Actividad 09

Análisis y Diseño de Algoritmos

Dr. Carlos Villaseñor

Alumno: Eduardo Alberto Rodríguez García

1. Usa programación dinámica para resolver el problema de encontrar el máximo conjunto independiente con mayor peso. Usa el archivo MWIS.txt para obtener los datos, cada dato es el peso de un vértice en un grafo camino. Contesta los siguiente: ¿Cuáles de los vértices 1, 2, 3, 4, 17, 117, 517, y 997 forman parte de la solución óptima?

def mwis(w):

    # Obtiene el valor optimo

    n = len(w)

    a = [0, w[0]]

    for i in range(2, n+1):

        a.append(max(

            a[i-1],

            a[i-2] + w[i-1]

        ))

    # Reconstruir la solucion

    i = n

    s = set()

    while i >= 1:

        if a[i] == a[i-1]:

            i -= 1

        else:

            s.add(i-1)

            i -= 2

    print(s)

w = []

with open("MWIS.txt", "r") as f:

    for line in f:

        line = line.strip()

        w.append(int(line))

mwis(w)

Patrón de fondo

Descripción generada automáticamente

2. Usa programación dinámica para resolver el siguiente problema de la mochila. Suponga una capacidad máxima de 140 unidades. ¿Cuál es el valor óptimo de la mochila?, ¿Cuáles son los objetos que debemos tomar?

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Valor | 79 | 32 | 47 | 18 | 26 | 85 | 33 | 40 | 45 | 59 |
| Peso | 85 | 26 | 48 | 21 | 22 | 95 | 43 | 45 | 55 | 52 |

import numpy as np

object\_dict = {

    0:{"value":79, "weight":85},

    1:{"value":32, "weight":26},

    2:{"value":47, "weight":48},

    3:{"value":18, "weight":21},

    4:{"value":26, "weight":22},

    5:{"value":85, "weight":95},

    6:{"value":33, "weight":43},

    7:{"value":40, "weight":45},

    8:{"value":45, "weight":55},

    9:{"value":59, "weight":52}

    }

number\_of\_values = len(object\_dict)

item\_values\_list = range(number\_of\_values + 1)

max\_weight = 140

weights\_list = range(max\_weight + 1)

cells = [[0 for w in weights\_list] for i in item\_values\_list]

for i in item\_values\_list:

    for w in weights\_list:

        if i == 0 or w == 0:

            cells[i][w] = 0

        elif object\_dict[i-1]["weight"] <= w:

            current\_item\_value = object\_dict[i-1]["value"]

            current\_item\_weight = object\_dict[i-1]["weight"]

            cells[i][w] = max(cells[i-1][w], current\_item\_value + cells[i-1][w-current\_item\_weight])

        else:

            cells[i][w] = cells[i-1][w]

most\_value = cells[number\_of\_values][max\_weight]

print(np.matrix(cells))

print('Most optimal value stolen:', most\_value)

remaining\_weight = max\_weight

for i in range(number\_of\_values, 0, -1):

    if most\_value <= 0:

        break

    if most\_value == cells[i - 1][remaining\_weight]:

        continue

    else:

        print('Item included: ', list(object\_dict.keys())[i - 1])

        most\_value = most\_value - object\_dict[i - 1]["value"]

        remaining\_weight = remaining\_weight - object\_dict[i - 1]["weight"]

Imagen de la pantalla de un computador

Descripción generada automáticamente con confianza baja