IIC2133 – Estructuras de Datos y Algoritmos

Interrogación 3

Hora inicio: 9:00 del 11 de diciembre del 2020

Hora máxima de entrega: 23:59 del 11 de diciembre del 2020

0. Responde esta pregunta en papel y lápiz, incluyendo tu firma al final. Nos reservamos el derecho a no corregir tu prueba según tu respuesta a este ítem. Puedes adjuntar esta pregunta a cualquiera de las preguntas que subas a SIDING.

    a. ¿Cuál es tu nombre completo?
    b. ¿Te comprometes a no preguntar ni responder dudas de la prueba a nadie que no sea parte del cuerpo docente del curso, ya sea de manera directa o indirecta?

Responde sólo 3 de las 4 preguntas a continuación. Si respondes las 4, se escogerá arbitrariamente cuales 3 corregir, y la otra se considerará como no entregada.

Si una pregunta pide que hagas algo en una complejidad específica, debes justificar por qué tu respuesta tiene esa complejidad. Si no, tendrás como máximo 1/3 del puntaje correspondiente.

Deberás citar tus fuentes. Si detectamos plagio tendrás un 1 en la pregunta correspondiente.

1. La agencia publicitaria para la que trabajas ha sido contratada por XIAOMI MEJOR RELACIÓN PRECIO CALIDAD para posicionar letreros publicitarios en la ruta de Arica a Punta Arenas. Para esto tienes una lista de $\boldsymbol{n}$ ubicaciones de letreros disponibles, con sus distancias $\boldsymbol{d_1}, \boldsymbol{d_2}, \dots, \boldsymbol{d_n}$ medidas desde Arica y ordenadas crecientemente, y una estimación de que el *i*-ésimo letrero es visto por $\boldsymbol{w_i}$ personas al día.

    El cliente quiere maximizar la cantidad de personas que ven los letreros al día. Tú debes tener en cuenta que las autoridades no permiten que haya dos letreros del mismo producto o compañía a una distancia menor a $\boldsymbol{k}$ entre ellos.

    ¿Cuáles ubicaciones de letreros debes utilizar para esto?

    a) Demuestra que este problema tiene subestructura óptima.
    b) Supón la siguiente estrategia codiciosa para resolver el problema: recorrer la lista de ubicaciones de letreros de Arica a Punta Arenas, y poner un letrero en cada ubicación que cumpla con la restricción de las autoridades con respecto a la distancia con el último letrero escogido. Demuestra que, si todos los letreros tienen el mismo $\boldsymbol{w}$, esta estrategia es óptima.

2. La gente de la tierra de Omashu se toma los grupos de amigos muy en serio. Tan en serio, que podemos describirlos matemáticamente (son clases de equivalencia):

   - Si a es amigo de b, entonces b es amigo de a
   - Cada persona es amiga de así misma y cada persona pertenece a un solo grupo de amigos
   - Si a forma parte del grupo X, y b es amigo de a, entonces necesariamente b forma parte de X.

   a) Dada una lista F de pares de forma (a, b) que indican amistad entre la persona a y la persona b, describe un algoritmo lineal en el número de pares que calcule la cantidad de grupos de amigos distintos que existen en Omashu.
   b) Además de la lista F anterior, se te da una lista U con tríos de la forma (a, b, w). Cada trío indica que a y b no son amigos, pero podrían serlo si se les paga una cantidad positiva w de dinero. Describe un algoritmo a lo más *linearítmico* (es decir, del tipo *n* log*n*) que calcule el costo mínimo necesario para que todos los habitantes de Omashu formen un solo gran grupo de amigos.

3. Considera el algoritmo de Bellman-Ford para un grafo dirigido, con costos y sin ciclos de costo acumulado negativo:

```
bellman − ford(s):
    for each u ∈ V:
        d[u] ← ∞
        π[u] ← null
    d[s] ← 0
    for k = 1..|V| − 1:
        for each (u, v) ∈ E:
            if d[v] > d[u] + w(u, v):
                d[v] ← d[u] + w(u, v)
                π[v] ← u
```

   Este algoritmo calcula las rutas de menor costo desde el vértice s a todos los demás vértices del grafo en tiempo $\Theta(V \cdot E)$. Queremos mejorar su rendimiento.

   Sea $f(v)$ el número de aristas de la ruta de menor costo de s a v.

   a) Definimos L como el máximo $f(v)$ entre todos los posibles v. Modifica el algoritmo de Bellman-Ford para que su complejidad sea $\Theta(L \cdot E)$ para cualquier grafo.
   b) Sea $g(v)$ la cantidad de aristas que llegan a v. Modifica el algoritmo de Bellman-Ford para que el tiempo que tome esté dado por la siguiente expresión:

$$T(V, E) = \sum_{v \in V} f(v) \cdot g(v)$$

4. Se acercan las fiestas, y el generoso Krampus te presenta $\boldsymbol{n}$ regalos, de los cuales debes escoger $\boldsymbol{k}$**.** Cada regalo trae la etiqueta con su precio, y como eres una persona materialista, quieres maximizar la suma de los precios de los regalos que elijas. Los regalos están dispuestos en la forma de un árbol binario de navidad (no de búsqueda), donde cada nodo corresponde a un regalo. Hay una sola restricción para los regalos que puedes escoger: si escoges el regalo $\boldsymbol{u}$**,** necesariamente debes escoger el padre de $\boldsymbol{u}$ en el árbol, y así sucesivamente.

   Escribe un algoritmo de programación dinámica que indique los $\boldsymbol{k}$ regalos que debes escoger de manera de maximizar el precio total.