WaitNoMore

Asher Kirshtein

April, 8 2022

1 My Algorithm

My Algorithm is quite simple. I sort based on my ratio going from greatest to smallest in order to optimize the wait time. I simply just have my own private class of jobs where I override the comparable method to sort based on the ratio of weight to time.

The optimal solution must be of at least time $O(NlogN)$ since we are literally sorting the arguments into the most optimal order so it needs a sort which means it will take at least $O(NlogN)$. In other words there is no way to get around the cost of sorting. So both my algorithm and the optimal algorithm are linearithmic. The issue we are solving is how can we order/sort the jobs we have to have the most optimized ordering.

2 Proof

I'm going to be using a Greedy Exchange Argument.

The optimal algorithm has to be $O(NlogN)$ since we are ordering/sorting the array into the most optimized ordering. So since we need to sort; the fastest known comparison based sort is $O(NlogN)$.

Assume that our equation X^* isn't optimal and the optimal solution is X

If $X^* \neq X$ then there must be a $Job_a \geq Job_b$ in terms of ratio size. Since it isn't our algorithm which is descending based on ratio; there must be two ratios next to each other so that one is increasing. Or the ordering was swapped or exchanged at that point in the array.

At that point we were at the same point as the optimal solution. Now $Job_a > Job_b$ by ratio. So in my algorithm Job_a comes first and has a weighted wait time of $(timewaited_a * weight_a)$ which is larger than the Job_b equation $(timeWaited_b * weight_b)$ in the "Optimal" solution we making Job_b come before Job_a.

Since $timewaited_a * weight_a < timeWaited_b * weight_b$ by doing job_b first we are increasing the overall $(weight * wait)$ time making the optimal solution not optimal
QED

3 RunTime

In my algorithm the first price I pay is $O(N)$ to take all of the inputs and put them into a list. I then sort the list which costs me $O(NlogN)$. I then Iterate through again and add all of the wait times together which is another $O(N)$ operation. So overall the time is $O(3NlogN)$ or we can just say that it is linearithmic.