Análisis de algoritmos:
Resolución al problema de la mochila 0-1.

Álvaro Antonio Soto Escobar
asotoesc@itam.mx
CU: 160705

Septiembre 2016

1. Introducción

En algoritmia, el problema de la mochila, comúnmente abreviado por KP (del inglés Knapsack problem) es un problema de optimización combinatoria, es decir, que busca la mejor solución entre un conjunto finito de posibles soluciones a un problema. Modela una situación análoga al llenar una mochila, incapaz de soportar más de un peso determinado, con todo o parte de un conjunto de objetos, cada uno con un peso y valor específicos. Los objetos colocados en la mochila deben maximizar el valor total sin exceder el peso máximo.

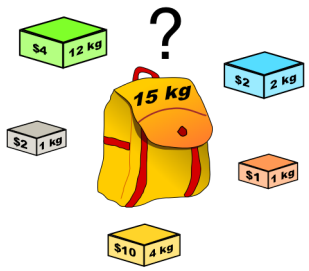

Figura 1: Ejemplo del problema de la mochila

2. Definición del problema

Supongamos que tenemos **n** distintos tipos de ítems, que van del **1** al **n** De cada tipo de ítem se tienen q_i ítems disponibles, donde q_i es un entero positivo. Cada tipo de ítem i tiene un **beneficio** asociado dado por v_i y un **peso** (o volumen) w_i. Usualmente se asume que el beneficio y el peso no son negativos. Para simplificar la representación, se suele asumir que los ítems están listados en

orden creciente según el peso (o volumen). Por otro lado se tiene una mochila, donde se pueden introducir los ítems, que soporta un **peso máximo** (o volumen máximo) W. El problema consiste en meter en la mochila ítems de tal forma que se **maximice el valor de los ítems que contiene y siempre que no se supere el peso (o volumen) máximo que puede soportar la misma**. La solución al problema vendrá dado por la secuencia de variables x_1, x_2, ..., x_n donde el valor de x_i indica cuantas copias se meterán en la mochila del tipo de ítem i.
El problema se puede expresar matemáticamente por medio del siguiente programa lineal:

$$\begin{aligned} \textit{maximizar} \quad & \sum_{i=1}^{n} v_i x_i \\ \textit{talque} \quad & \sum_{i=1}^{n} w_i x_i \leq W \\ y \quad & 1 \leq q_i \leq \infty. \end{aligned}$$

Si q_i=1 para i=1,2,...,n se dice que se trata del problema de la mochila 0-1. Si uno o más q_i es infinito entonces se dice que se trata del problema de la mochila no acotado también llamado a veces problema de la mochila entera. En otro caso se dice que se trata del problema de la mochila acotado. [1]

3. Solución

```
#include<stdio.h>

int max(int a, int b){
    return (a > b)? a : b;
}

int knapSack(int W, int wt[], int v[], int n){
    int i, j;
    int T[n+1][W+1];

    /* starting max value for first iteration */
    for ( j = 0; j <= W; j++) T[0][j] = 0;

    for (i = 1; i <= n; i++){
        for (j = 0; j <= W; j++){
            if (j >= wt[i-1]){
                T[i][j] = max(T[i-1][j], T[i-1][j-wt[i-1]] + v[i-1]);
            }else{
                T[i][j] = T[i-1][j];
            }
        }
    }
```

```
    return T[n][W];
}

int main(){
    int wt[] = {11, 7, 5, 4, 3, 3, 2, 2, 2, 2, 1};
    int v[] = {20, 10, 11, 5, 25, 50, 12, 6, 4, 5, 30};
    int W = 20;
    int n = 11;
    printf("Optimal value => %d\n", knapSack(W, wt, v, n));
    return 0;
}
```

4. Salida de ejecución

```
alvaro@skyline.headup.local:
hist:540  jobs:0 $ gcc knapsack.c  -o knapsack
alvaro@skyline.headup.local:
hist:541  jobs:0 $ ./knapsack
Optimal value => 143
```

5. Referencias

1. https://en.wikipedia.org/wiki/Knapsack_problem
2. http://www.es.ele.tue.nl/education/5MC10/Solutions/knapsack.pdf
3. https://www.youtube.com/watch?v=8LusJS5-AGo