COMP3121-Ass2-Q5

z5302513, Kihwan Baek

July 2021

Firstly, we sort the array of jobs using 'merge sort' based on the profit in increasing order and it takes $O(n \log n)$ time. After that, let m be the maximum deadline and make an array of length m. And then, we start to check the array of jobs from the beginning and it takes $O(n)$ time.

Assume the first job in the sorted array (which provides the largest profit) as j_1 and put this job in t_1th slot of the array. After that, from the ith job in the array, if the t_i slot of the array is empty, then put the job in the slot. Otherwise, we look for the location to put the job from i-1 to 1 and it takes $O(n-1) = O(n)$ time. If it is not possible to schedule the job (all slots are not available from 1 to i), just reject the job.

Finally, we can get the finished array and we are given a time of length a (a is a integer, $a >= 0$), the sub-array from index 1 to a will be the time schedule which maximizes the total profit. And also, the total time complexity will be $O(n * n) = O(n^2)$.