



# DEPARTAMENTO DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

## Algoritmos y Estructuras de Datos III

Primer cuatrimestre 2021 (*dictado a distancia*)

Planaridad y coloreo de grafos

# Grafos planares

Volvamos al problema de Möbius y el rey de la tercera clase:

- ▶ Había una vez un rey con cinco hijos. En su testamento escribió que, después de su muerte, los hijos deberían dividir el reino en cinco regiones de forma tal que cada una limitara con las otras cuatro. Y preguntó si esto era posible.
- ▶ Supongamos que esto fuera posible y construyamos un grafo  $G$  poniendo como vértices cada una de las regiones y haciendo dos vértices adyacentes si las regiones son limítrofes.
- ▶ En este caso, habremos construido  $K_5$ .
- ▶ Si podemos dibujar este grafo en un papel de forma tal que no se crucen las aristas, el testamento del rey se podría cumplir.
- ▶ En caso contrario, no se podría.

# Grafos planares

- ▶ Una *representación planar* de un grafo  $G$  es un conjunto de puntos en el plano, que se corresponden con los vértices de  $G$ , unidos por curvas, que se corresponden con las aristas de  $G$ , sin que estas curvas se crucen entre sí.
- ▶ Un grafo es *planar* si admite una representación planar.
- ▶ Dada una representación planar de un grafo  $G$ , una *región* es el conjunto de todos los puntos alcanzables desde un punto (que no sea un vértice ni parte de una arista) sin atravesar vértices ni aristas.
- ▶ Toda representación planar de un grafo tiene exactamente una región de área infinita, la *región exterior*.

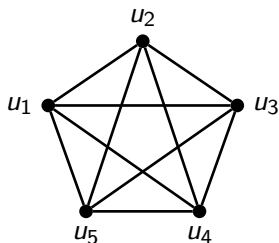
## Grafos planares

- ▶ La *frontera* de una región es el circuito que rodea a la región (puede tener vértices y aristas repetidos).
- ▶ El *grado o tamaño* de una región es el número de aristas que tiene su frontera.
- ▶ El problema de planaridad consiste en, dado un grafo  $G$ , decidir si  $G$  es planar.
- ▶ Este problema está bien resuelto computacionalmente, es decir, existen algoritmos polinomiales para resolverlos.
- ▶ Este concepto es utilizado en ingeniería electrónica, mecánica y civil: en el diseño de circuitos impresos, de rutas y autopistas, en sistemas de irrigación mediante canales que no pueden cruzarse, en localización de instalaciones en mapas.

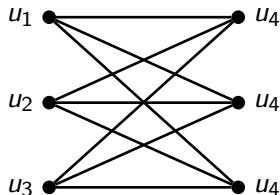
# Grafos planares

**Proposición:**  $K_5$  y  $K_{3,3}$  son grafos no planares.  $K_5$  es el grafo no planar con el menor número de vértices y  $K_{3,3}$  es el que tiene el menor número de aristas.

$K_5$



$K_{3,3}$



**Propiedad:** Si un grafo contiene un subgrafo no-planar es no-planar.

## Grafos planares - Ecuación poliedral de Euler, 1752

**Teorema:** Si  $G$  es un grafo conexo planar entonces cualquier representación planar de  $G$  determina  $r = m - n + 2$  regiones en el plano.

**Corolario:** Si  $G$  es conexo y planar con  $n \geq 3$ , entonces  $m \leq 3n - 6$ .

**Corolario:**  $K_5$  es no planar.

**Corolario:** Si  $G$  es conexo, bipartito y planar con  $n \geq 3$ , entonces  $m \leq 2n - 4$ .

**Corolario:**  $K_{3,3}$  es no planar.

## Coloreo de vértices

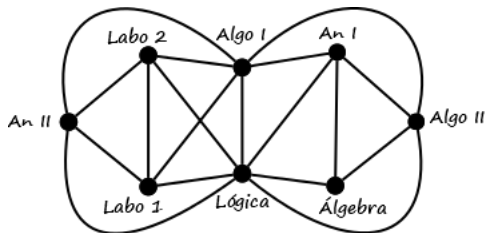
Analicemos el siguiente problema: Supongamos que se tienen cuatro aulas y las siguientes materias con sus respectivos horarios para un mismo día:

Algebra	8 a 12 hs.
Análisis I	10 a 14 hs.
Análisis II	14 a 18 hs.
Lógica	11 a 15 hs.
Algoritmos I	12 a 16 hs.
Algoritmos II	9 a 13 hs.
Laboratorio 1	14 a 18 hs.
Laboratorio 2	14 a 18 hs.

¿Es posible asignar aulas de forma que se puedan dictar todas las materias respetando los horarios?

# Coloreo de vértices

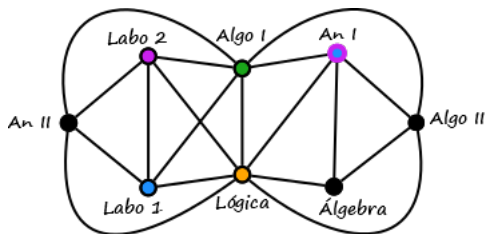
- Podemos modelar el problema mediante un grafo  $G = (V, X)$ .
- $V$  representa a las materias.
- Dos vértices adyacentes si las materias correspondientes se solapan en sus horarios.
- Este es el grafo que obtenemos:





## Coloreo de vértices

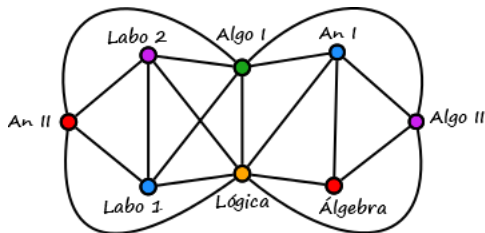
- ▶ Supongamos que tenemos las aulas verde, naranja, violeta y celeste.
- ▶ Le asignamos una, por ejemplo la naranja, a Lógica.
- ▶ Algoritmos I no puede usar el aula naranja porque de 12 a 15 hs. se cursan las dos materias. Le asignamos la verde a Algo I.
- ▶ Laboratorio 1 no puede utilizar ninguna de esas dos aulas, porque se solapa con ambas materias, entonces le asignamos el aula celeste.
- ▶ A Análisis 1 no le podemos asignar ni el aula naranja ni la verde, pero sí podemos la celeste o la violeta.
- ▶ Para Laboratorio 2, la única alternativa es la violeta, porque se solapa con Labo 1, Algo 1 y Lógica.



# Coloreo de vértices

Cuando queremos asignarle un aula a Análisis II, nos encontramos con que:

- ▶ No le podemos asignar la celeste (porque se solapa con Labo1).
- ▶ Ni la verde (porque se solapa con Algo I).
- ▶ Ni la violeta (porque se solapa con Labo2).
- ▶ Ni la naranja (porque se solapa con Lógica)!
- ▶ Ops! Nos encontramos en problemas.
- ▶ Pero podemos ver que si agregamos un aula, la roja, si podemos dictar todas las materias:



## Coloreo de vértices

- ▶ Un *coloreo de los vértices* de un grafo  $G = (V, X)$  es una asignación  $f : V \rightarrow C$ , tal que  $f(v) \neq f(u) \forall (u, v) \in X$ .
- ▶ Los elementos de  $C$  son llamados *colores*. Muchas veces los colores son enteros positivos.
- ▶ Para todo entero positivo  $k$ , un *k-coloreo* de  $G$  es un coloreo de los vértices de  $G$  que usa exactamente  $k$  colores.
- ▶ Un grafo  $G$  se dice *k-coloreable* si existe un  $k$ -coloreo de  $G$ .
- ▶ El *número cromático* de  $G$ ,  $\chi(G)$ , es el menor número de colores necesarios para colorear los vértices de  $G$ .
- ▶ Un grafo  $G$  se dice *k-cromático* si  $\chi(G) = k$ .

## Coloreo de vértices - Ejemplos

- ▶  $\chi(K_n) = n$ .
- ▶ Si  $G$  es un grafo bipartito con  $m > 0$ , entonces  $\chi(G) = 2$ .
- ▶ Si  $H_{2k}$  es un circuito simple par, entonces  $\chi(H_{2k}) = 2$ .
- ▶ Si  $H_{2k+1}$  es un circuito simple impar, entonces  $\chi(H_{2k+1}) = 3$ .
- ▶ Si  $T$  es un árbol con  $n > 1$ , entonces  $\chi(T) = 2$ .

## Cotas para $\chi$

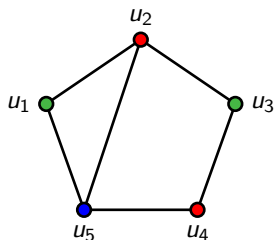
- ▶ Hay numerosas aplicaciones que se modelan mediante coloreo de grafos o alguna de sus variantes, como asignación de frecuencias, asignación de registros, asignación de recursos, planificación horaria.
- ▶ Para grafos en general, no se conocen algoritmos polinomiales para encontrar su número cromático.
- ▶ Comencemos estudiando algunas de cotas de  $\chi$ .

## Cotas para $\chi$

**Proposición:** Si  $H$  es un subgrafo de  $G$  entonces  $\chi(H) \leq \chi(G)$ .

**Definición:** Una clique en un grafo es un subgrafo completo maximal. El número clique  $\omega(G)$  de un grafo es el número de vértices de una clique máxima de  $G$ .

**Proposición:** Para cualquier grafo  $G$ ,  $\chi(G) \geq \omega(G)$ .



$\chi(G) \leq 3$  (porque exhibimos un 3-coloreo)

$\omega(G) \geq 3$  (porque  $\{u_1, u_2, u_5\}$  es clique)

$$\chi(G) = \omega(G) = 3$$

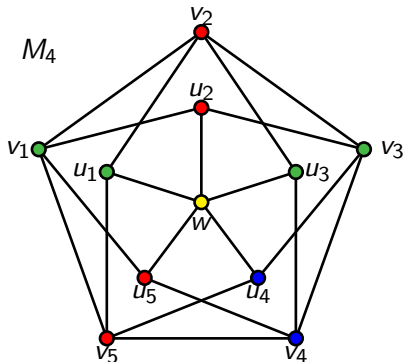
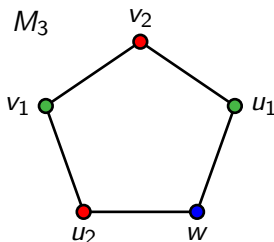
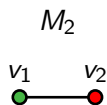
¿Es buena esta cota?

# Grafos de Mycielski

## Definición (por inducción):

1.  $M_1 = K_1$
2.  $M_2 = K_2$
3. Para  $i \geq 2$ ,  $M_{i+1}$  se construye a partir de  $M_i$  de la siguiente forma:
  - ▶ Si  $M_i$  tiene  $p$  vértices,  $v_1, \dots, v_p$ ,  $M_{i+1}$  tendrá  $2p + 1$  vértices,  $v_1, \dots, v_p, u_1, \dots, u_p, w$ , donde  $u_i$  es copia de  $v_i$ .
  - ▶ El conjunto de aristas de  $M_{i+1}$  tendrá todas las aristas de  $M_i$ , las aristas uniendo  $u_i$  con los vecinos de  $v_i$  en  $M_i$  y las aristas uniendo  $w$  con cada  $u_i$ .

# Grafos de Mycielski



¿Cuál es el número cromático de  $M_i$ ?

$$\chi(M_i) = i$$

¿Cuál es la clique máxima de  $M_i$ ?

$$\omega(M_i) = 2$$



## Cotas para $\chi$

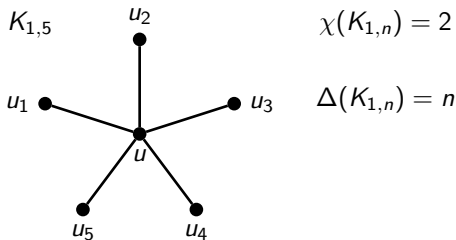
**Proposición:** Si  $\Delta(G)$  es el grado máximo de  $G$  entonces

$$\chi(G) \leq \Delta(G) + 1.$$

**Teorema (Brooks):** Sea  $G$  un grafo conexo que no es un circuito impar ni un grafo completo. Entonces

$$\chi(G) \leq \Delta(G).$$

¿Son buenas estas cotas?



# Problema de los cuatro colores

**Teorema de los 4 colores (Appel, Haken, 1976):** Si  $G$  es un grafo planar, entonces

$$\chi(G) \leq 4.$$

**Teorema (Heawood, 1890):** Si  $G$  es un grafo planar, entonces

$$\chi(G) \leq 5.$$

# Algoritmos para coloreo de grafos

- ▶ Problema *difícil*, computacionalmente no resuelto.
- ▶ No se conocen algoritmos polinomiales para calcular  $\chi(G)$  dado un grafo general  $G$ .
- ▶ Existen muchos enfoques algorítmicos para este problema:
  - ▶ Heurísticas y metaheurísticas.
  - ▶ Algoritmos basados en backtracking (por ejemplo: DSATUR, Brelaz, 1979).
  - ▶ Algoritmos exactos basados en programación lineal entera.

## Algoritmo (heurística) secuencial (S)

*secuencialColoreo*( $G$ )

**entrada:**  $G = (V, X)$  con un orden en los vertices  $v_1, \dots, v_n$

**salida:**  $f$  coloreo de los vertices de  $G$

**para**  $i = 1$  **hasta**  $n$  **hacer**

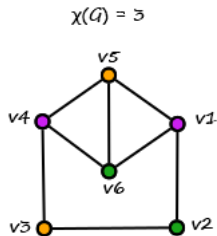
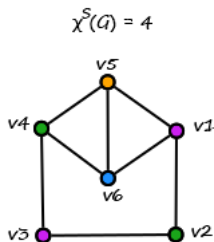
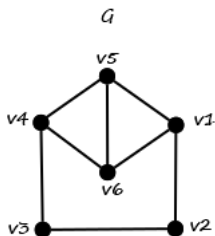
$f[v_i] \leftarrow \min\{h \in \mathbb{N} : f[v_j] \neq h \ \forall (v_j, v_i) \in X, 1 \leq j \leq i - 1\}$

**fin para**

**retornar** coloreo definido por  $f$

## Algoritmo (heurística) secuencial (S)

En el siguiente ejemplo vemos que mediante este algoritmo no obtenemos un coloreo óptimo. Llamamos  $\chi^S(G)$  al valor retornado por la heurística secuencial:



# Algoritmo secuencial (S)

Definimos

$$u_S(G, v_1, v_2, \dots, v_n) = \max_{1 \leq i \leq n} \min\{i, d(v_i) + 1\}.$$

**Proposición:** Si  $\chi_S(G)$  es el número de colores usado por el algoritmo secuencial para colorear  $G$  cuando los vértices son considerados en el orden  $v_1, \dots, v_n$ , entonces

$$\chi(G) \leq \chi^S(G) \leq u_S(G, v_1, v_2, \dots, v_n).$$

# Algoritmo secuencial (S)

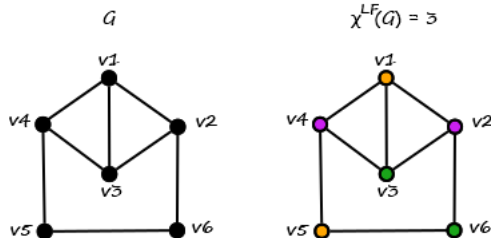
¿Importa el orden en que se colorean los vértices con el algoritmo secuencial?

- ▶ Esta cota induce a pensar que un orden donde los vértices de mayor grado se colorean primero podría dar un mejor resultado de la heurística secuencial.
- ▶ Esto se base en que, como  $f[v_i] \leq \min\{i, d(v_i) + 1\}$ , para los vértices que tienen el segundo de estos valor grande se trataría que el primero sea chico.

# Algoritmo secuencial (LFS)

*Orden Largest First (LF)*: En este orden,  $u_1, \dots, u_n$ , los vértices son ordenados de mayor a menor grado,  $d(u_1) \geq d(u_2) \geq \dots \geq d(u_n)$ .

Llamamos  $\chi^{LF}(G)$  al valor retornado por la heurística secuencial con orden LF:





## Algoritmo secuencial (LFS)

**Proposición:** Si  $u_{LF}(G) = u_S(G, u_1, u_2, \dots, u_n)$  donde  $u_1, u_2, \dots, u_n$  están ordenados según LF. Entonces

$$u_{LF}(G) \leq \text{mín } u_S(G, v_1, v_2, \dots, v_n)$$

donde el mínimo está tomado sobre todos los ordenes posibles,  $v_1, \dots, v_n$ .

¿Esto implica que siempre el algoritmo secuencial da un resultado mejor si se usa LF?

# Algoritmo secuencial

Otra cota (mejor) para el número de colores usados por el algoritmo secuencial aplicado con orden  $v_1, \dots, v_n$  es:

$$\chi^S(G) \leq 1 + \max_{1 \leq i \leq n} \{d_{G_i}(v_i)\}$$

donde  $d_{G_i}(v_i)$  es el grado del vértice  $v_i$  en el grafo inducido por  $v_1, v_2, \dots, v_i$ .

Esta cota sugiere utilizar un orden donde  $d_{G_i}(v_i)$  sea lo más chico posible en cada paso.

# Algoritmo secuencial (SLS)

## Orden *Smallest Last* (SL):

1. fijar como  $v_n$  al vértice de mínimo grado de  $G$ .
2. para  $i = n - 1, \dots, 1$  definir como  $v_i$  el vértice de grado mínimo en el subgrafo de  $G$  inducido por  $V \setminus \{v_n, v_{n-1}, \dots, v_{i+1}\}$ .

Definimos

$$u_{SL}(G) = 1 + \max_{1 \leq i \leq n} \min_{1 \leq j \leq i} \{d_{G_i}(v_j)\}$$

donde  $d_{G_i}(v_j)$  es el grado del vértice  $v_j$  en el grafo inducido por  $V \setminus \{v_n, v_{n-1}, \dots, v_{i+1}\}$ .

## Algoritmo secuencial - Cotas

Se puede demostrar (ejercicio) que:

- ▶  $\chi_{SL}(G) \leq u_{SL}(G)$ , donde  $\chi_{SL}(G)$  es el máximo color usado por la heurística secuencial con orden SL.
- ▶  $u_{SL}(G) \leq u_{LF}(G)$ .
- ▶ La heurística secuencial con orden SL colorea un grafo planar con 6 colores o menos.

# Algoritmo secuencial con intercambio (SI)

- ▶ Es una mejora del algoritmo secuencial.
- ▶ Cuando es necesario utilizar un color nuevo, se trata de recolorear los vértices ya coloreados para no necesitar ese nuevo color.
- ▶ Supongamos que estamos coloreando el vértice  $v$  y necesitamos un color nuevo, el  $k + 1$ . Entonces  $v$  tiene vecinos coloreados con todos los colores  $1, \dots, k$ . El algoritmo intenta modificar el color de algunos vecinos de  $v_i$  para liberar un color de los  $1, \dots, k$  y no necesitar el color nuevo.
- ▶ Si existen  $p$  y  $q$  dos colores utilizados en el coloreo parcial, tal que en todas las componentes conexas de  $H_{pq}$ , el subgrafo inducido por los vertices de colores  $p$  y  $q$ , los vértices adyacentes a  $v$  tienen el mismo color, podemos intercambiar los colores  $p$  y  $q$  en las componentes de  $H_{pq}$  con vértices adyacentes a  $v$  con color  $p$ .
- ▶ De esta manera, obtendremos un coloreo parcial de  $G$  con el color  $p$  no utilizado en la vecindad de  $v$ .

## Algoritmo secuencial con intercambio (SI)

*secuencialConIntercambio*( $G$ )

$k \leftarrow 0$

**para**  $i = 1$  **hasta**  $n$  **hacer**

$g \leftarrow \min\{h \in \mathbb{N} : f[v_j] \neq h \ \forall (v_j, v_i) \in X, 1 \leq j \leq i-1\}$

**si**  $g \leq k$  **hacer**

$f[v_i] \leftarrow g$

**sino**

**si** existen  $1 \leq p < q \leq k$ , tales que

un  $p, q$ -intercambio libera  $p$  de  $N(v_i)$  **entonces**

realizar el  $p, q$ -intercambio

$f[v_i] \leftarrow p$

**sino**

$f[v_i] \leftarrow g$

$k \leftarrow k + 1$

**fin si**

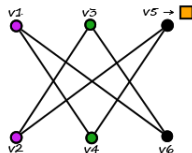
**fin si**

**fin para**

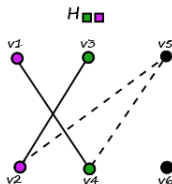
**retornar** coloreo definido por  $f$



# Algoritmo secuencial con intercambio (SI)

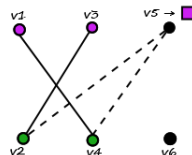
Cuando el algoritmo quiere pintar el vértice  $v_5$  evita utilizar un nuevo color realizando un intercambio.






Cuando el algoritmo va a pintar a  $v_5$  necesita un nuevo color



En todas las componentes de  $H$    los vértices adyacentes a  $v_5$  tienen el mismo color



Intercambia los colores  y  en todas las componentes con vecinos de color 

## Algoritmo secuencial con intercambio (SI)

- ▶ Requiere un mayor esfuerzo computacional.
- ▶ No siempre es mejor el algoritmo SI que el algoritmo S.
- ▶ Se puede demostrar que:
  - ▶ SI colorea un grafo bipartito con 2 colores (ejercicio).
  - ▶ SI con el ordenamiento SL colorea un grafo planar con 5 colores como máximo.



# Algoritmo secuencial con bracktracking (exacto)

- ▶ En cada paso se extiende una solución parcial coloreando un vértice aún no coloreado.
- ▶ Una solución puede estar representada por  $(a_1, \dots, a_n)$ , con  $a_i$  indicando el color asignado al vértice  $v_i$ . Una solución parcial estará representada por  $(a_1, \dots, a_k)$ ,  $k \leq n$ .
- ▶ Dada una solución parcial  $(a_1, \dots, a_k)$ ,  $k < n$  que representa un coloreo parcial válido, construiremos el conjunto de sus vértices hijos,  $\text{Sucesores}(a, k)$ , asegurando que siga siendo un coloreo válido.
- ▶ Entonces el color asignado al vértice  $v_k$  no puede ser un color ya utilizado para pintar un vértice vecino de  $v_k$ .
- ▶ Además, para evitar generar soluciones que no serán óptimas (usan más colores que el mejor coloreo encontrado hasta el momento) y soluciones simétricas (se considera una solución que usará la misma cantidad de colores) se aplican procesos de poda.

## Algoritmo secuencial con bracktracking (exacto)

$$\text{Sucesores}(a, k) = \{(a, a_k) : a_k \in U\}$$

con  $j \in U$  si:

- ▶  $j$  no es color asignado a un vecino de  $v_k$  ya coloreado (poda por factibilidad)
- ▶  $j \leq d(v_k) + 1$  (poda por simetría)
- ▶  $1 \leq j \leq l + 1$  (poda por simetría)
- ▶ si ya se encontró un coloreo del grafo con  $q$  colores entonces  $j \leq q - 1$  (poda por optimalidad),

donde  $l$  es el máximo color usado en la solución parcial  $a$ ,

$$l = \max_{1 \leq i \leq k-1} a_i.$$

## Algoritmo secuencial con bracktracking (exacto)

- ▶ En el árbol de búsqueda se abre una rama a partir de cada vértice (correspondiente a un coloreo de  $v_1, \dots, v_{k-1}$ ), para cada elemento de  $U$ .
- ▶ Se avanza por las ramas coloreando los siguientes vértices hasta que ocurre alguna de las siguientes situaciones:
  - ▶ se llegó a un vértice con  $U = \emptyset$ : a partir de esta situación se hace *backtracking* a partir de  $v_{k-1}$ .
  - ▶ se coloreó  $v_n$ : se encontró un nuevo coloreo del grafo, hay que actualizar  $q$  y hacer *backtracking*.
- ▶ En base a estas ideas, podemos implementar un algoritmo iterativo de backtracking.

# Algoritmo secuencial con bracktracking con poda (exacto)

Utilizaremos estas variables:

- ▶  $q$ : cantidad de colores usados en la mejor solución encontrada hasta el momento.
- ▶  $f^*$ : mejor solución encontrada hasta el momento.
- ▶  $k$ : vértice siendo considerado.
- ▶  $f$ : solución parcial.
- ▶  $l$ : cantidad de colores utilizados en la solución parcial actual.
- ▶  $l_k$ :  $l$  para el vértice  $v_k$ .
- ▶  $cotaInf$ : cota inferior para el número cromático del grafo.

# Algoritmo secuencial con bracktracking con poda (exacto)

- ▶ Como los colores son indistinguibles, sin pérdida de generalidad fijamos en 1 el color de  $v_1$ ,  $f[v_1] = 1$ .
- ▶ Cuando se llega a colorear el vértice  $v_n$  se obtiene un coloreo completo:
  - ▶ Se actualiza  $f^*$  y  $q$  al mayor color utilizado  $l$ .
  - ▶ Se borran los colores mayores o iguales a  $l$  de los conjuntos  $U'$ s.
  - ▶ Se poda la rama desde el vértice que utiliza el color  $l$  por primera vez ( $k \leftarrow j - 1$ ), porque ya no nos interesa encontrar otra solución con  $l$  o más colores.

# Algoritmo secuencial con bracktracking (exacto)

*bktColoreo*( $G$ )

$q \leftarrow n + 1$ ,  $f[v_1] = 1$ ,  $k \leftarrow 1$ ,  $l \leftarrow 1$

*avanzar*  $\leftarrow$  VERDADERO

**repetir**

**si** *avanzar* **entonces**  $k \leftarrow k + 1$ ,  $l_k \leftarrow l$ , determinar  $U_k$

**si**  $U_k = \emptyset$  **entonces**

*avanzar*  $\leftarrow$  FALSO,  $k \leftarrow k - 1$ ,  $l \leftarrow l_k$

**sino**

$j \leftarrow \min U_k$ ,  $U_k \leftarrow U_k \setminus \{j\}$ ,  $f(v_k) \leftarrow j$

**si**  $j > l$  **entonces**  $l \leftarrow l + 1$

**si**  $k < n$  **entonces**

*avanzar*  $\leftarrow$  VERDADERO

**sino**

$f^* \leftarrow f$

$j \leftarrow \min\{i \text{ tal que } f[v_i] = l\}$

borrar  $l, l + 1, \dots, q - 1$  de  $U_1, \dots, U_{j-1}$

$q \leftarrow l$ ,  $l \leftarrow q - 1$ ,  $k \leftarrow j - 1$

*avanzar*  $\leftarrow$  FALSO

**hasta**  $k = 1$  **o**  $q = \text{cotaInf}$

**retornar**  $f^*$  y  $q$

# Coloreo de aristas

- ▶ Un *coloreo válido de las aristas* de un grafo  $G$  es un asignación de colores a las mismas en la cual dos aristas que tienen un vértice en común no tengan el mismo color.
- ▶ El *índice cromático*  $\chi'(G)$  de un grafo  $G$  es el menor número de colores con que se pueden colorear las aristas de un grafo.

**Teorema de Vizing:** Para todo grafo  $G$  se verifica que

$$\Delta(G) \leq \chi'(G) \leq \Delta(G) + 1.$$