Trabajo Práctico n.º 2

Teoría de Algoritmos 1 - 2c 2024 Trabajo Práctico 2

Lineamientos básicos

- El trabajo se realizará en grupos de cuatro o cinco personas.
- Se debe entregar el informe en formato pdf y código fuente en (.zip) en el aula virtual de la materia.
- El lenguaje de implementación es libre. Recomendamos utilizar C, C++ o Python. Sin embargo si se desea utilizar algún otro, se debe pactar con los docentes.
- Incluir en el informe los requisitos y procedimientos para su compilación y ejecución. La ausencia de esta información no permite probar el trabajo y deberá ser re-entregado con esta información.
- El informe debe presentar carátula con el nombre del grupo, datos de los integrantes y y fecha de entrega. Debe incluir número de hoja en cada página. No debe superar las 20 páginas + carátula + índice + referencias.
- Debe entregar en el informe las fuentes consultadas en una sección de referencias.
- En caso de re-entrega, entregar luego del informe original un apartado con las correcciones realizadas

Parte 1: Las tareas del servidor público.

En una ciudad con estructura tipo damero (cuadrícula) trabaja un servidor público muy particular. Tiene asignado un rectángulo de nxm cuadras y debe seleccionar un subconjunto de ellas para realizar un trabajo. El mismo comienza en la cuadra más al sureste. Se puede mover a otra cuadra. Únicamente puede trasladarse a alguna cuadra que se encuentre al oeste o al norte de donde se encuentra (puede saltarse una o varias en esas direcciónes). Todas las cuadras tienen una ganancia posible si determina realizar una tarea allí. Desea obtener la mayor ganancia posible. Pero existe una restricción adicional, por cada nueva cuadra que seleccione la ganancia debe ser mayor a la anterior. Sino, no lo puede realizar. Ayude al empleado a resolver el problema.

Se pide:

1. Resolver el problema utilizando programación dinámica. (incluya en su solución definición del subproblema, relación de recurrencia y pseudocódigo)
2. Analice la complejidad espacial y temporal de su propuesta

3. De un breve ejemplo paso a paso de funcionamiento de su propuesta
4. Programe su solución. Incluya las instrucciones de compilación y/o ejecución. Brinda 2 ejemplos para probar su programa.
5. Analice: ¿La complejidad de su propuesta es igual a la de su programa?

Formato de los archivos:

El programa debe recibir por parametro tres parametros. El primero la cantidad de cuadras de este a oeste. Luego la cantidad de cuadras de norte a sur. Finalmente un archivo que contenga las ganancias por cuadras Ejemplo:

```
python tareas.py 5 4 manzanas.txt 0
```

El archivo de ganancias corresponde a un archivo de texto donde se representa en una matriz las ganancias. cada línea corresponde a una fila de manzanas. Separado por comas se encuentran la manzana de cada columna. La ganancia es un numero entero. Ejemplo:

```
100,150,300,100,50
80,100,80,120,100
200,100,90,120,100
140,60,80,90,50
```

El programa deberá mostrar por pantalla las manzanas seleccionadas según su coordenadas y la ganancia total.:

```
Manzanas: (4,5) (3,5) (3,4) (1,3)
Ganancia: 50 + 100 + 120 + 300 = 4700
```

Parte 2: La fortaleza de la red de transporte.o

Contamos con la información de las rutas que unen un conjunto de “n” ciudades entre sí. Cada ruta es de doble mano. Comienza en una ciudad y finaliza en otra. Es posible utilizando una ruta o más llegar desde cualquier ciudad a otra. El ministerio de transporte quiere saber cual es el mínimo número de rutas que en caso de cortarlas por reparación provoque una desconexión entre alguna de las ciudades.

Se pide::

1. Considerar la siguiente propuesta: “Aquella ciudad que cuenta con menor cantidad de rutas entrantes se debe considerar como la causante de debilidad de la red de transporte. Sus rutas adyacentes corresponden al mínimo buscado”. Demostrar la optimalidad o invalidez de la afirmación.
2. Independientemente del punto anterior se solicita generar una propuesta mediante redes de flujo que solucione el problema. Explicar la idea de esta.

3. Presentar pseudocódigo
4. Realizar un análisis de optimalidad.
5. Realizar análisis de complejidad temporal y espacial. Considere las estructuras de datos que utiliza para llegar a estos.
6. Programar la solución. Incluya la información necesaria para su ejecución. Compare la complejidad de su algoritmo con la del programa.
7. ¿Es posible expresar su solución como una reducción polinomial? En caso afirmativo explique cómo y en caso negativo justifique su respuesta.

Formato de los archivos

El programa debe recibir por parámetro el path del archivo de caminos entre ciudades. Ejemplo

```
python transporte.py rutas.txt
```

El formato de la línea es: ciudad origen, ciudad destino.]

Ejemplo: "rutas.txt"

```
A,B
A,D
A,E
B,C
C,D
C,E
D,E
```

Debe resolver el problema y retornar por pantalla la cantidad y que rutas no se pueden cortar. En el ejemplo:

```
Cantidad mínima de rutas: 2
Rutas: A,B B,C .
```

Parte 3: Un casting para el reality show

Contamos con un conjunto de "n" personas que conforman un grupo de un próximo reality show de supervivencia extrema. Algunas de esas personas se conocen entre sí y tienen una relación de amistad preexistente. Se desea separar a las personas en dos equipos con la condición que los que tienen amistad queden siempre en el mismo equipo. Para lograrlo se nos permite eliminar con mucho "j" personas que puedan resultar conflictivas para cumplir con el cometido. La producción del programa desea saber si dado un casting determinado es posible lograr los solicitado.

Se pide::

1. Realice un análisis teórico entre las clases de complejidad P, NP, NP-H y NP-C y la relación entre ellos.
2. Demostrar que, dada una posible solución que obtenemos, se puede fácilmente determinar si se puede cumplir o no con la tarea solicitada.
3. Demostrar que si desconocemos la solución la misma es difícil de resolver. Utilizar para eso el problema “Mínimum Node Deletion bipartite Subgraph” (suponiendo que sabemos que este es NP-C).
4. Demostrar que el problema “Mínimum Node Deletion bipartite Subgraph” pertenece a NP-C. (Para la demostración puede ayudarse con diferentes problemas, recomendamos “Clique problem”)
5. En base a los puntos anteriores a qué clases de complejidad pertenece el problema del “Casting del reality”? Justificar
6. Una persona afirma tener un método eficiente para responder el pedido cualquiera sea la instancia. Utilizando el concepto de transitividad y la definición de NP-C explique qué ocurriría si se demuestra que la afirmación es correcta.
7. Un tercer problema al que llamaremos X se puede reducir polinomialmente al problema de “Casting del reality”, qué podemos decir acerca de su complejidad?

Definiciones útiles:

Mínimum Node Deletion bipartite Subgraph: Dado un grafo G=(V,E) y un valor k entero positivo. Queremos determinar si es posible construir un grafo bipartito eliminando no más de k nodos.

Clique: Dado un grafo G=(V,E) y un valor k entero positivo. Queremos determinar si existe un subconjunto de nodos V′ de V tal que estos conformen un subgrafo completo utilizando los ejes de E de tamaño k o mayor. n