

# Algoritmo de coloreo de grafos

Pedro O. Pérez M., PhD.

Análisis y diseño de algoritmos avanzados  
Tecnológico de Monterrey

*pperezm@tec.mx*

12-2022

## ① Problema de Coloreo de grafos

### Grafo Bipartita

Definición

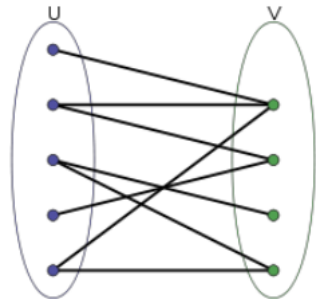
Algoritmo

### Problema general

Definición

Algoritmo de Welsh & Powell

Un grafo bipartita (o bigrafo)  $G = (V, E)$  es un grafo cuyos vértices pueden ser divididos en dos conjuntos disjuntos  $R$  y  $S$  tal que cada arco conecta a un vértice en  $R$  con un vértice en  $S$ .



---

## Procedure 1 BIGRAPH

---

**Input:**  $G$  : Graph

$Q$  : Queue,  $Color$  : Array,  $isBipartite$  : boolean

$INIT(Color, -1)$

$isBipartite \leftarrow true$

$vertex \leftarrow$  some vertex in  $G$

$Color[vertex] \leftarrow 1$

$Q.enqueue(vertex)$

**while**  $Q$  is not empty **do**

    NEXT SLIDE

**end while**

**return**  $isBipartite$

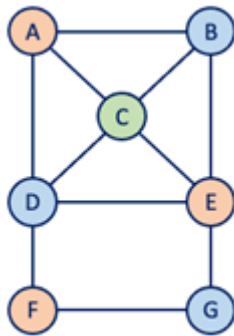
---

---

```
u ← Q.dequeue()
for each (u, v) incident in u do
  if Color[v] = −1 then
    Color[v] ← 1 − Color[u]
    Q.enqueue(v)
  else
    if Color[v] = Color[u] then
      isBipartite ← false
    end if
  end if
end for
```

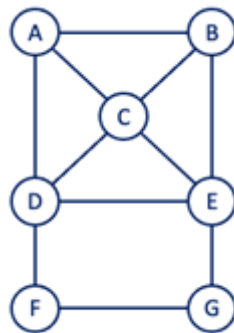
---

El coloreo de grafos es un problema en donde se va coloreando (etiquetando) a cada vértice de un grafo no dirigido, con la restricción de los vértices adyacentes no tengan el mismo color o etiqueta. Esto es que cada vértice tenga un color diferente a los de sus vecinos. Esta idea puede servir para solucionar problemas de selección para grupos, equipos, etc. donde los vértices son las personas u objetos y las restricciones se vean como arcos que los une.



El algoritmo de coloreo de grafos propuesta por Welsh & Powell en 1967, es un algoritmo ávido que consiste en obtener el grado de cada vértice (cantidad de arcos que llegan a él) y ordenar en forma descendente. Se empieza colorear el de mayor grado, y se va verificando en forma ordenada si se puede colorear con el mismo color a algún otro vértice. Una vez terminando esta iteración, se continua con el siguiente vértice de mayor grado, utilizando la misma estrategia, hasta llegar al último.

- ① Grado 4: C, D, E
- ② Grado 3: A, B
- ③ Grado 2: F, G





---

## Procedure 2 WELSH\_POWERELL

---

**Input:**  $G$  : Graph

$Color$  : Array,  $colorAssigned$  : Integer

Sort vertices in  $G$  descending by degree,  $sortedV$

$colorAssigned \leftarrow 0$

**for** each  $v$  in  $sortedV$  **do**

**if** has  $v$  NOT been colored? **then**

$colorAssigned \leftarrow colorAssigned + 1$

$Color[v] \leftarrow colorAssigned$

**for** each remaining vertex,  $u$ , in  $sortedV$  **do**

**if**  $u$  has not been colored and is not adjacent to a node with  $assignedColor$  **then**

$Color[u] \leftarrow colorAssigned$

**end if**

**end for**

**end if**

**end for**

**return**  $Color$

---