Job Scheduling Using Greedy Approach

Aditi Goyal (B19EE003)

1 Problem

Given class duration and deadline for n professors, schedule them such that their annoyance is least. We have to develop an algorithm so as to minimise the maximum of the annoyance among all the professors.

2 Algorithm

Algorithm 1: Class Scheduling Using Greedy Approach

```
Input: n class duration and deadline in an array
Define a struct prof with parameters order, duration, deadline to use to store the input in an
  array;
Perform merge sort on the array based on deadline;
Perform merge sort on the array based on durations for the same deadline slots.Use getCount
  function to find number of classes with same deadline to define start and end;
Function MergeSort(prof array[], start, end):
    perform merge sort here for the defined start and end;
    return arr[];
Function getCount(prof array[], start, end):
    while class is in arr[] do
        if class.deadline == deadline then
            count ++;
        else
    end
    return count;
timer=0;
annoyance=0;
for class in array do
    timer+ = class.duration;
    annoyance+ = max(timer − class.deadline, 0);
end
Output: order of classes, total annoyance
```

3 Cost of Algorithm

The time complexity of algorithm is $\mathcal{O}(n \log n)$.
The cost is due to using the merge sort which has an average time complexity of $\mathcal{O}(n \log n)$. The rest of the operations run in linear time ie. $\mathcal{O}(n)$.

4 Proof of Algorithm

The main aim is to minimize the maximum annoyance of individual professors.

Proving our approach of earliest deadline first is optimal by contradiction. Consider an *inversion* to be swapping of the adjacent i and j location elements in a sorted sequence A. First we need to prove that such an *inversion* does not change the annoyance.

Let t_i and t_j be the time duration of a class and d_i and d_j be the deadline.
Assume L_1 be the annoyance before swapping,
$$L_1 = t_i - d_i + t_i + t_j - d_j.$$
Assume L_2 be the annoyance before swapping,
$$L_2 = t_j - d_j + t_i + t_j - d_i.$$

Hence we can clearly see that $L_1 = L_2$, so our initial claim that *inversion* does not change the annoyance holds true.

To prove by contradiction, we assume an optimal sequence A^* which has the fewest of these *inversions* amongst all optimal sequences.
If A^* will have 0 *inversions* then it will be same as A, hence A^* will have atleast 1 *inversion*.
Now if we *invert* adjacent jobs i and j scheduled in A^*, it will not change the annoyance as proved above but it will reduce the number of *inversions* by 1.
This statement is in contradiction to the condition we had imposed on A^* having the fewest inversions amongst all optimal sequences.
In this way we can *invert* all the *inversions* in A^* till there are 0 *inversions*, leading us to A.

Thus proving that, A in which the earliest deadline is scheduled first or so to say is sorted in ascending order, is the optimal solution obtained by greedy approach.