Job Sequencing with deadlines

Greedy method

Job Sequencing

- N jobs $1,2,\ldots,n$ are given where each job needs 1 unit of time to complete.
- Each job i has a profit p_i, which is awarded if the job can be completed before its deadline d_i.
- The objective is to earn maximum profit when only one job can be scheduled or processed at any given time (Assume one machine).

Job Sequencing

	1	2	3	4	5
d	3	2	4	4	2
p	10	60	40	20	70

- Job 1 has a deadline of 3, which means it can be completed in the First time slot or second time slot or third time slot where each time slot is of *1 unit* duration.
- To obtain maximum profit, we greedily choose the *highest profit* job and schedule it in the *farthest empty time* slot available.

Job Sequencing

	1	2	3	4	5
d	3	2	4	4	2
p	10	60	20	40	70

- Sort the jobs in the non-increasing profit order. Record the jobs in the array ***SJ***

	1	2	3	4	5
SJ	5	2	4	3	1

Job Sequencing

	1	2	3	4	5
d	3	2	4	4	2
p	10	60	20	40	70

- Create a time slot array ***TS***, whose size is $\max(d[i])\,, 1 \leq \mathrm{i} \leq \mathrm{n}$
- We place the job ***j*** with the highest profit in the largest *available* slot ***k*** of ***TS*** such that ***k <= d[j]***

	1	2	3	4	5
SJ	5	2	4	3	1

Job Sequencing

	1	2	3	4	5
d	3	2	4	4	2
p	10	60	20	40	70

	1	2	3	4	5
SJ	5	2	4	3	1

	1	2	3	4
TS	0	0	0	0

Profit = 0

Job Sequencing

	1	2	3	4	5
d	3	2	4	4	2
p	10	60	20	40	70

	1	2	3	4	5
SJ	5	2	4	3	1

	1	2	3	4
TS	0	5	0	0

Profit = 70

Job Sequencing

	1	2	3	4	5
d	3	2	4	4	2
p	10	60	20	40	70

	1	2	3	4	5
SJ	5	2	4	3	1

	1	2	3	4
TS	0	5	0	0

Profit = 70

Job Sequencing

	1	2	3	4	5
d	3	2	4	4	2
p	10	60	20	40	70

	1	2	3	4	5
SJ	5	2	4	3	1

	1	2	3	4
TS	2	5	0	0

Profit = 130

Job Sequencing

	1	2	3	4	5
d	3	2	4	4	2
p	10	60	20	40	70

	1	2	3	4	5
SJ	5	2	4	3	1

	1	2	3	4
TS	2	5	0	0

Profit = 130

Job Sequencing

	1	2	3	4	5
d	3	2	4	4	2
p	10	60	20	40	70

	1	2	3	4	5
SJ	5	2	4	3	1

	1	2	3	4
TS	2	5	0	4

Profit = 170

Job Sequencing

	1	2	3	4	5
d	3	2	4	4	2
p	10	60	20	40	70

	1	2	3	4	5
SJ	5	2	4	3	1

	1	2	3	4
TS	2	5	0	4

Profit = 170

Job Sequencing

	1	2	3	4	5
d	3	2	4	4	2
p	10	60	20	40	70

	1	2	3	4	5
SJ	5	2	4	3	1

	1	2	3	4
TS	2	5	3	4

Profit = 190

```
Algorithm JobSequencing(n,d,p)
//n is the number of jobs
//d[1..n] stores deadline of job i in d[i]
//p[1..n] stores profit of job i in p[i]
{
    Create a sorted array SJ where SJ[1..n] stores the
    job ids such that p[SJ[i]]<=p[SJ[j]], i<j
    profit:=0;
    for i:= 1 to n do
    {
        j:=SJ[i];
        for k:= d[j] to 1 step -1 do
        {
            if(TS[k]=0) then
                TS[k]:=j;
                profit:=profit+p[j];
                break;
        }
    }
    return profit;
}
```

```
Algorithm JobSequencing(n,d,p)
//n is the number of jobs
//d[1..n] stores deadline of job i in d[i]
//p[1..n] stores profit of job i in p[i]
{
    Create a sorted array SJ where SJ[1..n] stores the job
    ids such that p[SJ[i]]<=p[SJ[j]], i<j
    profit:=0;
    for i:= 1 to n do
    {
        j:=SJ[i];
        for k:= d[j] to 1 step -1 do
        {
            if(TS[k]=0) then
                TS[k]:=j;
                profit:=profit+p[j];
                break;
        }
    }
    return profit;
}
```

Job Sequencing

- The for loop at **`line 9`** executes for n times.
- The for loop at **`line 12`** executes for a maximum of $d_{\max}$ times.
- The complexity is $O(n * d_{max}) = O(n^2)$
- The complexity is $O(n)$ with disjoint set ADT.