Ond170030.
Omkar. N. Dixit

CS 6363.004 : Algorithms Spring 2018

Programming Project

* Recurrence for Maximum Profit

$$r(i,K) = \begin{cases} 0 & \text{if } i = 0. \\ \max\limits_{0 \le q_i \le x_i} \left(p_i q_i - \min\left(f_i(n_i - q_i), c\right) + r(i-1, K - w_i q_i) \right) & \text{if } i > 0. \\ & \text{if } n_i > q_i \\ & \text{if } K - w_i q_i \ge 0 \\ \max\limits_{0 \le q_i \le x_i} \left(p_i q_i + r(i-1, K - w_i q_i) \right) & \text{if } p > 0. \\ & \text{if } n_i \le q_i \\ & \text{if } K - w_i q_i \ge 0 \end{cases}$$

i = i^{th} item.
w_i = weight of gold required for the i^{th} item.
p_i = Profit of one quantity of that i^{th} item.
n_i = minimum quantity of i^{th} item required / lower limit
x_i = maximum quantity of i^{th} item / upper limit
f_i = fine for one quantity deficit
c_i = cap for the fine of that item.
K = local gold available.

* Proof of correctness

① Feasibility → There is a Solution $r(i,K)$ that has some profit for i is in range of items. & K is in range Gold

By Induction on $i+K$.

② Optimality → let $opt(i,K)$ be the optimal solution to the problem

Claim → $r(i,K) \ge opt(i,K)$.

we do this by doing Induction on $i+K$.

- **Base Case** $i = 0$.

this will be the zero item or just a dummy item. everything corresponding to $i=0$ & k with any value will be zero.

In this case the optimal solution will be zero.

$\therefore$ $opt(0,k) = 0$ for any value of k $0 \le k \le G_1$.

And,

$\gamma(0,k) = 0$ for any value of k $0 \le k \le G_1$.

$\therefore$ for $i = 0$ & $0 \le k \le G_1$

$\gamma(0,k) = opt(0,k) = 0$.

$\therefore$ Base case true.

- **Step** → $i+k > 0$. or $i+k \neq 0$.

Here we have to show if, for any value of i & k

$\gamma(i,k) \ge opt(i,k)$.

lets say that the opt take q_i copies of an item i^{th}

then

$$opt(i,k) = p_i q_i - \min(f_i(n_i - q_i), c_i) + opt(i-1, k - w_i q_i) \quad \text{——①}.$$

&

$$\gamma(i,k) = \max_{0 \le q_i \le x_i} \left(p_i q_i - \min(f_i(n_i - q_i), c) + \gamma(i-1, k - w_i q_i) \right) \text{ if } k - w_i q \to 0$$

lets make the same cut in our recurrence.

$$\gamma(i,k) \ge p_i q_i - \min(f_i(n_i - q_i), c_i) + \gamma(i-1, k - w_i q_i) \quad \text{——②}.$$

↑ because now we are using the optimal cut

now in $i-1 + K - w_i q_i'$

there will be two cases where.

Case 1 $q_i' = 0$.

$i-1 + K - 0 \quad < \; i+K$

$\therefore$ ~~$\gamma(i, K)$~~ $\gamma(i-1, K - w_i q_i') \geq opt(i-1, K - w_i q_i')$ —— (2)

--- done.

Case 2 $q_i' \neq 0$. or $q_i' > 0$

here as well.

$i-1 + K - w_i q_i' \; < i+K$.

$\therefore \gamma(i-1, K - w_i q_i') \geq opt(i-1, K - w_i q_i')$ —— (4)

---- done.

$\therefore$ now we can say that

$$\gamma(i, K) \geq p_i q_i' - \min(f_i(n_i - q_i'), c) + opt(i-1, K - w_i q_i')$$

---- By Induction Hypothesis.

from ① ② ③ ④.

$$\gamma(i, K) \geq opt(i, K).$$

we can do the same with the recurrence base. where there is no fine. i.e. if $n_i < q_i$

we just have to remove the $\min(f_i(n_i - q_i), c_i)$ from the above proof.

*Dynamic Programming Algorithm for calculating Maximum profit

→

```
for i = 1 to n
    for k = 0 to Gr.
        localMax = 0.
        for qi = 0 to xi
            if k - wi*qi >= 0
                if ni > qi :
                    // deficit
                    r(i,k) = pi*qi - min(fi*(ni - qi), Ci) + r(i-1, k - wi*qi)
                else :
                    // no deficit
                    r(i,k) = pi*qi + r(i-1, k - wi*qi)
```

n → number of Items.
n_i → Item at i^{th} index. ~~max~~ minimum quantity.
Gr → Total Gold.
q_i → quantity of item.
x_i → ~~minimum~~ maximum. quantity required of i^{th} Item.
p_i → profit from one quantity of item i.
f_i → ~~₹~~ fine for i^{th} item.
w_i → weight of i^{th} item.

r(n, Gr) will give the Maximum profit.

* Recurrence for count of different optimal Solutions.

$$cnt(i,k) = \begin{cases} 1 & \text{if } i = 0. \\ \text{Sum}(cnt(i-1, k - w_i q_i)) & \text{if } r(i,k) = p_i q_i - \min(f_i(n_i - q_i), c_i) + r(i-1, k - w_i q_i) \\ 0 \le q_i \le x_i & \\ k - w_i q_i > 0. & \end{cases}$$

or

$$r(i,k) = p_i q_i + r(i-1, k - w_i q_i)$$

i = i^{th} item.
w_i = weight of gold required for i^{th} item.
p_i = Profit of one quantity of that i^{th} item.
n_i = minimum quantity of i^{th} item required / lower limit
x_i = maximum quantity of i^{th} item / upper limit
f_i = fine for one quantity deficit
c_i = cap for the fine on that i^{th} item.
k = local gold available.

* Proof of Correctness.

① Feasibility → There is a solution $cnt(i,k)$ that gives the count of no of optimal solutions.
By Induction on $i+k$.

② Optimality → let $opt(i,k)$ be the optimal solution to the problem

Claim → $cnt(i,k) \geq opt(i,k)$

we will do this by doing Induction on $i+k$.

- **Base Case** : $i = 0$

when there no item or just a dummy item we have.
$r(0,K) = 0$, which is the only answer.

So $Opt(0,K) = Cnt(0,K) = 1$

$\therefore$ Base Case true.

- **Step** $i + K > 0$

here we have to show if for any value of $i \& K$.

$$Cnt(i,K) \geq opt(i,K)$$

lets say opt takes q_i copies of an item.

then

$$Opt(i,K) = Sum(Opt(i-1, K - w_i q_i)) \quad \text{——— (1)}$$

q_i being all optimal solution with q_i copies of i^{th} item.

lets try the same in our recurrence

$$Cnt(i,K) \geq Sum(cnt(i-1, K - w_i q_i)) \quad \text{——— (2)}$$

↑ because we are using the same as opt

now in $i-1 + K - w_i q_i$ there will be two cases, where.

Case 1 $q_i = 0$

$i-1 + K - 0 < i + K$

$\therefore$ $Cnt(i-1, K - w_i q_i) \geq Opt(i-1, K - w_i q_i)$ ——— (3)

——— done.

Case 2 $q_i' > 0$.

Here again

$i-1 + k - w_i q_i' < i + k$.

$\therefore Cnt(i-1, k - w_i q_i') \geq opt(i-1, k - w_i q_i')$ ——— (4)

done

from (1) (2) (3) (4)

$Cnt(i-1, K) \geq opt(i, k)$

We do same in deficit & no deficit case.

returning $cnt(n, G)$ will give number of Optimal Solutions.

* Dynamic Programming Algorithm for counting number of optimal Solutions.

```
for i = 1 to n.
    for K = 0 to G
        for q = 0 to xi
            if k - wiqi ≥ 0
                if ni > qi :
                    # deficit
                    cnt(i,k) = cnt(i,k) + cnt(i-1, k - wi qi)
                else :
                    # no deficit
                    cnt(i,k) = cnt(i,k) + cnt(i-1, k - wi qi)
```

n = no. of Items.
n_i = minimum quantity of item i required
G = Total Gold.
q_i = quantity of item i
x_i = maximum quatity of i^{th} item.
P_i = Profit from one quantity of item i
f_i = fine for i^{th} item.
w_i = weight of i^{th} item.

$cnt(n, G)$ will give number of optimal Solutions

RT Analysis →

Since we are running three loops.

① i → 0 to n.
② k → 0 to G
③ q → 0 to x_i

So the Running Time will be multiplication of.

$$n \times G \times x_i$$

But different items will have different amount upper limit, so in this case the max of x_i will dominate

$$\therefore RT = O(n \times G \times \max(x_i))$$

* Enumerate Algorithm to print optimal Solution values

```
enumerate (S, i, G, jewl, r):
    if i == 0:
        for x in S[1:]:
            print (x, end=' ')
        print()
    else:
        for q in range (0, x+1):
            if (G - w*q >= 0):
                if n > q:
                    // deficit
                    if r[i][G] == p*q - min(f(n-q), c)
                                  + r[i-1][G - w*q]:
                        S[i] = q
                        enumerate (S, i-1, G-w*q, jewl, r)
                else:
                    // no deficit
                    if r[i][G] == p*q + r[i-1][G - w*q]:
                        S[i] = q
                        enumerate (S, i-1, G - w*q, jewl, r)
```

jewl is items list G is Gold
S is Solution list r is ~~opt~~ Maximum profit array.
i no. of items