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

Tarea 3.

Fecha entrega: Jueves 3 de Diciembre, 23:59. Por u-cursos.

Instrucciones:

1. **Extensión máxima**: Entregue su tarea en a lo más **14 planas**.
   Indicación relevante: No copie el enunciado de la tarea en su desarrollo.

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 8 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.

La tarea tiene un total de 72 puntos, pero se corregirá bajo un máximo de 60. Es decir, si su puntaje es P, su nota será (10+$\min(P, 60)$)

Problema 1 Colaboración autorizada (20 puntos)
Orientaciones eulerianas y matchings perfectos en grafos bipartitos regulares

Sabemos del teorema de Hall que todo grafo bipartito regular tiene un matching perfecto que puede ser encontrado, usando caminos aumentantes, en tiempo $O(n(n+m))$. En este problema diseñaremos un algoritmo más rápido para encontrar matchings perfectos para grafos bipartitos regulares.

Recordamos que todo grafo bipartito regular tiene exactamente $n/2$ vértices en cada lado y $n\Delta/2$ aristas, donde Δ es el grado común. Primero necesitaremos estudiar grafos eulerianos y orientaciones eulerianas.

Un multigrafo $G=(V,E)$ sin loops y con todos sus vértices de grado par (digamos $2k$) se llama Euleriano. Una orientación euleriana de G es un digrafo (no necesariamente simple), $D=(V,\vec{E})$ donde cada arista $e=uv$ ha sido orientada (es decir reemplazada o bien por el arco (u,v) o bien por el arco (v,u)) de tal forma que cada vértice v tiene exactamente k arcos de $\vec{E}$ entrando a v y k arcos de $\vec{E}$ en v.

El siguiente algoritmo entrega una orientación euleriana de G.

Algoritmo 1: Orientación Euleriana

Entrada: (multi)grafo G euleriano de grado común $2k$
$W \leftarrow V$ `// Vértices con aristas incidentes aun no orientadas`
$\vec{E} \leftarrow \emptyset$ `// orientación`
mientras $W \neq \emptyset$ **hacer**
  Elegir $v \in W$.
  mientras $\delta(v) \neq \emptyset$ **hacer**
    Sacar (y eliminar) una arista $e=vw$ de $\delta(v)$
    Agregar arco (v,w) a $\vec{E}$
    si $\delta(v)$ *quedó vacía* **entonces** $W \leftarrow W - v$
    $v \leftarrow w$.
  fin
fin
devolver $(V,\vec{E})$

a) **(4 puntos)** Argumente en no más de 2 párrafos sobre la correctitud del algoritmo (específicamente que si el algoritmo termina entonces entrega lo pedido). Demuestre además que el algoritmo termina en tiempo $O(n+m)$ y si el grafo tiene aristas, esto es $O(m)$.

b) **(4 puntos)** Sea ahora G_t un (multi)grafo bipartito 2^t-regular. Demuestre, usando la parte anterior, que en tiempo $O(n2^{t-1})$ puede encontrar un sub(multi)grafo G_{t-1} bipartito, con el mismo conjunto de vértices, y 2^{t-1} regular.

c) **(4 puntos)** Diseñe un algoritmo que encuentre un matching perfecto de G_t en tiempo $O(n2^{t-1})$.

Sea ahora $G=(V=L\cup R,E)$ un (multi)grafo bipartito Δ-regular con $2^{t-1}<\Delta<2^t$. Sea además M un matching perfecto cualquiera del grafo bipartito completo $(L\cup R,\{\{\ell,r\}\colon \ell\in L, r\in R)$. Para encontrar M puede por ejemplo, enumerar los vértices de L de 1 a $n/2$ y los vértices de R de $n/2+1$ a n y tomar $M=\{\{i,i+n\}\colon i\in n/2\}$. Diremos que una arista f de M es **falsa** si no existe arista e en G paralela a f (o sea conectando los mismos vértices)

d) **(4 puntos)** Supongamos que M tiene exactamente k aristas falsas. Considere el multigrafo 2^t-regular $H=H_{G,M}$ obtenido al agregar $(2^t-\Delta)$ copias de M a G. Diseñe un algoritmo que encuentre un matching perfecto M' de H con a lo más $\lfloor k/2\rfloor$ aristas falsas en tiempo $O(n2^t)$.

   Indicación: Llame falsas también a las copias de las aristas falsas. Modifique el algoritmo de la parte c de manera adecuada para que devuelva M'.

e) **(4 puntos)** Use las partes anteriores para diseñar un algoritmo que calcule un matching de un grafo Δ regular (para cualquier Δ) en tiempo $O(m\log n)$.

Problema 2 Colaboración autorizada (16 puntos) **Elementos de Hopcroft-Karp**.

a) **(4 puntos)** Demuestre que si M y M' son matchings de G con $|M'| = |M| + k$ entonces existen al menos k caminos M-aumentantes vértice-disjuntos en G. Concluya que alguno de ellos tiene largo (número de aristas) menor o igual que n/k.

b) **(4 puntos)** Sean M un matching de G, P un camino M-aumentante de largo (número de aristas) mínimo y $M' = M\Delta P$. Demuestre que para todo P' camino M'-aumentante,

$$|P'| \geq |P| + |P' \cap P|$$

Indicación: Estudie $M'\Delta P'$ y $|P'\Delta P|$.

c) **(4 puntos)** Sea M un matching y suponga que el camino M-aumentante más corto tiene largo ℓ. Sean $P_1, \ldots, P_k$ una colección maximal (es decir no se puede agregar ningún otro elemento a la lista) de caminos M-aumentantes de largo ℓ y vértice-disjuntos. Llamamos a cualquier colección de este tipo un conjunto bloqueador respecto a M. Sea $M' = M\Delta P_1 \Delta P_2 \Delta \ldots \Delta P_k$. Demuestre que todo P' camino M'-aumentante tiene largo al menos $\ell + 1$. **Indicación:** Pruebe primero que $|P'| \geq \ell$, ¿Qué pasaría si $|P'| = |P_i|$?

d) **(4 puntos)** No es difícil modificar BFS y DFS en $D(G, M)$ para encontrar un conjunto bloqueador respecto a M en tiempo $O(n + m)$. Suponga que tiene acceso a tal rutina y considere el siguiente algoritmo.

Algoritmo 2: Hopcroft-Karp 1973

Entrada: Grafo G bipartito
$M \leftarrow \emptyset$
Repetir
  Encontrar un conjunto bloqueador respecto a M, $(P_1, P_2, \ldots P_k)$
  si $k = 0$ **entonces devolver** M
  en otro caso $M \leftarrow M\Delta P_1 \Delta P_2 \Delta \cdots \Delta P_k$.

El algoritmo devuelve un matching M de tamaño máximo pues la única forma que $k = 0$ es que no exista camino M-aumentante. Más aún su complejidad es igual a $O(t(n + m))$ donde t es el número de iteraciones del comando repetir interno.

Demuestre que el número de iteraciones del algoritmo es $t = O(\sqrt{m})$ con lo que Hopcroft-Karp tiene complejidad $O(\sqrt{n}(n + m))$

Indicación ¿Cuál es el largo de cualquier camino M-aumentante después de t iteraciones? Luego de calcular esto, use la parte (a) para deducir una cota para $|\text{OPT}| - |M|$ donde OPT es un matching de largo máximo.

Problema 3. **Aplicaciones del Teorema de Hall** (16 puntos)

a) **(8 puntos)** Un **rectángulo latino** es una matriz de r filas y n columnas, con $r \leq n$ en cuyos casilleros se escriben símbolos en $[n]$ de tal forma que ningún símbolo aparezca más de una vez por fila o columna. Si $n = r$ lo llamamos cuadrado latino. Un ejemplo de rectángulo latino es :

1	3	2	5	4	6
2	5	6	1	3	4
4	2	1	3	6	5

Demuestre que todo rectángulo latino se puede completar a un cuadrado latino agregando $n-r$ filas y describa brevemente un algoritmo basado en matchings para hacerlo (no necesita demostrar correctitud o complejidad).

b) **(8 puntos)** Sea $G = (V, E)$ un grafo cualquiera y $p \colon V \to \mathbb{N}$. Una orientación de G que respete p es un digrafo $(V, \vec{E})$ donde cada arista se dirige hacia uno de sus extremos de tal forma que cada vértice tiene a lo más $p(v)$ arcos entrando.

Demuestre que existe una orientación de G que respeta p si y solo si para todo $X \subseteq V$, $\sum_{v \in X} p(v) \geq |E[X]|$, y en ese caso, describa brevemente un algoritmo basado en matchings para encontrar tal orientación (no necesita demostrar correctitud o complejidad)

Problema 4. **Matroides de cubrimiento por matchings (MCM)**[1] (20 puntos)

Sea $G = (V, E)$ un grafo cualquiera no necesariamente bipartito. Decimos que un conjunto $X \subseteq V$ es matchable si existe un matching M_X que cubre a X. Es decir, todo vértice $v \in X$ es incidente a alguna arista $e \in M_X$. La familia $\mathcal{I} \subseteq 2^V$ de los conjuntos matchables es tal que $(V, \mathcal{I})$ es un sistema de independencia. En efecto, el matching sin aristas cubre al conjunto vacío y si $X \subseteq Y$, y M_Y cubre a Y entonces M_Y también cubre a X.

a) **(5 puntos)** Sean $X \in \mathcal{I}$, M_X un matching que cubre a X y $v \in V \setminus X$. Demuestre que $X + v \in \mathcal{I}$ si y solo si o bien M_X cubre a v, o bien (M_X no cubre a v y además existe un camino M_X-alternante que parte en v y termina en un vértice fuera de $X + v$).

b) **(5 puntos)**Demuestre que $\mathcal{N}_G = (V, \mathcal{I})$ es una matroide.

Indicación: Si $|X| < |Y|$ con ambos en $\mathcal{I}$ no se tiene necesariamente que $|M_X| < |M_Y|$. Use la parte anterior.

La matroide $\mathcal{N}_G$ así definida se conoce como la matroide de cubrimiento por matchings de G (MCM).

Sea $G = (L \cup R, E)$ un grafo bipartito y $w \colon L \cup R \to \mathbb{R}_+$ una función de peso en los vértices del grafo. Definimos el peso de una arista e como la suma de los pesos de sus extremos y el peso de un matching como la suma de los pesos de las aristas que lo conforman (llamamos a estas funciones de peso inducidas por vértices).

Usaremos el hecho que $\mathcal{N}_G$ es una matroide para encontrar un matching de peso máximo de (G, w) más eficientemente que el algoritmo húngaro primal-dual.

c) **(5 puntos)** Demuestre que si $X \subseteq L \cup R$ es una base de peso máximo en $\mathcal{N}$ entonces todo matching M_X que cubra a X es matching de peso máximo (más aún, tal M_X debe cubrir solo a X y a ningún vértice adicional).

d) **(5 puntos)** Diseñe un algoritmo basado en el algoritmo glotón de matroides para encontrar un matching de peso máximo de (G, w). Su algoritmo no puede usar oráculos (es decir, debe implementar todos los testeos de independencia). Su algoritmo debe correr en tiempo $O(n(n + m))$.

Indicación: Mantenga siempre en memoria un matching M_X que cubra a su conjunto X independiente actual. Para determinar si $X + v$ es independiente use la parte a), y trabaje en un digrafo similar a $D(G, M_X)$.

[1]En inglés estas matroides se conocen como *matching matroids*, pero este nombre es confuso pues existe otro concepto llamado *matroid matching* que es totalmente diferente