October 15, 2015

CS 577 - Homework 5

Sejal Chauhan, Vinothkumar Siddharth, Mihir Shete

1 Graded written problem

Input: In a city there are n bus drivers. There are also n morning bus routes and n afternoon bus routes, each with various lengths. Each driver is assigned one morning route and one evening route. For any driver, if his total route length for a day exceeds d, he has to be paid overtime for every hour after the first d hours at a fixed rate per hour.

Output: Assign one morning route and one evening route to each bus driver so that the total overtime amount that the city authority has to pay is minimized.

1.1 Algorithm

Our greedy algorithm begins by sorting the n morning bus routes in reverse order (Longest route first) and the n afternoon bus routes in order. To find a set $G\{(m,a) : \forall m \in M \wedge \forall a \in A\}$, where M is set of all morning routes and A is the set of all evening routes, we will use the following *strategy*:

1. Starting from the first morning(largest) route we get the smallest available afternoon route and pair them.

Algorithm 1 Pseudocode of our greedy strategy

```
procedure FIND-ALL-PAIRS(M, A)
    i ← 0
    while i< |M| do
        Pair M[i], A[i]
        i ← i + 1
```

1.2 Exchange Argument

Let r be all the route-pairs in a solution that are greater than d and let l_i be the length of the route-pair . $p(G) = \sum_r (l_i - d)$ for our greedy solution G, similarly for an optimal Solution S $p(S) = \sum_r (l_i - d)$

If, $G = S$ then clearly $p(G) = p(S)$.

Otherwise, $G \neq S$. So, there must be some 2 routes in G and S, such that the morning routes are the same but the paired afternoon routes are different. More formally, if ($M_G(i)$, $A_G(i)$), ($M_G(j)$, $A_G(j)$) are 2 pairs in G and ($M_S(k)$, $A_S(k)$), ($M_S(l)$, $A_S(l)$) are 2 pairs in S, then

$$M_G(i) = M_S(k) \;\;\&\;\; M_G(j) = M_S(l)$$

but,

$$A_G(i) \neq A_S(k) \;\;\&\;\; A_G(j) \neq A_S(l)$$

Let, $M_{ik} = M_G(i) = M_S(k)$ and $M_{jl} = M_G(j) = M_S(l)$ Now, consider the following scenarios in S:

$M_{ik}>M_{jl}$

Let S' denote a solution which would have selected the $(M_{ik}, A_S(l))$ and $(M_{jl}, A_S(k))$ pair contrary to S. S' is similar to G in a way that G would also have chosen the same pairing as S'. So, given the constraint on morning routes S' will choose the afternoon route first for M_{ik} and it will pair with $A_S(l)$ if:

1. $A_S(l)<A_S(k)$ **and** $M_{ik} + A_S(l) \leq d$

   For the above constraints $M_{jl} + A_S(k)$ can be less than, more than or equal to d, so lets analyze all the conditions: Here, if

   $$M_{jl} + A_S(k)>d$$

   it follows that

   $$M_{ik} + A_S(k)>d$$

But,

$$p(S') = C + M_{jl} + A_S(k) - d$$
$$p(S) = C + M_{ik} + A_S(k) - d$$

$\because M_{jl} < M_{ik}$ it implies that $p(S') < p(S)$

$M_{jl} + A_S(k) \leq d$ then the overtime introduced by the swap is 0 and so $p(S') \leq p(S)$

2. $A_S(l) < A_S(k)$ **and** $M_{ik} + A_S(l) > d$
For the above constraints $M_{jl} + A_S(k)$ can be less than, more than or equal to d, so lets analyze all the conditions: Here, if $M_{jl} + A_S(k) > d$ and $M_{jl} + A_S(l) > d$ it is trivial to see that $p(S') = p(S)$ since the extra hours introduced for both the selections will be $M_{ik} + M_{jl} + A_S(k) + A_S(l) - 2$.

   But if $M_{jl} + A_S(k) > d$ and $M_{jl} + A_S(l) \leq d$

$$p(S) = M_{ik} + A_S(k) - d$$
$$p(S') = M_{ik} + A_S(l) + M_{jl} + A_S k - 2d = p(S) + M_{jl} + A_S(l) - d \leq p(S) \because M_{jl} + A_S(l) \leq d$$

   If, $M_{jl} + A_S(k) \leq d$ then it implies that $M_{jl} + A_S(l) \leq d$ from the constraints.

$$p(S') = C + M_{ik} + A_S(l) - d$$

   And

$$p(S) = C + M_{ik} + A_S(k) - d$$

   And since $A_S(l) < A_S(k)$ it is clear that $p(S') < p(S)$

From the above analysis it is clear that $p(S') \leq p(S)$ whenever the order is changed according to greedy strategy. Hence, our greedy strategy will provide an optimal solution.

1.3 Running Time Analysis

To sort get the routes in correct order for our greedy algorithm to process we will have to spend $O(n \log n)$ time. The **Find-All-Pairs** procedure will iterate over all n elements in M and all the operations in these iterations are constant time. So overall runtime is:

$$O(n \log n + c * n) = O(n \log n)$$