MA3705. Algoritmos Combinatoriales 2020.
Profesor: José Soto
Profesores Auxiliares: Antonia Labarca, Víctor Saez

Tarea 1.

Fecha entrega: Martes 6 de Octubre, 23:59. Por u-cursos.

Instrucciones:

1. **Extensión máxima**: Entregue su tarea en a lo más **12 planas**.
2. **Formato:** La tarea debe entregarse en formato pdf, con fondo de un solo color (blanco de preferencia). *No se aceptarán escaneos en .jpg u otro formato de imágenes.*
   - Si desarrolla su tarea en papel, entréguelo escaneados o en fotos de alta calidad y convertido a un archivo pdf único.
   - Puede desarrollar su tarea en algún formato manuscrito digital (tablet u otro instrumento), pero la salida debe ser un archivo .pdf único.
   - Si entrega su tarea (o parte de ella) tipeada, dicha parte debe estar escrita en Latex y *debe adjuntar el archivo .tex y cualquier archivo fuente adicional que usó para generar el pdf.* Debe subir estos archivos en ucursos adicionalmente al pdf. **No podrá adjuntar links a sistemas de edición externa como overleaf**.
3. **Tiempo de dedicación:** El tiempo estimado de *desarrollo* de la tarea es de 7 horas de dedicación. Esto no considera el tiempo de estudio previo, el tiempo dedicado en asistir a cátedras y auxiliares, ni el tiempo para *ponerse al día*. Tendrá un plazo de 14 días para entregarlo *sin tiempo adicional*. No espere hasta el último momento para escanear o fotografiar adecuadamente su tarea y cambiarla al formato solicitado (pdf). Entregue con suficiente anticipación a la hora límite. Note que ucursos permite sobreescribir lo que ya ha subido así que puede entregar a medida que vaya completando su tarea.
4. **Política de honestidad y colaboración** Algunos problemas tienen colaboración autorizada, para entender los límites y reglas al respecto usted debe leer la política de honestidad y colaboración del curso: `https://www.u-cursos.cl/ingenieria/2020/2/MA3705/1/blog/` Recuerde que si decide colaborar en un problema debe anotar el nombre de la(s) persona con las que colaboró al principio de su solución y que se revisará simetría por problema.
5. **Revisión:** Se podrá descontar hasta 1 punto en la nota final por falta de formato o extensión.

Definiciones usadas en esta tarea

Para los problemas 1, 2 y 3 usaremos las definiciones usadas en el documento llamado *Criptomorfismos en matroides* [1] ubicado en la sección de enlaces del curso o en el link al final de este documento.

Un grafo $G = (V, E)$ es bipartito si existe una partición L, R de V tal que $E = E[L, R]$, es decir cada arista tiene un extremo en L y el otro en R. Es claro (puede usarlo sin demostrar) que G es bipartito si y solo si todas sus componentes conexas inducen grafos bipartitos.

Decimos que v es alcanzable desde u en un grafo dirigido $G = (V, E)$ si existe un camino dirigido en G de u a v (es decir, una secuencia finita $u_0, u_1, \ldots, u_k$ de vértices distintos con $u_0 = u$, $u_k = v$, $(u_{i-1}, u_i) \in E$, para $i \in [k]$.

Problema 1. Colaboración autorizada (8 puntos)

Sea $(S, \mathcal{B})$ un clutter no vacío que satisface el axioma de intercambio (es decir, satisface las propiedades de la sección 3.1 de [1]). Defina

$$\mathcal{I} = \{X \subseteq S\colon\ \exists B \in \mathcal{B}, X \subseteq B\}$$
$$\mathcal{B}' = \{X \in \mathcal{I}\colon\ \forall e \in S \setminus X, X + e \notin \mathcal{I}\}$$

Demuestre que $(S, \mathcal{I})$ es el sistema de independencia de una matroide (es decir, satisface cualquiera de las definiciones 2.1; 2.2 o 2.3 de [1]) y que su conjunto de bases $\mathcal{B}'$ satisface $\mathcal{B}' = \mathcal{B}$.

Problema 2 Colaboración autorizada (8 puntos)

Sea $(S, \mathcal{C})$ un clutter que no contiene al vacío y que satisface el axioma de eliminación de circuitos (es decir, satisface las propiedades de la Sección 4.1 de [1]). Defina

$$\mathcal{I} = \{X \subseteq S\colon \forall C \in \mathcal{C}, C \not\subseteq X\}$$
$$\mathcal{C}' = \{X \in 2^S \setminus \mathcal{I}\colon \forall e \in X, X - e \in \mathcal{I}\}$$

Demuestre que $(S, \mathcal{I})$ es el sistema de independencia de una matroide (es decir, satisface cualquiera de las definiciones 2.1; 2.2 o 2.3 de [1]) y que su conjunto de circuitos $\mathcal{C}'$ satisface $\mathcal{C}' = \mathcal{C}$.

Problema 3 Colaboración autorizada (8 puntos)

Sea $r\colon 2^S \to \mathbb{N}$ una función que satisface las propiedades de normalización, aumentos unitarios y submodularidad débil (es decir, satisface las propiedades de la Sección 5.2 de [1]) Defina

$$\mathcal{I} = \{X \subseteq S\colon r(X) = |X|\}$$
$$r'\colon 2^S \to \mathbb{N}$$
$$r'(X) = \text{máx}\{|I|\colon I \in \mathcal{I}, I \subseteq X\}$$

Demuestre que $(S, \mathcal{I})$ es el sistema de independencia de una matroide (es decir, satisface cualquiera de las definiciones 2.1; 2.2 o 2.3 de [1]) y que su función de rango r' satisface $r' = r$.

Problema 4: (10 puntos)

En clases se verá una variante de BFS (o DFS) para encontrar los vértices alcanzables desde un vértice u en un grafo **dirigido** $G = (V, E)$. Puede usar dicho algoritmo o alguna modificación en este ejercicio.

Sea G un grafo no dirigido cuyas aristas han sido particionadas en s colores: $E_1 \cup E_2 \cup \cdots \cup E_s = E$. (Puede suponer que dado e su color es $c(e) = j \in [s]$). Un paseo alternante de u a v es un paseo de u a v cuyas aristas $e_1 e_2 \ldots e_k$ son tales que dos aristas consecutivas no son del mismo color.

(a) 3 puntos: Demuestre que si $s = 2$ y G es bipartito entonces el paseo alternante más corto de u a v es un camino.

(b) 3 puntos: Demuestre que para todo $s \geq 2$ y G cualquiera (no necesariamente bipartito), el paseo alternante más corto de u a v pasa por cada vértice a lo más 2 veces.

(c) 4 puntos: (no relacionado con las partes anteriores). Sea G un grafo conexo cualquiera, $s \geq 2$, y $u \in V(G)$. Diseñe un algoritmo de complejidad $O((n+m)s)$ que encuentre el conjunto W de todos los vértices v para los cuales existe un paseo alternante de u a v en el grafo.

Pregunta extra 1 (Difícil, hasta 10 puntos extra. Si la hace se debe entregar por correo por separado) Encuentre un algoritmo de complejidad $O(n+m)$ para la parte (c).

Problema 5: (10 puntos)

(a) 3 puntos: Sean a, b, c tres vértices de un grafo G tal que $bc \in E$. Sean P un camino de a a b; y Q un camino de a a c. Demuestre que si P y Q tienen la misma paridad (ambos de largo par; o ambos de largo impar) entonces G tiene un ciclo de largo impar.

(b) 3 puntos: Demuestre que G es bipartito si y solo si G no tiene ciclos de largo impar.
Indicación: Para la dirección complicada use la parte anterior.

(c) 4 puntos: Sea G un grafo *no necesariamente conexo*. Diseñe un algoritmo de complejidad $O(n + m)$ para determinar si G es bipartito o no, y que en el caso que G sea bipartito reporte la bipartición L, Rs.
Indicación: Debe demostrar correctitud y esbozar la complejidad.

Problema 6: (16 puntos)

El objetivo de este problema es demostrar correctitud y estudiar la complejidad del algoritmo de Borůvka visto en clases para encontrar un árbol generador de peso mínimo en un grafo conexo. De cátedra ya sabemos que si el grafo entregado al final (V, F) es acícliclo entonces es un MST, por lo que para correctitud solo debemos probar que el grafo entregado es acícliclo. Para simplificar la exposición, suponga que G es conexo y que $c\colon E \to \mathbb{R}$ es una función de costos **inyectiva**.

Llame $G_i = (V, F_i)$ al grafo al principio de la iteración i y sea $\mathcal{U}_i$ el conjunto de componentes conexas de G_i. En particular $\mathcal{U}_1 = \{\{v\}\colon v \in V\}$). El algoritmo elige para cada componente $U \in \mathcal{U}_i$ una arista $e_U = uv$ (la de menor peso en $\delta(U)$, que es única por inyectividad). Llame $f(e_U) \in \mathcal{U}_i$ a la componente a la cual pertenece el extremo de e_U que está fuera de U. Defina entonces $G'_i = (\mathcal{U}_i, F'_i)$ como el multigrafo sin loops cuyos vértices son las componentes en $\mathcal{U}_i$ y donde para cada $U \in \mathcal{U}_i$ se agrega una arista $\tilde{U} := U f(e_U)$ a F'_i. Por simplicidad extienda la función de costos como: $c(\tilde{U}) = c(e_U)$.

(a) 3 puntos: Demuestre que los únicos ciclos del multigrafo G'_i tienen largo 2 (es decir son pares aristas paralelas). Más aún demuestre que si $\tilde{U}$ es una arista paralela a $\tilde{W}$ en H_i entonces $e_U = e_W$. **Indicación:** Parta suponiendo que $\tilde{U}$ es la arista más liviana de un ciclo.

(b) 3 puntos: Sea $G''_i = (\mathcal{U}_i, F''_i)$ el grafo simple obtenido de G'_i al mantener solo una arista de cada par de aristas paralelas de G'_i. De la parte anterior se obtiene que G''_i no tiene ciclos. Use esto para demostrar por inducción que $G_i = (V, F_i)$ no tiene ciclos y concluya que el algoritmo entrega un árbol generador al final.

Indicación: Demuestre que si (V, F_i) es acíclico pero (V, F_{i+1}) tiene un ciclo C entonces se puede usar C para encontrar un ciclo en G''_i.

(c) 7 puntos: Necesitamos dar una implementación más precisa para cada iteración del algoritmo. Para esto supongamos sin pérdida de generalidad que $V = [n]$.

Defina para cada vértice u el valor $f_i(u) = \min\{v\colon v \text{ está en la misma componente que } u \text{ en } G_i\}$.

Describa un algoritmo que dado (V, F_i) calcule $f_i(u)$ para cada $u \in U$ en tiempo $O(n + |F_i|)$.

Describa un segundo algoritmo que dado E, F_i, la función de costo c y todos los $f_i(u)$ calcule el conjunto de aristas $\{e_U\colon U \in \mathcal{U}_i\}$ a agregar en la iteración i-ésima en tiempo $O(n + m) = O(m)$

Use ambos algoritmos para concluir que (V, F_{i+1}) se puede calcular a partir de (V, F) en tiempo $O(m)$.

(c) 3 puntos: Demuestre que el número de iteraciones del algoritmo es $O(\log n)$ y concluya de las partes anteriores que el algoritmo de Borůvka se puede implementar en tiempo $O((n + m) \log n) = O(m \log n)$.

Pregunta extra 2 (Difícil, hasta 6 puntos extra. Si la hace se debe entregar por correo por separado).

Recuerde que el algoritmo de Prim puede encontrar el MST de un grafo en tiempo $O(n \log n + m)$. Combine los algoritmos de Prim y Borůvka para obtener un algoritmo que calcule un MST de G en tiempo $O(m \log \log n)$. Indicación: Haga cierto número de iteraciones de Borůvka y luego adapte Prim para terminar.

Referencias

[1] Soto, J.A. *Criptomorfismos en Matroides*, Material curso MA3705, `https://www.overleaf.com/read/tsnrcqwpbjtd`