Alexandra Yavnilovitch
Abby Rechkin

Homework #5

a) A counter example for the greedy strategy of sorting by largest l_i first and using the bonus speed on the first k parts of the sorted list is:

$[(60\text{ mi}, 100\text{ mph}), (50\text{ mi}, 10\text{ mph})]$ with $k=1$ and $v=10$ mph

Following the suggested greedy strategy, we would add v to the (60 mi, 100 mph) part. So, our total time would be $\frac{60\text{ mi}}{100\text{ mph}+10\text{ mph}} + \frac{50\text{ mi}}{10\text{ mph}} = 5.545$ hours.

However, adding v to the (50mi, 10 mph) part instead would make the total time $\frac{60\text{ mi}}{100\text{ mph}} + \frac{50\text{ mi}}{10\text{ mph}+10\text{ mph}} = 3.1$ hours.

~~which~~

3.1 hours < 5.545 hours, so the proposed greedy strategy does not work.

b) A counter example for the greedy strategy of sorting by smallest speed limit v_i first and using the bonus speed on the first k parts of the sorted list is:

$(5\text{ mi}, 10\text{ mph}), (100\text{ mi}, 20\text{ mph})$ with $k=1$ and $v=10$ mph

Following the suggested greedy strategy, we would add v to the (5mi, 10 mph) part. So, our total time would be $\frac{5\text{ mi}}{10\text{ mph}+10\text{ mph}} + \frac{100\text{ mi}}{20\text{ mph}} = 5.25$ hours.

However, adding v to the (100 mi, 20 mph) part instead would make the total time $\frac{5\text{ mi}}{10\text{ mph}} + \frac{100\text{ mi}}{20\text{ mph}+10\text{ mph}} = 3.833$ hours.

3.833 hours < 5.25 hours, so the proposed greedy strategy does not work.

Greedy Algorithm

<u>Input</u>: An array L containing the parts of the route. Each part consists of the length l_i and the speed limit v_i. A positive integer K which denotes the number of times the algorithm breaks the speed; $0 \le K \le n$. A positive integer V which is the bonus speed. A positive integer n which is the length of the array.

<u>Output</u>: The minimum travel time

Algorithm

We will sort by largest $\left[\frac{l_i}{v_i} - \frac{l_i}{v_i + V}\right]$ quantity and add the bonus speed V to the first K parts.

* We utilise the type Interval which contains l_i, v_i, and $\frac{l_i}{v_i} - \frac{l_i}{v_i+V}$ in the algorithm.

```
* struct Interval{
        int difference
        int length
        int speedlimit
  }
procedure   minTime(L[(l1,v1),...(ln,vn)], K, V, n){
        time ← 0
        A new array Parts of type Interval that is length n
        i ← 1
        while(i <= n)
            Parts[i].difference = li/vi - li/(vi+V)
            Parts[i].length = li
            Parts[i].speedlimit = vi
            i ← i+1
        sort Parts in descending order by difference
```

```
11        j ← 1
12        while(j <= n)
13            if  j < K
14                time ± ( Parts[j].length / (Parts[j].speedlimit + v) )

15            else
16                time ± ( Parts[j].length / Parts[j].speedlimit )

17        return time
```

Total Time complexity

The first while loop on line 5 performs 3 $O(1)$ operations locally but iterates through all n elements of the input array. This section is $O(n)$.

As discussed extensively in lecture, the sorting operation on line 10 is $O(n \log n)$.

As we execute the last while loop on line 12, the addition operations add $O(1)$ locally while the entire while loop takes $O(n)$ as we iterate through the Parts array.

Then, the resulting time complexity would be $O(n) + O(n \log n) + O(n)$ which is just $O(n \log n)$.

Proof of Greedy Algorithm

We will use an exchange argument to show that our greedy solution which is sorting by largest $\frac{l_i}{v_i} - \frac{l_i}{v_i + v}$ computed and adding the bonus speed to the first k parts is optimal.

Assume there exists an optimal solution S which returns the minimum travel time. We will show that S can be converted into the greedy solution without increasing the total travel time.

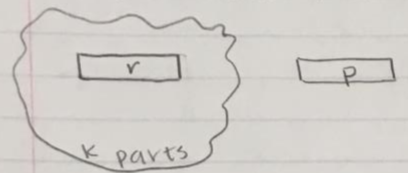
We know that in our greedy algorithm the following condition holds:

① $\frac{l_r}{v_r} - \frac{l_r}{v_r + v} \geq \frac{l_p}{v_p} - \frac{l_p}{v_p + v}$ where r is a part that is included in the k parts where the bonus speed is added and p is a part whose speedlimit isn't adjusted.

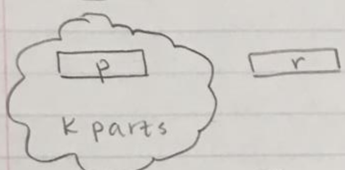
Since, we know that solution S differs from the greedy approach inevitably the above condition will be broken for a part r and a part p in S. This means that the time difference between the time required to travel the interval with and without the added bonus speed for a part r that is included in the k parts has to be less than or equal to the time difference for a part p that is not included.

② $\frac{l_r}{v_r} - \frac{l_r}{v_r + v} \leq \frac{l_p}{v_p} - \frac{l_p}{v_p + v}$

We want to show that swapping a pair of intervals in S which does not follow condition 1 does not increase the total travel time.



We assume that for the initial state of S, the total travel time is To.



We swap intervals r and p to create a new ordering S'. We assume that the total travel time is To'.

$$T_0 = \sum_{j=1}^{k} \frac{l_j}{v_j + v} + \sum_{j=k+1}^{n} \frac{l_j}{v_j + v}$$ where j is the indexing value for the S ordering.

To compute the To' for S', we want to eliminate the times of r and p from To and add their respective times in S'.

$$③ \quad T_0' = T_0 \underbrace{- \frac{l_r}{v_r + v} - \frac{l_p}{v_p}}_{\text{remove old times}} \underbrace{+ \frac{l_r}{v_r} + \frac{l_p}{v_p + v}}_{\text{add in new times}}$$

If we move all of the expressions from the RHS to the LHS of condition 2, we have the following:

$$④ \quad \frac{l_r}{v_r} - \frac{l_r}{v_r + v} - \frac{l_p}{v_p} + \frac{l_p}{v_p + v} \leq 0$$

If we substitute expression 4 into 3, we get the following relation: $T_0' = T_0 + x$ where $x \leq 0$ meaning that $T_0' \leq T_0$. We may conclude that if continue swapping intervals in S that we will eventually arrive at the greedy solution which is at least as optimal as S.