Job Selection (or) Job Sequencing using Branch and Bound

- Job Sequencing using Branch and Bound: Given n jobs with profit, execution time, and deadline, achieve the schedule which maximizes the profit.
- Each job **i** is represented by the tuple **(Pi, di, ti),** where **Pi, di,** and **ti** represent profit, deadline, and execution time associated with job **i.**
- The brute force method finds 2^n schedules for **n** jobs and finds the best from them. Branch and bound is a more efficient way of finding the optimal schedule.
- Branch and Bound can be used only for minimization problems.
  - Hence we convert maximization problem to minimization problem.
  - We convert profit to penalty, so minimum penalty means maximum profit.
  - Calculate **cost function** for node x in state space tree

$$\hat{C} = \sum_{i \notin s_k} P_i$$

- S_k is the subset of jobs selected at node k.
- Calculate **upper bound**

$$u = \sum_{i \in s} P_i$$

Job	P_i	d_i	t_i
1	5	1	1
2	10	3	2
3	6	2	1
4	3	1	1

- $\hat{C}$ is sum of penalties till the k job considered.
- u is sum of all penalties except that included in solution s.

Upper Bound (UB) = Sum of all penalties except those that are considered

Cost (C) = Sum of penalties till the job considered.

Job selection is to select optimal subset of jobs such that penalty is minimized.

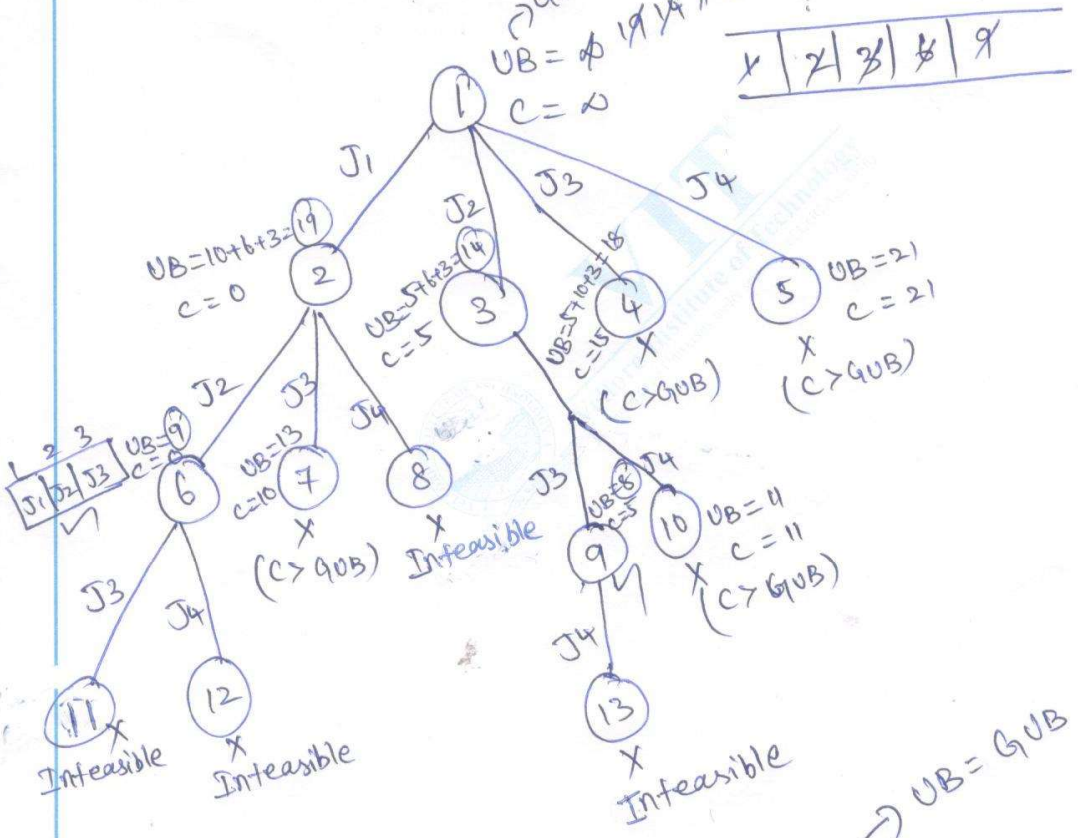
Note:

- Initially Global UB (GUB) = ∞
- If UB < GUB then update GUB
- If C > GUB then kill that node
- Check for feasibility before you calculate UB and C. The total time of jobs considered must be less than max deadline. If greater, then it is infeasible.

Job sequencing using FIFO B&B

FIFO B&B:

Job (i)	1	2	3	4
Penalty (pi)	5	10	6	3
Deadline (di)	1	3	2	1
Time (ti)	1	2	1	1



Valid path for job selection = Node 9 (least UB) → UB = GUB

Jobs done = J2, J3 ⇒ 10+6 = 16 ⇒ Profit

Jobs not done = J1, J4 ⇒ 5+3 = 8 ⇒ penalty

Job sequencing using LIFO B&B