Task / Job Scheduling Algorithm

- A Task Scheduling Problem is the problem of optimally scheduling unit time takes on a single processors, where each task has a deadline

- For the solution of this we can apply greedy strategy

- In a task scheduling problem a "unit - time task" is a job, run on a computer.

Example

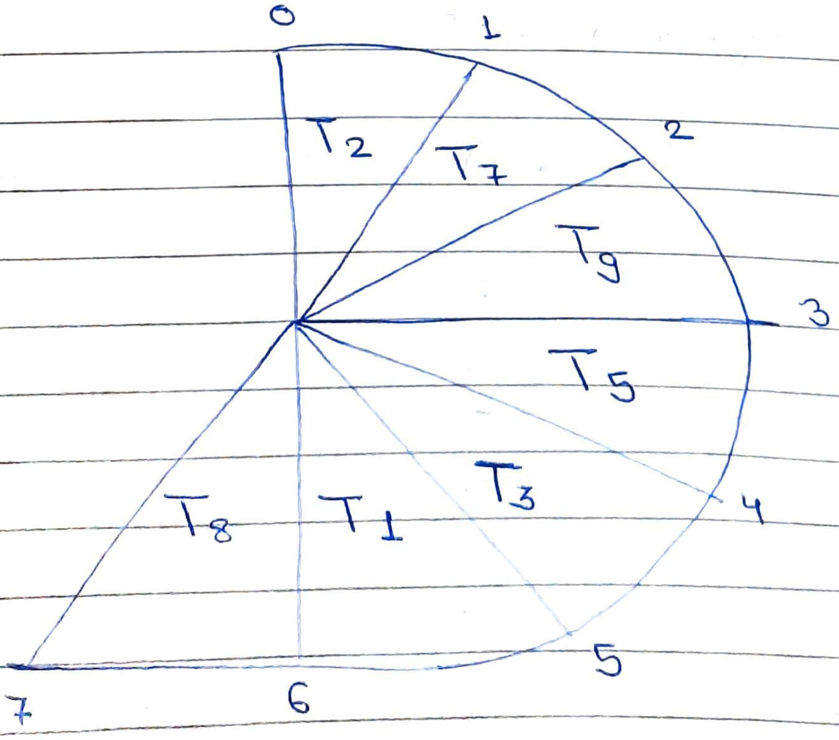
Let n = 9, (T_1, T_2, T_3, - - - T_9)
Deadlines (7, 2, 5, 3, 4, 5, 2, 7, 3)
Profit (15, 20, 30, 18, 18, 10, 23, 16, 25)
Find the optimal soln

Tasks	Deadlines	Profit
T_1	7	15
T_2	2	20
T_3	5	30
T_4	3	18
T_5	4	18
T_6	5	10
T_7	2	23
T_8	7	16
T_9	3	25

-> Now arrange the table in decreasing o
of their profit.

Tasks	Deadline	Profit
T_3	5	30
T_9	3	25
T_7	2	23
T_2	2	20
T_4	3	18
T_5	4	18
T_8	7	16
T_1	7	15
T_6	5	10

Now we will allocate the time as per their deadlines to the task



→ First we will select task T_3
The deadline is 5.

→ Next we select task T_9 with deadline 3.

→ Next task T_7 is selected with deadline 2

→ Now task T_2 is selected.
Task T_2 is having deadline 2 and 1-2 Slot is already allocated to T_7.

Thus, T_2 will be allocated from 0-1

→ Now task T_4 is having deadline 3.
But as shown in fig all the time slots from 0-3 are occupied.

Thus, we will eliminate task T_4

→ Now task T_5 is selected with deadline 4.

→ Now task T_8 is selected with deadline 7.

→ Now task T_1 is selected with deadline 7. But (6-7) time slot is already occupied.
Thus, T_1 will be give Slot (5-6)

- Now all the time slots are occupied thus Task no. 6 i.e. T_6 will be eliminated

Selected Tasks

Task	Deadline	Profit
T_3	5	30
T_9	3	25
T_7	2	23
T_2	2	20
T_5	4	18
T_8	7	16
T_1	7	15
		$\boxed{147}$

Thus, total profit incurred is 147

Activity Selection Problem

Points to remember (Steps)

- For this algorithm we have a list of activities with their starting time and finishing time.

- The goal is to select maximum number of non-conflicting activities that can be performed by a person or a machine, assuming that the person or machine involved can work on a single activity at a time.

- Any two activities are said to be non-conflicting if starting time of one activity is greater than or equal to the finishing time of the other activity

- In order to solve this problem we first sort the activities as per their finishing time in ascending order.

- Then non-conflicting activities are selected.

- Now consider the following e.g. with start and finish time of ~~fo~~ 8 activities.

Activity	Start	Finish
a_1	1	3
a_2	0	4
a_3	1	2
a_4	4	6
a_5	2	9
a_6	5	8
a_7	3	5
a_8	4	5

- Now we will find the non-conflicting activities

1. Sort the activities as per finishing time in ascending order
2. Select the first activity
3. Select new activity if its starting time is greater than or equal to the previously selected activity
4. Repeat step 3 till all activities are checked

→

Sorted Activity	Start	Finish
a_3	1	2
a_1	1	3
a_2	0	4
a_7	3	5
a_8	4	5
a_4	4	6
a_6	5	8
a_5	2	9

→ Select first activity

Selected Activity	-	a_3
Start	-	1
finish	-	2

→ Select next activity whose start time is greater than or equal to the finish time of the previously selected activity.

	Sorted Activity	Start	Finish
i →	a_3	1	2
j →	a_1	1	3
	a_2	0	4
	a_7	3	5
	a_8	4	5
	a_4	4	6
	a_6	5	8
	a_5	2	9

- previously selected activity = i
neoct activity = j

finish time of i = 2
Start time of j = 1

is Start time of j >= finish time of i
$1 >= 2$
No

So, we move to neoct activity

→

	Sorted Activity	Start	finish
i →	a_3	1	2
j →	a_2	0	4

finish time of i = 2
Start time of j = 0

∴ Start time of j >= finish time of i
i.e. $0 >= 2$
No

So we move to neoct activity

	Sorted Activity	Start	Finish
i →	a_3	1	2
j →	a_7	3	5

Start time of j = 3
finish time of i = 2

∴ Start time of j >= finish time of i
i.e. 3 >= 2

∴ we select the activity j

→ Now i will be incremented to newly selected activity. and j will move to the next activity

	Sorted Activity	Start	Finish
i →	a_7	3	5
j →	a_8	4	5

Now, Start time of j = 4
finish time of i = ~~4~~ 5

∴ Start time of j $\ngeq$ finish time of i

∴ we move to next activity

	Sorted Activity	Start	Finish
i →	a_7	3	5
j →	a_4	4	6

Now start time of j = 4
finish time of i = 5

∴ Start time of j $\ngeq$ finish time of i

So, we move to next activity

Sorted Activity	Start	Finish
i -> a_7	3	5
j -> a_6	5	8

Start time of $j = 4$
finish time of $i = 5$

∴ Start time of j ≥ finish time of i
∴ we select the activity j

-> Now i will be incremented to newly selected activity and j will move to the next activity

Sorted Activity	Start	Finish
i -> a_6	5	8
j -> a_5	2	9

Start time of $j = 2$
finish time of $i = 8$

∴ Start time of j ≱ finish time of i
∴ Thus, this activity won't be selected

Thus, the selected activities are as follow

Select Activities	Start	Finish
a_3	1	2
a_7	3	5
a_6	5	8