Análisis de Algoritmos

Reporte Programación Dinámica Mochila

Ingeniería en Sistemas Computacionales

Docente: Roberto Oswaldo Cruz Leija

Alumno: Héctor Mauricio Zamudio Domínguez

Fecha: 7 de noviembre de 2019

Fundamentos

La programación dinámica es un método para reducir el tiempo de ejecución de un algoritmo mediante la utilización de subproblemas superpuestos y subestructuras óptimas.

Es importe remarcar que la implementación de este tipo de programación es posible cuando el problema original cuenta con dichos subproblemas; los subproblemas son resueltos y esta solución ayuda a solución general del problema. Utilizar programación dinámica implica realizar una estructura que sea de utilizada para guardar y consultar las soluciones de los problemas de manera sencilla.

Un problema donde se puede aplicar la programación dinámica es el problema de la mochila (mejor conocido como "knapsack problem" en inglés). El problema consiste en elegir un subconjunto de n artículos maximizando el beneficio obtenido considerando el peso total de los artículos seleccionados, sin exceder la capacidad c de la mochila. Este problema cuenta con subproblemas debido a que, si encontramos una solución óptima para una mochila de capacidad c-1, esa información nos resulta útil para dar solución para una mochila de capacidad c ya que el problema se "redujo" tanto que solo resta dar solución a una mochila de capacidad c=1 con los artículos no involucrados en la solución anterior.

Procedimiento

La solución en programación a este problema se realizo con una clase Mochila que se compone de una lista de Ítems con cierto peso y beneficio, además de la capacidad máxima de la mochila.

La estructura para almacenar las soluciones a los subproblemas consiste en una matriz Mij donde i es la capacidad de la mochila y son los ítems disponibles. Para cada posición se almacena el beneficio máximo para cada i-ésima capacidad de la mochila, donde si incluir el j-ésimo ítem no rebasa el peso máximo de la matriz y además aporta más beneficio (beneficio consultado de la solución anterior) se actualiza el máximo beneficio ya que se ha logrado mejorar, en caso contrario continua con la solución actual. Una vez completada la matriz, únicamente se realiza una recuperación de los ítems utilizados en la solución.

Resultados y conclusiones

Para la siguiente lista de Ítems, y para una mochila de capacidad 10

Ítem	peso	Beneficio
0	3	34
1	6	28
2	6	90
3	1	23
4	9	11
5	1	19
6	11	7

```
  0.0   0.0   0.0  34.0  34.0  34.0  34.0  34.0  34.0  34.0  34.0
  0.0   0.0   0.0  34.0  34.0  34.0  34.0  34.0  34.0  62.0  62.0
  0.0   0.0   0.0  34.0  34.0  34.0  90.0  90.0  90.0 124.0 124.0
  0.0  23.0  23.0  34.0  57.0  57.0  90.0 113.0 113.0 124.0 147.0
  0.0  23.0  23.0  34.0  57.0  57.0  90.0 113.0 113.0 124.0 147.0
  0.0  23.0  42.0  42.0  57.0  76.0  90.0 113.0 132.0 132.0 147.0
  0.0  23.0  42.0  42.0  57.0  76.0  90.0 113.0 132.0 132.0 147.0
Articulos solucion
1,23
6,90
3,34
Maximo beneficio: 147
```

Ahora si la capacidad aumente una unidad con el mismo conjunto de ítems.

```
   0.0   0.0   0.0  34.0  34.0  34.0  34.0  34.0  34.0  34.0  34.0  34.0
   0.0   0.0   0.0  34.0  34.0  34.0  34.0  34.0  34.0  62.0  62.0  62.0
   0.0   0.0   0.0  34.0  34.0  34.0  90.0  90.0  90.0 124.0 124.0 124.0
   0.0  23.0  23.0  34.0  57.0  57.0  90.0 113.0 113.0 124.0 147.0 147.0
   0.0  23.0  23.0  34.0  57.0  57.0  90.0 113.0 113.0 124.0 147.0 147.0
   0.0  23.0  42.0  42.0  57.0  76.0  90.0 113.0 132.0 132.0 147.0 166.0
   0.0  23.0  42.0  42.0  57.0  76.0  90.0 113.0 132.0 132.0 147.0 166.0
Articulos solucion
1,19
1,23
6,90
3,34
Maximo beneficio: 166
```

Observamos que logra ingresar el único articulo de peso = 1 que logra conformar la nueva solución, siendo que antes no era tomado en cuanta ya que su beneficio era menor al ítem de mismo peso pero mayor beneficio (1,23).

Como conclusión, el implementar la programación dinámica resulta una ventaja para este tipo de problemas debido a que a pesar de que el problema puede ser resuelto de distintas maneras, por fuerza bruta o recursividad, ambos calculan las soluciones para subproblemas que ya habían resulto en anteriores iteraciones, por otro lado con la programación dinámica únicamente se calcula una vez la solución para los subproblemas y estas soluciones son consultadas si se requiere o no reduciendo significativamente el esfuerzo computacional de encontrar la solución.

Referencias

https://es.wikipedia.org/wiki/Programaci%C3%B3n_din%C3%A1mica

https://www.uaeh.edu.mx/scige/boletin/tlahuelilpan/n6/e2.html