Disclaimer: respuestas válidas para el primer cuatrimestre de 2024. Puede ser que alguna complejidad de algún algoritmo cambie en otra versión de la materia, o que se vean otros algoritmos y alguna respuesta cambie por eso, etc.

Versión 1.1: Se agregó una opción correcta a la pregunta 4. En la pregunta 5, las X en verde indican que se considera correcto marcar exactamente una de las dos opciones indicadas.

Versión 1.1b: Se numeran las opciones para poder seguir la devolución

Técnicas de Diseño de Algoritmos
Primer Cuatrimestre 2024
2do Parcial (T. Mañana)

Nombre y Apellido	L.U.	N°Orden
.....	.....	.....

*Duración: **3 horas**. Este examen es a **libro cerrado**. Todas las preguntas tienen $k > 1$ opciones correctas. Las respuestas donde se marquen exactamente esas k dan 2 puntos, aquellas donde se marquen al menos $\lceil k/2 \rceil$ correctas y más correctas que incorrectas dan 1 punto, las que tengan menos de $\lceil k/2 \rceil$ correctas o igual o más incorrectas que correctas dan 0, y si todas las marcadas son incorrectas dan $-0{,}5$ puntos. Para aprobar el parcial se deben sumar $p \geq 10$ puntos y la nota será $p/2$.*

Pregunta 1 Sea $G = (V, E)$ un grafo conexo, $f\colon E \to \mathbb{R}$ una función de peso de G, $v \in V$ y $e \in E$. ¿Cuáles de estas afirmaciones son verdaderas?

1 ☐ Sea $\alpha \in \mathbb{R}$ y $f_\alpha\colon E \to \mathbb{R}$ tal que $f_\alpha(e) = \alpha f(e)$. Para cualquier valor de α, si T es un AGM de G respecto de f_α, entonces T es un AGM de G respecto de f.

2 ☐ Si e pertenece a todo árbol generador mínimo de G, entonces e es puente.

3 ☒ Si e tiene costo estrictamente menor que cualquier otra arista de G, entonces e pertenece a todo árbol generador mínimo de G.

4 ☐ Todo camino mínimo de G es parte de algún AGM de G.

5 ☐ Si los pesos de todas las aristas son positivos, siempre existe un árbol de caminos mínimos desde v que es a su vez un AGM de G.

6 ☒ Si e incide en v y tiene el menor costo entre todas las aristas incidentes en v, entonces e está en algún árbol generador mínimo de G.

7 ☒ Si todas las aristas de G tienen el mismo peso, entonces cualquier árbol de caminos mínimos desde v es un AGM.

8 ☒ Si todas las aristas de G tienen peso distinto, entonces el AGM de G es único.

Pregunta 2 Marque las opciones que sean verdaderas para el algoritmo de árbol generador mínimo de Kruskal, si se toma como entrada un grafo con n vértices y m aristas,

1 ☐ La implementación que vimos en la materia es un algoritmo de programación dinámica.

2 ☐ No es posible implementarlo para ejecutar en $O(n^2)$

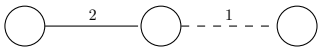
3 ☒ Es posible implementarlo para ejecutar en $O(m \log n)$.

4 ☒ La implementación que vimos en la materia utiliza un Disjoint Set (Union-Find).

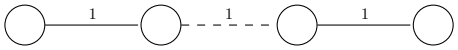
5 ☒ La implementación que vimos en la materia utiliza algún algoritmo de ordenamiento que para un vector de tamaño k tome tiempo $O(k \log k)$

6 ☐ La implementación que vimos en la materia utiliza un Heap binario.

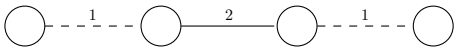
Pregunta 3 A continuación se muestran algunos grafos, marcando algunos de los ejes con líneas punteadas y el resto con líneas sólidas. Responder, para cada caso, cuál algoritmo de árbol generador mínimo visto en clase es posible que se esté ejecutando, sabiendo que, por ahora, seleccionó a las aristas sólidas.

1. 

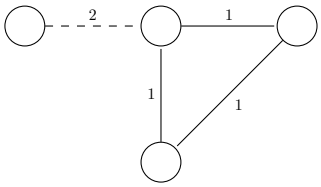
1 [X] Prim 2 [ ] Kruskal 3 [ ] Ninguno

2. 

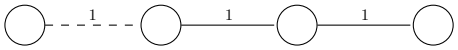
4 [ ] Prim 5 [X] Kruskal 6 [ ] Ninguno

3. 

7 [ ] Prim 8 [ ] Kruskal 9 [X] Ninguno

4. 

10 [ ] Prim 11 [ ] Kruskal 12 [X] Ninguno

5. 

13 [X] Prim 14 [X] Kruskal 15 [ ] Ninguno

Pregunta 4 ¿Cuáles de las siguientes afirmaciones acerca de camino mínimo son correctas?

1 [X] El algoritmo de Dijkstra tiene implementaciones con complejidades distintas que puede convenir utilizar dependiendo de si el grafo de entrada es ralo o denso.

2 [ ] Para un grafo G, y v, w vértices de $V(G)$, si hay un camino de v a w cuya primera arista es de mínimo peso entre las que inciden en v, entonces esa arista pertenece a algún camino mínimo entre v y w.

3 [X] Si un digrafo D no posee ciclos, se puede resolver el problema de camino mínimo desde un vértice de D en tiempo lineal.

4 [X] Un algoritmo correcto para resolver camino máximo en un grafo G es invertir los signos de los pesos de las aristas y calcular camino mínimo.

5 [ ] Si un grafo G tiene aristas de peso negativo, es posible resolver camino mínimo con Dijkstra si primero se suma el módulo del peso de la arista mínima a todas las aristas.

Sea f_p una función de peso en las aristas de un digrafo D que no genera ciclos negativos, y $d(x, y)$ la distancia de un vértice x a un vértice y respecto de f_p, dados los vértices v, w y una arista $u \to z$ de D,

6 [ ] Si $d(v, w) = d(v, u) + f_p(u \to z) + d(z, w)$, entonces $u \to z$ pertenece a todo camino mínimo de v a w en G.

7 [X] Nunca es posible que $d(v, w) > d(v, u) + f_p(u \to z) + d(z, w)$.

Pregunta 5 ¿Cual(es) de los algoritmos de camino mínimo con la implementación que vimos en la materia es el más conveniente en complejidad temporal para resolver cada uno de estos escenarios?

Calcular los pesos de los caminos mínimos desde un vértice v de un grafo G donde todas las aristas tienen peso igual a 42.

1 [X] BFS 2 [ ] Dijkstra 3 [ ] Bellman-Ford 4 [ ] Ninguno de estos

Calcular los pesos de los caminos mínimos desde un vértice v de un grafo G con aristas de peso negativo pero sin ciclos de peso negativo.

5 [ ] BFS 6 [ ] Dijkstra 7 [X] Bellman-Ford 8 [ ] Ninguno de estos

Calcular los pesos de los caminos mínimos desde un vértice v de un grafo G con ciclos de peso negativo.

9 [ ] BFS 10 [ ] Dijkstra 11 [X] Bellman-Ford 12 [X] Ninguno de estos

Resolver un sistema de inecuaciones con variables $x_1, \ldots, x_n$ y desigualdades de la forma $x_i - x_j \leq C_{ij}$, $C_{ij} \in \mathbb{R}$.

13 [ ] BFS 14 [ ] Dijkstra 15 [X] Bellman-Ford 16 [ ] Ninguno de estos

Calcular los pesos de los caminos mínimos desde un vértice v de un grafo G en el que todas las aristas tienen peso positivo.

17 [ ] BFS 18 [X] Dijkstra 19 [ ] Bellmann-Ford 20 [ ] Ninguno de estos

Pregunta 6 La gran empresa de E-Commerce MerladoCibre está desplegando una plataforma para hacer envíos por todo el país. En cada ciudad donde está instalada, tiene un depósito principal y un conjunto de $n-1$ centros de distribución $d_1, \ldots, d_{n-1}$. Cada vez que un paquete es enviado a una persona, éste sale del depósito principal d_0 y va pasando por distintos centros de distribución hasta alcanzar el más cercano a la dirección indicada. MerladoCibre tiene definidas m rutas directas para enviar un paquete entre dos de sus instalaciones d_i, d_j $(0 \leq i \neq j \leq n-1)$; no necesariamente se tiene la ruta directa de d_i a d_j para todo i, j. Es más, algunas de estas rutas fueron incorporadas recientemente y están marcadas como "experimentales". Para cada ruta directa se tiene calculado el costo de enviar un paquete por la misma, dependiente de factores como el almacenamiento, el combustible, la logística, etc. MerladoCibre quiere determinar, para un centro de distribución d_t dado, el costo mínimo para enviar un paquete del depósito d_0 hasta d_t combinando rutas directas, pero, por seguridad, tiene la política de que toda combinación posible debe pasar por a lo sumo una ruta experimental. Si armamos un digrafo G con vértices $d_0, \ldots, d_{n-1}$ y una arista $d_i \to d_j$ por cada ruta directa del centro d_i al centro d_j, más una función de peso $f_p \colon E \to \mathbb{R}$ tal que $f_p(d_i \to d_j)$ es el costo del envío de d_i a d_j ¿Cuál de estas estrategias sirve para resolver el problema de MerladoCibre?

1 ☐ Cambiar la función de peso f_p por $f'_p \colon E \to (\mathbb{R}, \{0,1\})$ tal que para $0 \leq i \neq j \leq n-1$, $f'_p(d_i \to d_j) = (f_p(d_i \to d_j), x)$ donde $x = 1$ si $d_i \to d_j$ es una ruta experimental. Utilizar Dijkstra comparando lexicográficamente f'_e (es decir, si hay empate por costo se desempata por menor cantidad de aristas experimentales). Si el resultado pasa por más de una ruta experimental, no es posible llegar pasando a lo sumo por una ruta experimental. En caso contrario, el costo en el par obtenido es la respuesta.

2 ☒ Generar un grafo G' con $2n-1$ vértices $d_0^0, \ldots, d_{n-1}^0, d_1^1, \ldots, d_{n-1}^1$. Por cada arista no experimental $d_i \to d_j$, poner en G' $d_i^0 \to d_j^0$ y $d_i^1 \to d_j^1$, y por cada arista experimental $d_i \to d_j$, poner en G' la arista $d_i^0 \to d_j^1$. Calcular el vector de distancias D desde d_0 en G'. La respuesta es el mínimo entre $D[d_t^0]$ y $D[d_t^1]$. Si ambos valores son ∞, entonces no es posible llegar pasando por a lo sumo una ruta experimental.

3 ☒ Obtener un grafo G' quitando de G todas las aristas correspondientes a rutas experimentales. Calcular en G' el vector de distancias D desde d_0 con Dijkstra, transponer G' y calcular el vector de distancias D^T desde d_t. Luego, si X es el conjunto de aristas experimentales, usar los vectores de distancias calculados para obtener $a = \min_{d \to q \in X}\{D[d] + f_p(d \to q) + D^T[q]\}$. La respuesta es el mínimo entre a y $D[d_t]$; si ese valor es ∞ no es posible llegar pasando por a lo sumo una ruta experimental.

4 ☐ Agregar un vértice x a G. Reemplazar toda arista $d_i \to d_j$ de G por dos aristas $d_i \to x$ y $x \to d_j$. Calcular el vector de distancias D desde d_0 con Dijkstra. La respuesta es $D[d_t]$. Si $D[d_t] = \infty$, entonces no es posible llegar pasando por a lo sumo una ruta experimental.

Sin tener más información sobre la cantidad de rutas directas entre centros de distribución, ¿Cual es la complejidad temporal más ajustada para la opción viable más eficiente? (Suponiendo la implementación vista en la materia de los algoritmos)

5 ☒ $O(\min\{m \log n, n^2\})$ 6 ☐ $O(nm)$ 7 ☐ $O(n+m)$ 8 ☐ $O(n^3)$ 9 ☐ $O(\min\{m \log n, nm\})$

Pregunta 7 ¿Cuáles de estas afirmaciones sobre el problema de flujo máximo en una red N son verdaderas?

1 ☐ Si la capacidad de cada arista de N es impar, entonces existe un flujo máximo en el cual el flujo sobre cada arista de N es impar.

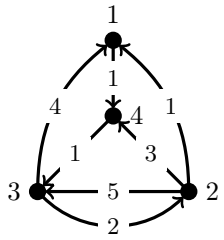
2 ☐ Si la capacidad de cada arista de N es impar, entonces el valor del flujo máximo es impar.

3 ☒ Si la capacidad de cada arista de N es par, entonces existe un flujo máximo en el cual el flujo sobre cada arista de N es par.

4 ☒ Si la capacidad de cada arista de N es par, entonces el valor del flujo máximo es par.

5 ☒ Si todas las aristas de N tienen capacidad racional, entonces el valor del flujo máximo es racional.

Pregunta 8 Para el siguiente grafo se realizó una ejecución parcial de Floyd-Warshall (se paró justo al terminar una ejecución del ciclo exterior, i.e., la matriz se recorrió completa una cantidad $k \leq n$ veces), para la implementación que vimos en la materia. Las filas y las columnas están en el orden de las etiquetas de los vértices.



$$M = \begin{pmatrix} 0 & \infty & \infty & 1 \\ 1 & 0 & 5 & 2 \\ 3 & 2 & 0 & 4 \\ \infty & \infty & 1 & 0 \end{pmatrix}$$

¿Cuál es el valor de k?

1 ☐ 1 2 ☒ 2 3 ☐ 3 4 ☐ 4

¿Cuál será el valor en la posición $M_{4,2}$ al finalizar la ejecución del algoritmo?

5 ☐ ∞ 6 ☐ 4 7 ☒ 3 8 ☐ No es posible saberlo

Pregunta 9 Las plataformas de video no paran de mostrar publicidades del nuevo juego para celulares llamado *Diamond Maniacs*. En este juego existen diamantes de k colores distintos; al principio de cada nivel se nos proveen $d_i \geq 0$ diamantes del color i, y el objetivo es obtener una cantidad $q_i \geq 0$ de diamantes de ese color, donde $1 \leq i \leq k$ y $\sum_{1 \leq i \leq k} d_i = \sum_{1 \leq i \leq k} q_i = T$. Durante la partida contamos con algunos botones de intercambio $i \to j$, que nos permiten cambiar un diamante de color i por uno de color j. Cada uno de los botones se puede tocar una cantidad limitada $b_{ij} \geq 0$ de veces durante la partida. Si nos quedamos sin botones que podamos activar (porque agotamos la cantidad de tocadas o porque todos los botones requieren diamantes de colores que no tenemos) y no logramos el objetivo, perdemos. Hastiada de ver a la gente de la publicidad jugar pésimamente y perder, Ludmila se propone mostrar que los niveles de *Diamond Maniacs* se pueden resolver planteándolos como problemas de flujo. ¿Cómo se debe armar una red donde el flujo máximo represente la solución a este problema (de haberla)? Para todas las opciones, siempre se tiene un vértice fuente s y un sumidero t.

1 ☒ Se pone un vértice i por cada color $i \leq k$ y un vértice v_{ij} por cada par $i \neq j$. Hay una arista $s \to i$ con peso d_i y una arista $j \to t$ con peso q_j para todo $i, j \leq k$. Por cada par i, j, hay una arista de i a v_{ij} y una arista de v_{ij} a j, ambas de capacidad b_{ij}.

2 ☐ Se ponen dos vértices i_1, i_2 por cada color i, y un vértice v_{ij} por cada par $i \neq j$. Hay una arista $s \to i_1$ con capacidad d_i, una arista $j_2 \to t$ con capacidad q_j, aristas $i_1 \to v_{ij}$ y $v_{ij} \to j_2$ con capacidad b_{ij}.

3 ☒ Se pone un vértice i por cada color $i \leq k$. Hay una arista $s \to i$ con peso d_i y una arista $j \to t$ con peso q_j para todo $i, j \leq k$. Además, hay una arista $i \to j$ con capacidad b_{ij} para cada par $i \neq j$.

4 ☐ Se ponen dos vertices i_1, i_2 por cada color $i \leq k$. Para cada $i \leq k$ hay una arista $s \to i_1$ con capacidad d_i, una arista de $i_2 \to t$ con capacidad q_i, y para cada par $i \neq j$ hay una arista $i_i \to j_2$ con capacidad b_{ij}.

De los algoritmos que consideran correctos: ¿Cuál es la complejidad temporal ajustada del más eficiente? Suponiendo las implementaciones vistas en la materia de los algoritmos.

5 ☐ $O(\text{mín}\{k^2, T^2\})$ 6 ☒ $O(\text{mín}\{k^2T, k^5\})$ 7 ☐ $O(\text{mín}\{kT, k^5\})$ 8 ☐ $O(\text{mín}\{k^3T, k^6\})$

Pregunta 10 En ajedrez, el caballo se mueve en forma de "L", dos celdas en una dirección y una celda en dirección perpendicular (ver figura 1). Se tiene un tablero de ajedrez de N por N celdas, con K caballos ubicados en algunas de sus celdas. Se quiere retirar una mínima cantidad de caballos de manera tal que, entre los caballos restantes, no queden dos que se ataquen mutuamente. Decimos que un caballo ataca a otro si puede moverse a la celda que ocupa. ¿Cuáles de los siguientes algoritmos son correctos?

1 ☐ Si hay más caballos en casilleros blancos que negros, saco todos los caballos en casilla negra, en caso contrario, saco todos los que están en una casilla blanca.

2 ☒ Represento el problema con un grafo donde cada vértice representa a un caballo en el tablero, y las aristas unen pares de caballos que se atacan mutuamente. El problema es equivalente al de encontrar un conjunto mínimo de vértices que cubran todas las aristas del grafo (*vertex cover*). Este grafo admite solución a ese problema con un algoritmo de flujo.

3 ☒ Para cada subconjunto de caballos, ver si cumple que al quitarlos no queda ningún par de caballos que se ataquen. Retornar, de los subconjuntos que cumplen, alguno de tamaño mínimo.

4 ☐ Represento el problema con un grafo donde tengo un vértice por cada caballo en el tablero y además un vértice por cada posición en el tablero. Cada caballo tiene una arista a su posición en el tablero y a las posiciones que puede atacar. El problema es equivalente al de encontrar un conjunto mínimo de aristas que cubran todos los vértices del grafo (*edge cover*). Este grafo admite solución a ese problema con un algoritmo de flujo.

De los algoritmos que consideran correctos: ¿Cuál es la complejidad temporal ajustada del más eficiente? Suponiendo las implementaciones vistas en la materia de los algoritmos.

5 ☒ $O(K^2)$ 6 ☐ $O(K^3)$ 7 ☐ $O(N^2)$ 8 ☐ $O(2^K * K^2)$ 9 ☐ $O(K)$

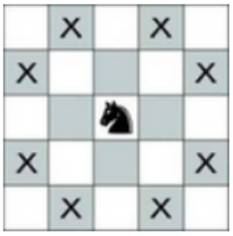


Figura 1: Movimientos del caballo en ajedrez.