SIDDAGANGA INSTITUTE OF TECHNOLOGY

TUMKUR-572102

(An Autonomous Institute affiliated to VTU Belagavi, Approved by AICTE, New Delhi)

2020-21

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Report on Open ended problem titled

“JOB SEQUENCING WITH DEADLINES”

SUBJECT: ANALYSIS AND DESIGN OF ALGORITHM

Team Members:

Kushala. R	(1SI18CS049)
Meghana. J.P	(1SI18CS056)

Under the Guidance of:

Vedamurthy.H.S

Anupama.B.S

TABLE OF CONTENTS

PROBLEM DESCRIPTION AND SOLUTION

Introduction

The sequencing of jobs on a single processor with deadline constraints is called as Job Sequencing with Deadlines. You are given a set of jobs. Each job has a defined deadline and some profit associated with it. The profit of a job is given only when that job is completed within its deadline.

Problem Statement

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 $\mathbf{i}^{\mathbf{th}}$ job $\boldsymbol{J_i}$ is $\boldsymbol{d_i}$ and the profit received from this job is $\boldsymbol{p_i}$. Hence, the optimal solution of this algorithm is a feasible solution with maximum profit.

Thus, D(i)>0D(i)>0 for 1⩽i⩽n1⩽i⩽n.
Initially, these jobs are ordered according to profit, i.e. p1⩾p2⩾p3⩾...⩾pn.

ALGORITHM

Algorithm: Job-Sequencing-With-Deadline (D, J, n, k)

// Input: No. of Jobs and Array of profits and deadlines of the given Jobs
// Output: Arrange of Jobs in sequence and the total profit

```
D(0) := J(0) := 0
k := 1
J(1) := 1   // means first job is selected
for i = 2 … n do
   r := k
   while D(J(r)) > D(i) and D(J(r)) ≠ r do
      r := r – 1
   if D(J(r)) ≤ D(i) and D(i) > r then
      for l = k … r + 1 by -1 do
         J(l + 1) := J(l)
         J(r + 1) := i
         k := k + 1
```

Steps for performing job sequencing with deadline using greedy approach is as follows:

1. Sort all the jobs based on the profit in an increasing order.
2. Let α be the maximum deadline that will define the size of array.
3. Create a solution array S with d slots.
4. Initialize the content of array S with zero.
5. Check for all jobs.
   a. If scheduling is possible a lot ith slot of array s to job i.
   b. Otherwise look for location (i-1), (i-2)...1.
   c. Schedule the job if possible else reject.
6. Return array S as the answer.
7. End.

Example:

Step-01:

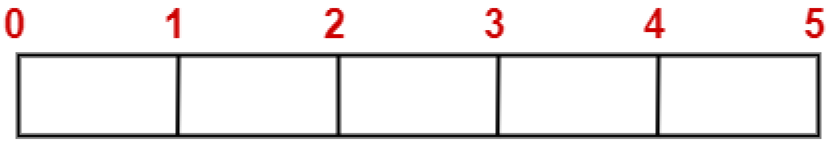
Sort all the given jobs in decreasing order of their profit-

Jobs	**J4**	**J1**	**J3**	**J2**	**J5**	**J6**
Deadlines	2	5	3	3	4	2
Profits	300	200	190	180	120	100

Step-02:

Value of maximum deadline = 5.

So, draw a Gantt chart with maximum time on Gantt chart = 5 units as shown-



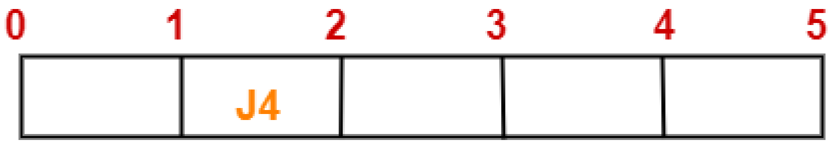
Gantt Chart

Now,

- We take each job one by one in the order they appear in Step-01.
- We place the job on Gantt chart as far as possible from 0.

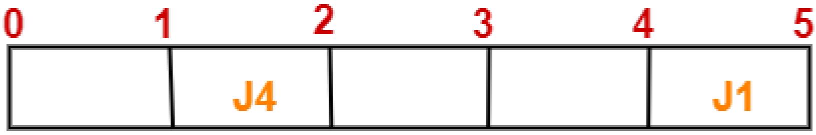
Step-03:

- We take job J4.
- Since its deadline is 2, so we place it in the first empty cell before deadline 2 as-



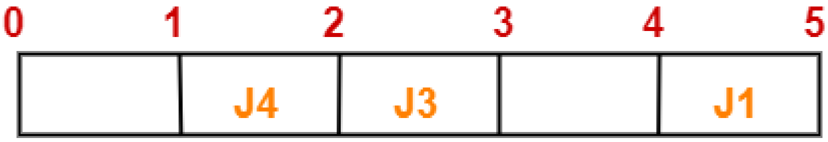
Step-04:

- We take job J1.
- Since its deadline is 5, so we place it in the first empty cell before deadline 5 as-



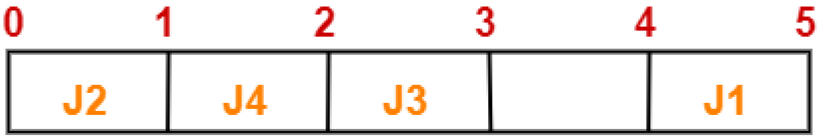
Step-05:

- We take job J3.
- Since its deadline is 3, so we place it in the first empty cell before deadline 3 as-



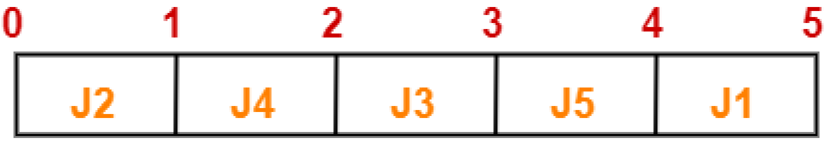
Step-06:

- We take job J2.
- Since its deadline is 3, so we place it in the first empty cell before deadline 3.
- Since the second and third cells are already filled, so we place job J2 in the first cell as-



Step-07:

- Now, we take job J5.
- Since its deadline is 4, so we place it in the first empty cell before deadline 4 as-



Now,

- The only job left is job J6 whose deadline is 2.
- All the slots before deadline 2 are already occupied.
- Thus, job J6 can not be completed.

Now, the given questions may be answered as-

Part-01:

The optimal schedule is-

J2 , J4 , J3 , J5 , J1

This is the required order in which the jobs must be completed in order to obtain the maximum profit.

Part-02:

- All the jobs are not completed in optimal schedule.
- This is because job J6 could not be completed within its deadline.

Part-03:

Maximum earned profit

= Sum of profit of all the jobs in optimal schedule

= Profit of job J2 + Profit of job J4 + Profit of job J3 + Profit of job J5 + Profit of job J1

= 180 + 300 + 190 + 120 + 200

= 990 units

DESIGN TECHNIQUE USED

Greedy choice technique.

ANALYSIS OF THE ALGORITHM

Part 1: Sorting the array profits in descending order using Insertion sort
* It takes nlogn time complexity.

Part 2: Sequencing and Linear Search.
* It takes number_of_jobs*number_of_jobs = number_of_jobs^2 time complexity.

The maximum of the nlogn+number_of_jobs^2 is number_of_jobs^2.

TIME COMPLEXITY OF JOB SEQUENCING ALGORITHM is O(number_of_jobs^2).

APPLICATIONS

- Gives the optimal solution to order jobs for a uniprocessor to get maximum pofit.
- Used in Operating Systems in Job Scheduling.

IMPLEMENTATION

****

Problem Statement : JOB SEQUENCING WITH DEADLINES
Implemented Language : C
****

```
#include<stdio.h>
#include<stdlib.h>

int flag=0;

//Insertion Sort to sort the profits of the given jobs in descending order

void insertion_sort(int profits_sorted[100], int n, int
deadlines[100])
{
    int temp,i,j,a[100];
    for(i=0;i<n;i++)
    {
        j=i;
        while(j>0  &&  profits_sorted[j-
        1]>profits_sorted[j])
        {
            temp = profits_sorted[j];
            profits_sorted[j] = profits_sorted[j-1];
            profits_sorted[j-1] = temp;
            j--;
        }
    }
    j=0;

    for(i=n-1;i>=0;i--)
    {
        a[j++] = profits_sorted[i];
    }
    for(i=0;i<n;i++)
    {
        profits_sorted[i] = a[i];
    }
}
```

//Recursive function to assign the particular job in the job_sequence array to get the optimal sequence

```
void  recursion_job_sequence(int job_number, int n, int
deadlines[100], int job_sequence[100])
{
    if(job_sequence[n]==-1 && n>0)
    {
        job_sequence[n] = job_number;
        flag++;
        return;
    }
    if(n<=0)
    {
        flag++;
        return;
    }
    else
    {
      recursion_job_sequence(job_number, n-1, deadlines,
                                          job_sequence);
    }
}

int main()
{
    int number_of_jobs, total_profit=0, job_profit,
    job_number, i, n, j;
    int profits[100], deadlines[100],
    profits_sorted[100], profits_copy[100],
    job_sequence[100];

    printf("Enter the number of jobs : ");
    scanf("%d",&number_of_jobs);

    printf("Enter the profits and deadlines for all the
    jobs\n");
```

```
for(i=0;i<number_of_jobs;i++)
{
    scanf("%d%d",&profits[i],&deadlines[i]);
}

for(i=0;i<number_of_jobs;i++)
{
    profits_sorted[i] = profits[i];
    profits_copy[i] = profits[i];
}

for(i=1;i<=number_of_jobs;i++)
{
    job_sequence[i] = -1;
}

insertion_sort(profits_sorted, number_of_jobs,
deadlines);

i=0;
while(flag<number_of_jobs)
{
    job_profit = profits_sorted[i];
    for(j=0;j<number_of_jobs;j++)
    {
        if(job_profit == profits_copy[j])
        {
            profits_copy[j]=-1;
            job_number = j;
            break;
        }
    }
    n=deadlines[job_number];
    recursion_job_sequence(job_number, n,
    deadlines, job_sequence);
    i++;
}
printf("******** JOB SEQUENCE ******** \n");
for(i=1;i<=number_of_jobs;i++)
```

```
    {
        if(job_sequence[i]!=-1)
            printf("Job %d\n",job_sequence[i]+1);
    }

    for(i=1;i<=number_of_jobs;i++)
    {
        if(job_sequence[i]!=-1)
            total_profit = total_profit +
            profits[job_sequence[i]];
    }
    printf("TOTAL PROFIT = %d\n",total_profit);
    return 0;
}
```

TEST CASES AND OUTPUT

Test Case :1

```
Enter the number of jobs : 6
Enter the profits and deadlines for all the jobs
30 4
20 2
60 2
30 2
10 1
80 4
******** JOB SEQUENCE ********
Job 4
Job 3
Job 1
Job 6
TOTAL PROFIT = 200
```

Test Case :2

```
Enter the number of jobs : 4
Enter the profits and deadlines for all the jobs
20 4
10 1
40 1
30 1
******** JOB SEQUENCE ********
Job 3
Job 1
TOTAL PROFIT = 60
```

Test Case :3

```
Enter the number of jobs : 5
Enter the profits and deadlines for all the jobs
100 2
19 1
27 2
25 1
15 3
******** JOB SEQUENCE ********
Job 3
Job 1
Job 5
TOTAL PROFIT = 142
```

Test Case :4

```
Enter the number of jobs : 4
Enter the profits and deadlines for all the jobs
50 2
10 1
15 2
25 1
******** JOB SEQUENCE ********
Job 4
Job 1
TOTAL PROFIT = 75
```

Test Case :5

```
Enter the number of jobs : 6
Enter the profits and deadlines for all the jobs
200 5
180 3
190 3
300 2
120 4
100 2
******** JOB SEQUENCE ********
Job 2
Job 4
Job 3
Job 5
Job 1
TOTAL PROFIT = 990
```

Test Case :6

```
Enter the number of jobs : 5
Enter the profits and deadlines for all the jobs
60 2
100 1
20 3
40 2
20 1
******** JOB SEQUENCE ********
Job 2
Job 1
Job 3
TOTAL PROFIT = 180
```

Siddaganga Institute of Technology, Tumakuru

(An Autonomous Institute affiliated to VTU Belagavi, Approved by AICTE, New Delhi)

Department of Computer Science and Engineering

2020-21

Assessment for IV semester Analysis and Design of Algorithm lab open ended problem

Title: JOB SEQUENCING WITH DEADLINES

Sl. No.	Name	USN	Evaluation Criteria				Total (25)	Signature
			Complexity of problem chosen (5)	‘Implementation (10)	Coding Standards followed (5)	Report (5)		
1.	Kushala.R	1SI18CS049						
2.	Meghana.J.P	1SI18CS056						

Signature of Faculty