UNTREF
UNIVERSIDAD NACIONAL
DE TRES DE FEBRERO

Gabriel Martín Moran
Ingeniería en Computación

Knspsack Problem utilizando Dynamic Programming

Análisis Numérico

Como trabajo final de la materia Análisis Numérico del primer cuatrimestre del año 2019 de la UNTREF, se realizó la implementación en Python del algoritmo llamado Dynamic Programming (Programación Dinámica) aplicado al Knapsack Problem (El problema de la mochila).

Este problema pertenece al grupo de los **np-completos**, es decir que a medida que aumenta el conjunto de datos para el que se quiere resover, también aumenta el tiempo de resolución de manera no polinómica, sino exponencial.

Dynamic Programming es una técnica que sólo se puede aplicar para resolver ciertos problemas, como es el caso del Knapsack Problem. En la mayoría de los casos, el tiempo de obtención de la solución utilizando este algoritmo es mucho menor que realizar pruebas combinatorias por fuerza bruta (tiempo exponencial). Sin embargo, en los peores casos, este algoritmo también se puede comportar de manera exponencial en cuanto a tiempo, por lo que podríamos decir que es un algoritmo que funciona bien en gran cantidad de casos pero no reduce el problema de la mochila a uno de tipo **p** (cuya resolución se obtiene en tiempo polinomico).

La técnica de resolución llamada Dynamic Programming (aplicada al Knapsack Problem), consta en llenar una tabla comprobando para cada celda el valor del mismo número de celda en la columna previa, contra el valor de la celda de la columna previa en la posición del peso máximo actual menos el peso del objeto que se está evaluando.

Capacidad	Item 0	Item 1	Item 2	Item 3	Item 4
0	0	0	0	0	0
1	0	0	0	0	0
2	0	16	16	16	16
3	0	16	19	19	19
4	0	16	19	23	23
5	0	16	35	35	35
6	0	16	35	39	39
7	0	16	35	42	44
	Value: 0 Weight: 0	Value: 16 Weight: 2	Value: 19 Weight: 3	Value: 23 Weight: 4	Value: 28 Weight: 5

Una vez completada la tabla, se utiliza una técnica llamada Traceback para obtener que objetos fueron los que se insertaron en la mochila, parándose en la última celda de la tabla (inferior derecha), retrocediendo sobre las columnas de la misma y consultando si hubo modificaciones con respecto al valor en la columna anterior para la misma fila. Si es así, entonces el objeto fue insertado y se cambia la fila a comprobar a la correspondiente de descontar al valor de la celda actual, el valor del objeto insertado; si por el contrario ambas celdas poseen el mismo valor, se retrocede una columna pero el número de fila se mantiene, repitiendo todo el proceso hasta llegar a la fila con el objeto 0.

Capacidad	Item 0	Item 1	Item 2	Item 3	Item 4
0	0	0	0	0	0
1	0	0	0	0	0
2	0	16	16	16	16
3	0	16	19	19	19
4	0	16	19	23	23
5	0	16	35	35	35
6	0	16	35	39	39
7	0	16	35	42	44
	Value: 0 Weight: 0	Value: 16 Weight: 2	Value: 19 Weight: 3	Value: 23 Weight: 4	Value: 28 Weight: 5

El código con la implementación de este algoritmo se encuentra publicado en GitHub en el enlace:

- https://github.com/GabrielMartinMoran/Knapsack-DynamicProgramming

En este repositorio, además del código programado en Python, esta publicada la documentación necesaria para comprender paso a paso el algoritmo.