TEORÍA DE ALGORITMOS 1

Guia de ejercicios: Programación dinámica

Por: ING. VÍCTOR DANIEL PODBEREZSKI
vpodberezski@fi.uba.ar

Enunciados

Para cada ejercicio considere, además de resolver lo solicitado, definir el problema, expresar la relación de recurrencia, brindar pseudocódigo y calcular la complejidad temporal, espacial.

Referencias

- ⭐ Fácil
- ⭐⭐ Medio
- ⭐⭐⭐ Difícil

1. ⭐Para una inversión inmobiliaria un grupo inversor desea desarrollar un barrio privado paralelo a la una ruta. Con ese motivo realizaron una evaluación de los diferentes terrenos en un trayecto de la misma. Diferentes inversores participarán, pero a condición de comprar algún terreno en particular. El grupo inversor determinó para cada propiedad su evaluación de ganancia. El mismo surge como la suma de inversiones ofrecida por el terreno menos el costo de compra. Debemos recomendar que terrenos contiguos comprar para que maximicen sus ganancias. Ejemplo: S = [-2, 3, -3, 4, -1, 2]. La mayor ganancia es de 5, comprando los terrenos de valor [4, -1, 2]. Solucionar el problema mediante un algoritmo de programación dinámica.

2. ⭐Dado un Grafo dirigido, acíclico G(V, E) con pesos en sus aristas y dos vértices “s” y “t”; queremos encontrar el camino de mayor peso que exista entre “s” y “t”. Resolver mediante programación dinámica.

3. ⭐El dueño de una empresa desea organizar una fiesta de la misma. Cómo quiere que la fiesta sea lo más tranquila posible, no quiere que asistan un empleado y su jefe directo. Pidió a su encargado que le ponga un rating de convivencia (Ri) a cada empleado. Con dicha información más un organigrama de la compañía, le pidió a un especialista en informática que diseñe un algoritmo para obtener el listado de los empleados que deberá invitar a la fiesta. Teniendo en cuenta que: Ri > 0, que no cuenta para el armado del algoritmo y que la

estructura jerárquica es en forma de árbol. Se pide resolver mediante programación dinámica.

4. ⭐La organización de una feria internacional tiene que programar diferentes eventos a realizar en su escenario principal. Para ello pueden elegir, en los diferentes días del evento, entre alguno de los siguientes rubros: un cantante, una compañía de danza, un show de variedades o un humorista. Disponen de una oferta de cada tipo para cada día y la posible ganancia por venta de entradas. Existen ciertas restricciones que se aplican. No se pueden repetir 2 días seguidos el mismo rubro. Además por el tiempo de preparación un día después de un cantante solo puede presentarse un humorista. Plantear la resolución mediante programación dinámica.

5. ⭐⭐⭐Una agencia de inteligencia ha conseguido interceptar algunos mensajes encriptados de una agencia rival. Mediante espionaje logran saber algunas cosas para desencriptar los mensajes. Por ejemplo, que para dificultar la tarea de los criptoanalistas los mensajes enviados no contienen espacios. Se ha organizado un grupo de trabajo para generar un algoritmo para quebrar la seguridad. El trabajo se dividió en diferentes partes. A usted le toca, dado un string “desencriptado” y sin espacios, determinar si lo que se lee corresponde a un posible mensaje en idioma castellano. El proceso debe ser rápido dado que se debe utilizar muchas veces. Cuenta con un diccionario de “n” palabras. y con una cadena de texto con el posible mensaje.

   Ejemplo: si el diccionario es “peso”, “pesado”, “oso”, “soso”, “pesa”, “dote”, ”a”, “te”,
   para la cadena de texto “osopesadotepesa”. Existe un posible mensaje con las palabras “oso”, “pesado”, “te”, “pesa”. Para la cadena de texto “ososoapesadote”. No existe un posible mensaje

   a. Construir una solución que informe si la cadena de entrada es un posible texto utilizando programación dinámica.
   b. Existe la posibilidad de que una cadena de texto puede corresponder a más de un mensaje. Modifique su solución para que se informen todos los posibles mensajes. Determine el impacto en las complejidades en el nuevo algoritmo.

6. ⭐⭐⭐Una empresa de transporte aéreo cuenta con un avión que cubre una ruta que une dos ciudades. Se especializa en llevar maquinaria pesada. Puede realizar un vuelo diario y trasladar una cantidad de peso determinada. Por cada kilo transportado cobra una tarifa. Por día recibe un pedido de transporte que puede aceptar o rechazar en su totalidad (no se pueden fraccionar). Si rechaza un pedido, lo pierde y no lo puede realizar otro día. Existe una regulación especial que indica que la carga máxima a transportar disminuye según una fórmula por día. Se restablece a su máximo únicamente si el avión pasa por revisión e inspección. Este proceso insume 3 días seguidos. El gerente de la empresa cuenta con una planilla que detalla los envíos solicitados en el próximo mes y la tabla de peso máximo por día pasado desde la última revisión. El avión llegó el día anterior de la revisión. Construir un algoritmo que permita maximizar las ganancias.

7. ⭐⭐Contamos con una secuencia de “n” matrices (A1,A2,...,An) a ser multiplicadas. Queremos computar el siguiente producto A1*A2*...*An. Considere utilizar la multiplicación estándar de matrices. si tengo la matriz A de tiene una dimensión de p x q, y la matriz B tiene

una dimensión de q x r, realizar C = A*B requiere p x q x r multiplicaciones escalares. Sabemos que podemos elegir diferentes modos de multiplicar las matrices. La elección del orden de la multiplicación tiene un impacto importante en el costo total del cálculo. Por ejemplo, para 4 matrices A1*A2*A3*A4 podemos multiplicar de las siguientes maneras:

(A1*(A2*(A3*A4))) =
(A1*((A2*A3)*A4)) =
((A1*A2)*(A3*A4)) =
((A1*(A2*A3))*A4) =
(((A1*A2)*A3)*A4)=

Plantee mediante programación dinámica un mecanismo de seleccionar el orden de multiplicación de forma de minimizar la cantidad de multiplicaciones escalares a realizar.

8. ⭐Contamos con una carretera de longitud M km que tiene distribuidos varios carteles publicitarios. Cada cartel "i" está ubicado en un "ki" kilómetro determinado (pueden ubicarse en cualquier posición o fracción de kilómetro) y a quien lo utiliza le asegura una ganancia "gi". Por una regulación no se puede contratar más de 1 cartel a 5km de otros. Queremos determinar qué carteles conviene contratar de tal forma de maximizar la ganancia a obtener.

9. ⭐Un ramal ferroviaria pone en concesión los patios de comida en todas las estaciones. Son en total "n" estaciones. Por cada estación se cuenta con el promedio de facturación de los últimos 5 años. Por normativa antimonopólica existe como limitante que ninguna empresa puede explotar 3 o más estaciones contiguas. Pero, no existe una cantidad máxima de estaciones a explotar. Un oferente nos solicita que le digamos cuales son las estaciones que le conviene obtener para maximizar sus ganancias. Plantee la solución mediante programación dinámica.

10. ⭐⭐Un amigo le propone un juego de cartas a otro. Las reglas son sencillas. Se mezcla un mazo de "n" cartas y se preparan en una fila boca arriba. El juego es por turnos. Un primer participante debe seleccionar una de las cartas de los extremos y se la queda. A continuación el contrincante debe seleccionar entre las cartas restantes una de los extremos. Repiten el procedimiento hasta que no queden cartas en la fila. Gana el jugador cuya suma de los valores de las cartas que eligió suman más. Los dos son buenos jugadores y muy competitivos. Es posible que tengan sus métodos para intentar conseguir el mejor puntaje. Proponer un método utilizando programación dinámica para obtener el mayor puntaje posible siendo el jugador que empieza el juego. Suponer que el adversario usa el mismo método e intenta obtener para sí el mayor valor posible.

11. ⭐⭐? La detección de ciclos negativos tiene una variedad de aplicaciones en varios campos. Por ejemplo en el diseño de circuitos electrónicos VLSI, se requiere aislar los bucles de retroalimentación negativa. Estos corresponden a ciclos de costo negativo en el grafo de ganancia del amplificador del circuito. Tomando como entrada de nuestro problema un grafo ponderado con valores enteros (positivos y/o negativos) dirigido donde un nodo corresponde al punto de partida, queremos conocer si existe al menos un ciclo negativo y en caso afirmativo mostrarlo en pantalla. Proponer una solución al problema que utiliza programación dinámica.

12. ⭐Un dueño de un camión de transporte se comprometió a trasladar una carga de una ciudad A a la ciudad B. Para realizar el recorrido puede optar por diferentes caminos que pasan por diferentes ciudades intermedias. Nos acerca un mapa donde para cada tramo que une las diferentes ciudades indica el costo de realizar el mismo. Vemos que algunos costos son positivos (combustible, peaje, etc) y otros negativos o cero (yendo por esos tramos puede ganar unos pesos haciendo algunos encargos "particulares"). Nos solicita que le informemos cuál sería el recorrido óptimo para minimizar el gasto total del viaje. Presentar un algoritmo polinomial utilizando programación dinámica que lo resuelva.

13. ⭐En un grafo se conoce como conjunto independiente a un subconjunto de vértices del mismo tal que ninguno sea adyacente a otro. Es decir, es un conjunto X de vértices tal que para ningún par de ellos existe alguna arista que los conecte. No se conoce un algoritmo eficiente que resuelva el problema. Sin embargo, para algunos casos especiales de grafos si es posible. Considerar el siguiente caso: Un grafo es un camino si se pueden escribir sus nodos como una sucesión V_1, V_2, ..., V_n donde cada nodo V_i tiene un eje únicamente con V_{i-1} y V_{i+1}. (Excepto en los extremos donde solo tienen un eje con el siguiente o el anterior). Considerar que cada nodo tiene un peso entero positivo. Construir un algoritmo utilizando programación dinámica que encuentre el set independiente de mayor peso.

14. ⭐Para un nuevo satélite a poner en órbita una empresa privada puede optar por incluir diversos sensores a bordo (por ejemplo: variación de temperatura, humedad en tierra, caudal de ríos, etc). Cada uno de ellos tiene un peso "pi" y una ganancia "gi" calculado por su uso durante la vida útil del satélite. Si bien les gustaría incluir todos, el satélite tiene una carga máxima P que puede llevar. Nos piden que generemos un algoritmo (utilizando programación dinámica) para resolver el problema. Indique si su solución es polinomial.

15. ⭐⭐Luego de aprender programación dinámica un estudiante en una charla de café le explica a un amigo que atiende un negocio el algoritmo de cambio mínimo de monedas. Con eso, afirma, podrá optimizar el despacho de mercadería. El comerciante despacha pedidos de cierto producto que viene en empaques prearmados de distintas cantidades o sueltos en unidades. Sin embargo, el amigo le replica que su algoritmo es poco realista. Supone que uno tiene una cantidad ilimitada de empaques de cada presentación. Luego de pensar unos momentos, el estudiante llega a una variante de este problema teniendo en cuenta esta restricción. ¿Podría usted detallar cuál es esta solución?

16. ⭐Se conoce como “Longest increasing subsequences” al problema de, dado un vector de numérico, encontrar la subsecuencia más larga de números (no necesariamente consecutivos) donde cada elemento sea mayor a los anteriores. Ejemplo: En la lista → 2, 1, 4, 2, 3, 9, 4, 6, 5, 4, 7. Podemos ver que la subsecuencia más larga es de longitud 6 y corresponde a la siguiente “1, 2, 3, 4, 6, 7”. Resolver el problema mediante programación dinámica.

17. ⭐ ⭐ Se define el problema 2-Partition de la siguiente manera: Se cuenta con un conjunto de “n” elementos. Cada uno de ellos tiene un valor asociado. Se desea separar los elementos en 2 conjuntos que cumplan con: La suma de los valores de cada conjunto sea igual entre ellos. No se conoce un algoritmo eficiente para resolver este problema. Sin embargo - al igual que otros problemas puede ser resuelto utilizando programación dinámica de forma pseudopolinomial. Presente una solución al problema utilizando dicha metodología.

18. ⭐ ⭐ Una variante del problema de la mochila corresponde a la posibilidad de incluir una cantidad ilimitada de cada uno de los elementos disponibles. En ese caso, tenemos una mochila de tamaño “k” y un conjunto de “n” elementos con stock ilimitado. Cada elemento tiene un peso y un costo. Queremos seleccionar el subconjunto de elementos que maximice la ganancia de la mochila sin superar su capacidad. Solucione el problema utilizando programación dinámica.

19. ⭐ ⭐ Dada una matriz booleana de n x m queremos encontrar la mayor submatriz cuadrada cuyos elementos sean sólo “true”. Diseñar un algoritmo mediante programación dinámica para resolverlo.

20. ⭐ ⭐ Recordemos el problema de selección de intervalos. Partimos de un recurso y un conjunto de intervalos. Cada intervalo cuenta con una fecha de inicio y otra de finalización. Queremos maximizar la cantidad de intervalos a seleccionar siempre que los mismos no se superpongan entre sí. Conocemos un algoritmo greedy que los resuelve de forma óptima. Queremos una variante donde la respuesta al problema corresponda a maximizar el tiempo de ocupación del recurso. Proponer un algoritmo utilizando programación dinámica que lo solucione.

21. ⭐ ⭐ Una empresa constructora planea iniciar un nuevo proyecto complejo. ha realizado un estudio de cada una de las tareas a realizar. Para cada tarea se conoce la duración de la misma y cuales son las tareas anteriores requeridas para comenzar a realizarla. Se pueden realizar en paralelo tantas tareas como podamos. Como requisito para iniciar se debe contratar un seguro por accidentes. Este seguro debe cubrir todo el ciclo de desarrollo. Es por eso que nos solicitan calcular (sin tener en cuenta posibles retrasos) cual es el tiempo máximo que nos puede demandar la conclusión del mismo. Ejemplo: Si tengo 4 tareas. A = (23 días, Requiere: C y D). B=(5 días, sin requerimientos). C = (5 días, requiere D y B), D=(10 días, sin requerimientos). El tiempo de desarrollo máximo es 38 = 10 (completar B y D) + 5 (completar C) + 23 (completar A). Resolver el problema utilizando programación dinámica.

22. ⭐ Se proyectó la construcción de una línea de tensión eléctrica. En su trayecto pasa cerca de “n” ciudades que tienen diferentes demandas de consumo eléctrico. Para la interconexión entre la línea y una central se debe construir una nexo. Por cuestiones regulatorias y constructivas no se pueden construir nexos a menos de “x” kilómetros entre sí. Contamos con el listado de las ciudades. Para cada ciudad nos informan su población y la ubicación en la línea del nexo a construir. Mediante programación dinámica proponga una solución que permita seleccionar qué ciudades conectar para maximizar la cantidad total de población cubierta por esta línea.

23. ⭐ El dueño de una cosechadora está teniendo una demanda muy elevada en los próximos 7 meses. Desde “n” campos lo quieren contratar para que preste sus servicios.

Lamentablemente no puede hacer todos los contratos puesto que varios de ellos se superponen en sus tiempos. Cuenta con un listado de los pedidos donde para cada uno de ellos se consigna: fecha de inicio, fecha de finalización, monto a ganar por realizarlo. Su idea es seleccionar la mayor cantidad de trabajos posibles. Mostrarle que esta solución puede no ser la óptima. Proponer una solución utilizando programación dinámica que nos otorgue el resultado óptimo (que trabajos elegir y cuanto se puede ganar).

24. ⭐⭐ Una empresa que realiza ciencia de datos debe realizar en las próximas “n” semanas procesos y cálculos intensivos. Para eso debe contratar tiempo de cómputo en la nube. Realizando una estimación conocen cuantas horas de cómputo necesitaran para cada una de las semanas. Por otro lado, luego de negociar con los principales proveedores tienen 2 opciones que puede combinar a gusto. Opción 1: Contratar a la empresa “Arganzón” por semana. En esa semana se cobra proporcional al tiempo de cómputo según un parámetro “r” (horas computo x r). Opción 2: Contratar a la empresa “Fuddle” por un lapso de 5 semanas contiguas. Durante el lapso contratado se paga una tarifa fija de “c”. Proponer una solución utilizando programación dinámica que nos indique la secuencia de elecciones a realizar para minimizar el costo total de cómputo.

25. ⭐⭐ Un repartidor tiene una moto que le permite llevar hasta “k” kilos. Comienza el reparto todas las mañanas en una empresa de mensajería. Existen tipos de paquetes estandarizados. Cada uno con un peso y un valor que se paga por su transporte. Por la mañana el repartidor llega a la empresa y le indican cuánta cantidad de cada tipo de paquetes están disponibles para el transporte en el día. Él puede determinar cuántos de cada uno de ellos transportar. Ayudarlo a seleccionar las cantidades para maximizar la ganancia sin superar la capacidad de la moto.

26. ⭐⭐ Considere la siguiente variante del problema de la mochila. Contamos con un conjunto ordenado de “n” productos numerados del 1 al “n”. Cada producto i tiene asociado un peso pi y un valor vi. Queremos seleccionar un subconjunto de elementos logrando que la suma de sus valores sea la máxima posible y sin superar K entre la suma de los pesos de los elementos seleccionados. Además se restringe la selección de 2 productos con numeración consecutiva.

Ejercicios resueltos

Cada ejercicio resuelto busca mostrar cómo se debe analizar, resolver y justificar la resolución del ejercicio, tanto en un trabajo práctico o en un parcial

1. ⭐Sea G=(V,E) un grafo dirigido. No se conoce un algoritmo polinomial para encontrar el camino más largo simple (El largo del camino corresponde a la cantidad de nodos por los que pasa). Sin embargo, para ciertos casos se puede resolver en forma eficiente. Consideremos que G corresponde a un grafo ordenado. En este caso se pueden disponer ordenados los nodos de forma que: i) Desde cada nodo pueden existir ejes salientes a nodos con mayor "índice" (posteriores en su ordenamiento) pero no a nodos de menor índice. y ii) Cada nodo excepto el último tiene ejes salientes. Proponer un algoritmo eficiente que resuelva este problema.

Pseudocódigo

```python
Sea G(V, E) el grafo dirigido, donde V representa los vértices y E las aristas.

Sea opt un Hash donde las claves son los vértices y los valores, el óptimo del
nodo i, representando el camino máximo finalizado en el nodo i.


V' = sortVertexByGrade(V) # ordenamos los vértices por su grado, de forma
                            ascendente

for v in V':
      if v.grade == 0:
            opt[v] = 0

      maxV = 0
      for adj in v.adjacent:
            if exists E(adj --> v): # existe una arista del nodo adj a v
                  maxV = max(opt[adj] + 1, maxV)
      opt[v] = maxV

maxSinglePathLenght = opt[n]
```

```
# reconstruimos la solución
maxSinglePath = [n]
initial_node = n

while  NOT v.incidentAdj
       maxV = 0
       maxVOpt = 0

       # recorremos los adyacentes incidentes a "v" y nos quedamos con el mayor
       valor del óptimo (aquel con el camino más largo hasta el momento)
       for adj in v.incidentAdj:
              if opt[adj] > maxVOpt:
                     maxV = adj
                     maxVOpt = opt[adj]
       maxSinglePath.add(maxV)
       v = maxV

return maxSinglePath
```

Ecuación de recurrencia

Se plantea que el óptimo del nodo i representa el camino máximo finalizado en el nodo i.

opt[i] = max { opt[j] } + 1, para todo nodo j que tiene aristas entrante al nodo i
opt[i] = 0 para todos los nodos sin ejes entrantes (caso base)

El valor máximo se encontrará en opt[n].

Explicación
La solución consiste en, en primer lugar, ordenar los nodos según su grado, para comenzar luego a recorrer los nodos en ese orden.

El primer nodo, tendrá grado saliente máximo y el último, tendrá grado saliente cero (según lo especificado en el enunciado).

Al iterar cada nodo, se calculará el óptimo como el máximo de los óptimos de los adyacentes + 1. Al haber ordenando por el grado de cada nodo, aseguramos que los óptimos de los adyacentes que inciden siempre estarán previamente calculados.

El valor máximo se encontrará en opt[n], dado que, por cómo realizamos el ordenamiento, el nodo “n” es quien no posee aristas salientes.

Análisis de complejidad

Complejidad temporal

La complejidad temporal de la solución es O(N^2) ya que

- ordenar los nodos por grado posee una complejidad de O(N.logN)
- Iterar los nodos y sus adyacente posee una complejidad de O(N^2) en el peor de los casos
- Reconstruir la solución posee una complejidad de O(N^2) en el peor de los casos

Complejidad espacial

La complejidad espacial es de O(N), dado que el vector `maxSinglePath` puede contener en el peor de los casos N elementos.