflict] - 4 (1, 8, 7, 2)
- [earliest-finish-time] - 4 (4, 2, 7, 8)

— o

(8) Give a ~~Compatible~~ complete mathematical formulation of the Scheduling problem to minimize the lateness of the Schedule. Identify the input, output.....

Soln

- Single job processes one job at a time.
- Job j requires t_j units of processing time & is due at time d_j.
- If j starts at time s_j, it finishes at time $f_j = s_j + t_j$
- Lateness = $l_j = \max\{0, f_j - d_j\}$.
- Goal: Schedule all jobs to minimize maximum lateness $L = \max l_j$.

Greedy Choices -

(i) [Shortest-processing-time-first] → Consider jobs in ascending order of processing time t_j.

	1	2
t_j	1	10
d_j	100	10

← (Counter example) hence, fails

(ii) [Smallest slack] → Considers jobs in ascending order of slack $d_j - t_j$.

	1	2
t_j	1	10
d_j	2	10

← (Counter example) hence, fails.

(iii) [Earliest-deadline-first] → Considers jobs in ascending order of deadline d_j.

From the fact that there is an optimal schedule having no inversions and no idle time combined with the facts that all schedules with no inversions and no idle time have the same maximum lateness & that our Schedule has no inversions and no idle time, we can conclude that our schedule is optimal.

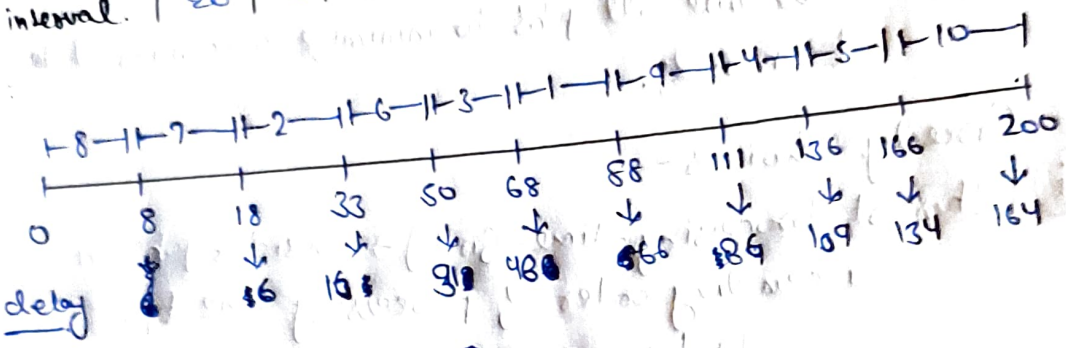
Maximum lateness for each greedy choice -

- [Earliest deadline first] - 98
- [Shortest-processing-time-first] -
- [Shortest-slack-time-first] -

(9) Its examination time and all the s... hurry to prepare study materials for the semester... There is a server rush at all....

Sol 9

Students	1	2	3	4	5	6	7	8	9	10
No of pages.	200	150	180	250	300	170	100	80	230	340
delay time.	22	17	20	27	32	19	12	10	25	36
interval	20	15	18	25	30	17	10	8	23	34

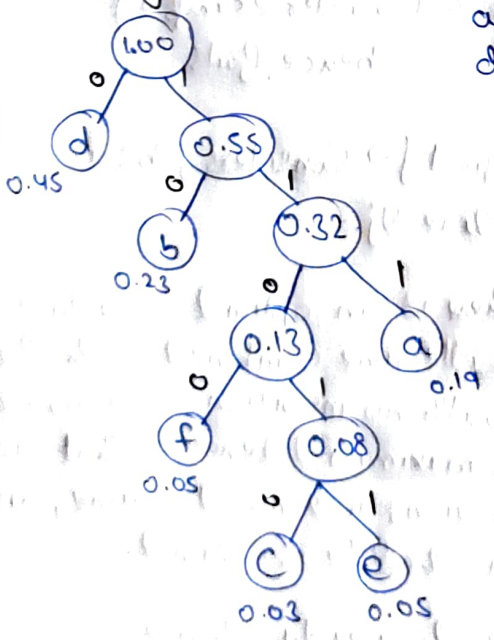


actual price = ₹2000
discount = ₹1320
net income = 2000 − 1320 = ₹680

(10) The frequencies of the characters present in a Symbol set are given as 0.19, 0.23, 0.03, 0.45, 0.05, & 0.05 ...

Sol 10

a = 0.19, b = 0.23, c = 0.03
d = 0.45, e = 0.05, f = 0.05



a = 111
b = 10
c = 11010
d = 0
e = 11011
f = 1100

bit size = 2.08 (with compression)

(ii) Average ~~bits~~ bits per character = $\frac{1}{6}$

= 0.16

For storing 6 symbols we need 3 bits atmost.

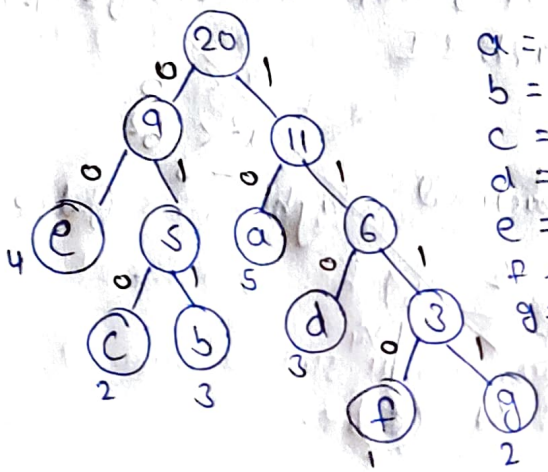
% Saved bits = $\left| \frac{2.08 - 3}{3} \times 100 \right|$

= 30.66 %

—o

(11) Find the Huffman's code to encode the string "a bbaccbaadddeefgggeea" and find the encoded message.....

a	b	c	d	e	f	g
5	3	2	3	4	1	2



a = 10
b = 011
c = 010
d = 110
e = 00
f = 1110
g = 1111

For storing 7 characters we need 3 bits.

Character.	Size.	% change (compressed).
a	2	-33.33%
b	3	0
c	3	0
d	3	0
e	2	-33.33%
f	4	33.33%
g	4	33.33%

(12) Exercise : question no1, chapter - 4...

The given statement is true, as making a spanning tree which is minimal we will required least weight. Since e^* is cheapest it will be prioritize first while drawing minimal spanning tree by any algorithm.

(13) Exercise : question no 2, chapter 4.....

(a) Since all edges are positive & distinct. there square will also be positive & distinct, also we are calculating for positive. So, $a, b \in R^+$

let. $a < b$

on squaring both sides,

$a^2 < b^2 \quad \forall a, b \in R^+$

—o

thus, T will still be minimal.

hence true.

—o

(b) true, P must still be minimum-cost s-t path for new instance. On squaring, ie. weights. let $a, b \in R^+$ &

$a < b$.

squaring both sides.

$a^2 < b^2 \quad \forall a, b \in R^+$

—o

So, P will still remain least.

(14) Exercise : question 13, chapter 4.....