Universidade Federal de Pernambuco - UFPE
Centro de Informática - CIn
IF969 - 2018.2
Autor: João Victor Marques dos Santos
E-mail: jvms@cin.ufpe.br

Problema proposto

O problema proposto é sobre uma situação onde, você dispõe de n itens com peso $\mathbf{P_n}$ e valor $\mathbf{C_n}$, e um recipiente (locomotiva) com peso suportável até **W**. O objetivo deste caso é levar o maior valor total possível sem ultrapassar o peso suportado pelo recipiente.

O problema se resume em o item $\mathbf{X_n}$ de valor $\mathbf{C_n}$ e peso $\mathbf{P_n}$ ser a o melhor item disponível para aquela situação a ser inserido no recipiente.

Solução ótima

A solução baseada em programação dinâmica começa ao analisarmos o caso e tentarmos dividir ele em subproblemas e buscar uma recorrência sobre o caso.

Se o item $\mathbf{X_n}$ é o melhor a ser inserido, ele faz parte da solução ótima, caso não, é ignorado para aquele momento e o algoritmo deve buscar outras opções, um meio de guardar estas tentativas e também saber como está o valor e o limite de peso do recipiente é a criação de uma tabela $\mathbf{M_{(n,p)}}$, onde **n** será a posição da linha que corresponde aos itens e **p** as colunas que se refere ao peso total (**W**) suportado, e será armazenado nestas posições os valores totais possíveis com p de peso total.

Se $\mathbf{X_n}$ faz parte da solução ótima, então o algoritmo irá armazenar na posição referente a Xn o valor do mesmo ($\mathbf{C_n}$) e o valor da melhor solução para **n-1** elementos com peso $\mathbf{p - p_n}$, já que o valor desta posição $\mathbf{n\text{-}1, p\text{-}p_n}$ seria a melhor possível e que também caberia $\mathbf{X_n}$ sem estourar o limite de peso suportado.

$$M_{(n,p)} = M_{(n-1,\ p-pn)} + C_n$$

Caso **Xn** não faça parte da solução ótima, o algoritmo recebe o valor da posição **n-1** elementos com o mesmo peso, já que a solução ótima pro caso até este ponto seria continuar com os mesmos valores de **Xn-1**.

$$M_{(n,p)} = M_{(n-1,p)}$$

Com isso temos a recorrência que pode ser analisada assim:

Se $\mathbf{p_n} \mathbf{<=} \mathbf{p}$ (se o peso de **n** cabe no recipiente de limite **p**): $\mathbf{M_{(n,p)} = M_{(n-1,\, p-pn)} + C_n}$

Se $\mathbf{p_n > p}$ (se o peso de **n** não couber no recipiente de limite **p**): $\mathbf{M_{(n,p)} = M_{(n-1,p)}}$

Se $\mathbf{W = 0}$ ou $\mathbf{X = 0}$ (se a capacidade total ou a existência de itens for igual a 0): **retorna 0**

Como o algoritmo implementado foi feito por bottom-up, a solução será iterando os elementos de baixo para cima até chegar na posição de maior valor $\mathbf{M_{(n,W)}}$

Considerando **i** como o número de **0 a n** itens, e p como peso de **0 a W**, podemos preencher a tabela linha a linha com as melhores combinações de item e peso baseada nos dois casos acima. Ao utilizarmos o max entre estes dois casos, sempre iremos receber o maior dentre os dois, assim tendo sempre o caso que faz parte da solução ótima.

Se $\mathbf{p_i <= p}$ (se o peso de **n** cabe no recipiente de limite **p**):

$\mathbf{M_{(i,p)} = \max\{M_{(i-1,\, p\, -\, pi)} + C_i, M_{(i-1,\, p)}\}}$

Caso o peso de pn seja maior que o peso p suportado, assumimos também que o valor da solução ótima de $\mathbf{M_{(i-1,p)}}$ é a adquirida para esta posição.

Se $\mathbf{p_i > p}$ (se o peso de **i** não couber no recipiente de limite **p**):

$\mathbf{M_{(i,p)} = M_{(i-1,p)}}$

A tabela nas posições de **i = 0** e **p = 0** podem ser puladas no caminho (para **i** de 1 até **n+1** e para **p** de 1 até **W+1**), pois na criação da tabela assumimos que todas as posições valem 0, não sendo necessário checar novamente essas posições que sempre vão ser iguais a 0 (**i = 0** significa 0 itens para analisar, **p=0** significa que o peso total suportado é 0, ou seja, nada.)

Organizando a solução de método funcional e iterativa, se tornaria isto:

```
Se W ou X == 0:
    retorna 0
para i de 1 até n+1
    para p de 1 até W+1
        se p[i-1] <= p:
            m[i][p] = max(m[i-1][p], (m[i-1][p-p[i]] +c[i]))
        senão:
            m[i][p] = m[i-1][p]
```

Quando o algoritmo chegar na última posição da tabela, $\mathbf{M_{(n,W)}}$, teremos o valor máximo deste caso.

Recuperar os itens utilizados na solução ótima

A solução referente a recuperação dos itens se torna possível após a tabela ser preenchida.

O algoritmo de recuperação dos itens utilizados para chegar neste valor máximo se baseia no fato de que:

Começamos com a posição $\mathbf{M_{(i,p)}}$, onde **p** é igual a **W**, que seria o peso máximo suportado pelo recipiente, com **i** caminhando de **n a 0** (do maior para o menor).

Se $\mathbf{M_{(i,p)}}$ é diferente de $\mathbf{M_{(i-1,p)}}$, significa que a solução ótima com o mesmo peso foi mudada, o que identifica que o item i foi utilizado para chegar no valor máximo.

Agora, para descobrir o resto dos itens possivelmente utilizados, já que p agora se trata da posição final ele é o peso máximo suportado pelo recipiente, então $\mathbf{p = p - p_i}$ irá para a posição do melhor valor antes de o item **i** ter sido adicionado, e assim repetindo o processo até resgatar todos os itens utilizados na solução ótima de peso total **W**.

```
auxiliar = lista
p = W
para i de n até 0:
        (if not) se não m[i][p] == m[i-1][p]:
                auxiliar.armazena(i)
                p = p-p[i]
imprima(auxiliar)
```