Vashisth Zatakia
60003200163 B3 IT

Aim: Implementation of Job sequencing with Deadlines

Theory:

In job sequencing problem, the objective is to find a sequence of jobs, which is completed within their deadlines and gives maximum profit.

Solution

Let us consider, a set of ***n*** given jobs which are associated with deadlines and profit is earned, if a job is completed by its deadline. These jobs need to be ordered in such a way that there is maximum profit.

It may happen that all of the given jobs may not be completed within their deadlines.

Assume, deadline of **ith** job J_i is d_i and the profit received from this job is p_i. Hence, the optimal solution of this algorithm is a feasible solution with maximum profit.

Thus, D(i)>0 for 1⩽i⩽n.

Initially, these jobs are ordered according to profit, i.e. p1⩾p2⩾p3⩾...⩾pn

Complexity Analysis :

In this algorithm, we are using two loops, one is within another. Hence, the complexity of this algorithm is **$O(n^2)$**

Code:

```
#include <stdio.h>
#include <stdlib.h>
struct Job
{
   int jobid;
   float profit;
   int deadline;
};
int comparator(const void *a, const void *b)
{
   struct Job *j = (struct Job *)a;
   struct Job *j1 = (struct Job *)b;
   return (j1->profit - j->profit);
```

```
}
int minValue(int x, int y)
{
   if (x < y)
      return x;
   return y;
}
int main()
{
   int i, j, n, dmax = 0, counter = 0;
   float total_profit = 0;
   printf("Enter number of processes:");
   scanf("%d", &n);
   struct Job *arr = (struct Job *)malloc(n * sizeof(struct Job));
   int alloc[n];
   for (i = 0; i < n; i++)
   {
      printf("Enter profit and deadine for job %d :", i + 1);
      scanf("%f %d", &arr[i].profit, &arr[i].deadline);
      arr[i].jobid = i + 1;
      alloc[i] = -1;
      if (arr[i].deadline > dmax)
      {
         dmax = arr[i].deadline;
      }
   }
   qsort(arr, n, sizeof(struct Job), comparator);
   for (i = 1; i <= n; i++)
   {
      for (j = minValue(dmax, arr[i - 1].deadline); j >= 1; j--)
      {
```

```
            if (alloc[j] == -1)
            {
                alloc[j] = i - 1;
                counter++;
                total_profit += arr[i - 1].profit;
                break;
            }
        }
        if (counter == dmax)
        {
            break;
        }
    }
    printf("Sequence:");
    for (i = 1; i <= dmax; i++)
    {
        printf("Job %d ", arr[alloc[i]].jobid);
    }
    printf("\nTotal profit= %f", total_profit);
}
```

Vashisth Zatakia
60003200163 B3 IT

Problem

Consider the following 5 jobs and their associated deadline and profit.

index	1	2	3	4	5
JOB	j1	j2	j3	j4	j5
DEADLINE	2	1	3	2	1
PROFIT	60	100	20	40	20

Sort the jobs according to their profit in descending order

Note! If two or more jobs are having the same profit then sort them as per their entry in the job list.

index	1	2	3	4	5
JOB	j2	j1	j4	j3	j5
DEADLINE	1	2	2	3	1
PROFIT	100	60	40	20	20

Find the maximum deadline value

Looking at the jobs we can say the max deadline value is 3.
So, **dmax** = 3

From this set of jobs, first we select J_2, as it can be completed within its deadline and contributes maximum profit.

- Next, J_1 is selected as it gives more profit compared to J_4.
- In the next clock, J_4 cannot be selected as its deadline is over, hence J_3 is selected as it executes within its deadline.
- The job J_5 is discarded as it cannot be executed within its deadline.

Thus, the solution is the sequence of jobs (J_2, J_1, J_3), which are being executed within their deadline and gives maximum profit.

Total profit of this sequence is **100 + 60 + 20 = 180**.

Output:

```
Enter number of processes:5
Enter profit and deadine for job 1 :60 2
Enter profit and deadine for job 2 :100 1
Enter profit and deadine for job 3 :20 3
Enter profit and deadine for job 4 :40 2
Enter profit and deadine for job 5 :20 1
Sequence:Job 2 Job 1 Job 3
Total profit= 180.000000
```