CSCI B505 - Fall 2018

Written Assignment 4 - Greedy Algorithms

1. Our goal is to reduce the no. of gas stops before reaching our goal which is New York. Let's say $d_1, d_2, d_3 \cdots d_n$ are the distances of the gas stations from 1 to n along the way before reaching New York.

$d_n - d_{n-1} \leq m$ → (distance the vehicle can travel with full tank)

$D \geq 0$ distance the vehicle can travel without gas breaks

if $d_2 - d_1 < m$

$D = D + d_2 - d_1$

then check the next distance. → Greedy approach.

if $(d_3 - d_2) + (d_2 - d_1) < m$

then $D = D + (d_3 - d_2) + (d_2 - d_1)$

→ if $d_2 - d_1 = m$ or $\{(d_2 - d_1) + (d_3 - d_2)\} \geq m$

then we have to make a gas stop.

So the Algorithm is like -

```
D <= 0
for i in range(1, n):
 if d_{i+1} - d_i < m
   then D = D + d_{i+1} - d_i
        if D + d_3 - d_2 <= m
 else if d_{i+1} - d_i == m
   return D
 else
   return D.
```

This will give us the optimal no. of gas stops as along the way we are stopping only when the vehicle will not be able to make it to the next stop.

2. In this problem we need to maximize the profit by scheduling events which is of 1 minute duration and gives ($g_i > 0$) profit.

First we need to sort the events based on end time.

Algorithm:-

① Schedule the event with maximum profit if 2 schedules of job are overlapping.

② Pick the next best event (profitable) such that start time of new event is greater than or equal to end time of old event.

③ If job are non conflicting then schedule them.

④ If multiple schedules of jobs are overlapping each other, then we will find out the combination of schedules which will maximize the profit, even if the individual profits are not the maximum.

→ Our algorithm will give us optimal solution because it is always maximizes the profit in case of job schedule overlap otherwise choose the best possible job which indeed gives us the maximum profit.

* In the above problem, we assumed that each event starts at t_i time and ends after 1 minute of starting time.

3. Our points are $x_1 \leq x_2 \leq x_3 \ldots \leq x_n$ are in increasing order given on the real line. Our goal is to find the smallest set of unit length closed interval of the form $[n, n+1]$.

We can initialize our set of intervals with null set.

$I = \phi$

$I_1 = [x_1, x_1+1]$ (base case. Taken the 1st ~~element~~ point of the sorted list of points and increased it with 1 for closing cond")

$I = I \cup I_1$

```
for j in range(0,n):
    if x_j in I_0:
        continue
    else:
        I_n = [x_j, x_j+1]
        I = I ∪ I_n
```

(By this we are checking if our point belongs to the already created interval)

(if not belongs to the same interval then we can create a new interval)

This solution will give us the least no. of intervals to which will yield all the numbers ($x_1 \ldots x_n$).

1 Penny = 1¢ = $0.01

1 Nickel = 5¢ = $0.05

1 Dime = 10¢ = $0.10

1 Quarter = 25¢ = $0.25

As per the question we have n cents. Our job is to reduce no. of coins to by exchanging with the above mentioned conversions.

→ As we know 5 pennies becomes 1 Nickel so we can have maximum of 4 pennies. Similarly we can have maximum of 1 Nickel & 2 Dimes.

→ If our collection have more than the number mentioned then we can try to reduce the no. of coins by converting it.

Algorithm :→ quarter, dime, nickel, penny

```
q=0, d=0, n=0, p=0
for i in (0,n):
    (quarter)
    if (n - 25xi > 0):
        q = q+1
    else: break

for i in (0,n):
    d=0 (dime)
for i in (0,n):
    if (n-25xq - ixp > 0):
        d = d+1
    else: break.
for i in (0,n):
    if (n - 25xq - 10xd - ix5 > 0):
        n = n+1
    else: break.
for i in (0,n):
    if (n-25xq-10xd-nx5
        - ix1>0):
        p = p+1
    else: break.
```

Our algorithm will always give optimal solution as we are already converting it to the highest possible number and then trying it for other conversion values.