

# UNIVERSIDAD AUTÓNOMA DE MÉXICO

## Facultad de Ciencias

Autores:

Fernanda Villafán Flores

Fernando Alvarado Palacios

Adrián Aguilera Moreno



Gráficas y Juegos

## Tarea 7

1. (a) Demuestre que si  $G$  tiene diámetro mayor que 3 (posiblemente infinito), entonces  $\overline{G}$  tiene diámetro menor que 3. Concluya que si  $G$  es inconexa, entonces  $\overline{G}$  es conexa.

**Demostración:** Sea  $G$  una gráfica.

Probaremos que  $\overline{G}$  tiene diámetro menor a 3.

Primero, sabemos que  $G$  tiene diámetro mayor a 3 entonces tomemos una trayectoria  $P$  de  $G$  tal que su longitud es  $n$  (con  $n > 3$ ).

La denotaremos como:

$$P = (x_0, x_1, \dots, x_n)$$

Ahora, por definición de  $\overline{G}$  es tal que:

$$|E_{\overline{G}}| = \binom{|V| - 1}{2} - |E_G|$$

Por tanto, en  $\overline{G}$  la trayectoria  $P$  cambia de la siguiente manera:

- El vértice  $x_0$  es adyacente a los vértices  $x_2, x_3, \dots, x_n$ , donde esto equivale a  $n - 1$  vértices.
- El vértice  $x_1$  es adyacente a los vértices  $x_3, x_4, \dots, x_n$ , donde esto equivale a  $n - 2$  vértices.
- El vértice  $x_2$  es adyacente a los vértices  $x_0, x_4, \dots, x_n$ , donde esto equivale a  $n - 2$  vértices.

Siguiendo este procedimiento, tenemos lo siguiente:

- El vértice  $x_i$  es adyacente a los vértices  $x_{i-2}, x_{i+2}, x_{i+3}, \dots, x_n$ , con  $i > 1$ .

Así, notemos lo siguiente:

En  $\overline{G}$   $x_0$  **no es** adyacente a  $x_1$ , entonces necesitamos otro vértice  $x_3$  para llegar a  $x_1$ . Lo que implica que nos toma distancia igual a 2 llegar de  $x_0$  a  $x_1$ .

De la misma forma,  $x_1$  **no es** adyacente a  $x_0$  ni a  $x_2$ , entonces necesitamos otro vértice  $x_4$  para llegar a  $x_0$  o  $x_2$ . Lo que implica que nos toma distancia igual a 2 llegar de  $x_1$  a  $x_0$  o de  $x_1$  a  $x_2$ .

De lo anterior obtenemos que:

El vértice  $x_i$  **no es** adyacente al vértice  $x_{i-1}$  ni al vértice  $x_{i+1}$ , entonces necesitamos otro vértice  $x_{i+2}$  para llegar a  $x_{i-1}$  o  $x_{i+1}$ . Lo que implica que nos toma distancia igual a 2 llegar de  $x_i$  a  $x_{i-1}$  o de  $x_i$  a  $x_{i+1}$ .

Por lo tanto,  $\overline{G}$  tiene diámetro menor a 3.

Aún más, si  $G$  es inconexa entonces  $\overline{G}$  es conexa (ya que por definición, para cualesquiera dos vértices distintos  $u, v \in G$  se tiene que  $uv \in E_{\overline{G}}$  si y sólo si  $uv \notin E_G$ ). Es decir, en  $\overline{G}$  estarán todas las aristas que no estén en  $G$ .  $\square$

- (b) Una gráfica  $G$  es autocomplementaria si  $G \cong \overline{G}$ . Demuestre que si  $G$  es autocomplementaria, entonces  $|V| \stackrel{4}{\equiv} 0$  o  $|V| \stackrel{4}{\equiv} 1$ .

**Demostración:** Primero, sabemos que si  $G \cong \overline{G}$  entonces  $V_G = V_{\overline{G}}$ .

Probaremos que  $|E_G| = |E_{\overline{G}}|$ .

Veamos lo siguiente:

$$|V| \stackrel{4}{\equiv} 1 \longrightarrow |V| \equiv 1 \pmod{4}$$

Recordando la definición de  $\bmod$ , tenemos:

$$\begin{aligned} |V| \equiv 1 \pmod{4} &\longrightarrow 4 \mid |V| - 1 \\ &\longrightarrow |V| - 1 = 4 \cdot k, \text{ con } k \in \mathbb{N} \\ &\longrightarrow |V| = 4 \cdot k + 1 \end{aligned}$$

Luego, por definición de  $\overline{G}$ , tenemos:

$$|E_{\overline{G}}| = \binom{|V| - 1}{2} - |E_G|$$

Sea  $n = |V_G|$ .

Así,

$$\begin{aligned} |E_G| &= |E_{\overline{G}}| \\ &= \binom{|V_G| - 1}{2} - |E_G| \\ &= \binom{n - 1}{2} - (n - 1), \text{ porque sabemos que } |E_G| = |V_G - 1| \text{ y } |V_G| = n \\ &= \frac{(n - 1)!}{2! \cdot ((n - 1) - 2)!} - (n - 1) \\ &= \frac{(n - 1)!}{2! \cdot (n - 1 - 2)!} - (n - 1) \\ &= \frac{(n - 1)!}{2! \cdot (n - 3)!} - (n - 1) \\ &= \frac{(n - 1)(n - 2)(n - 3)!}{2! \cdot (n - 3)!} - (n - 1), \text{ porque } n! = n \cdot (n - 1) \cdot (n - 2)! \\ &= \frac{(n - 1)(n - 2)\cancel{(n - 3)!}}{2! \cdot \cancel{(n - 3)!}} - (n - 1) \\ &= \frac{(n - 1)(n - 2)}{2!} - (n - 1) \\ &= \frac{(n - 1)(n - 2)}{2} - (n - 1), \text{ porque } 2! = 2 \cdot 1 = 2 \\ &= \frac{(n - 1)(n - 2)}{2} - \frac{2(n - 1)}{2} \\ &= \frac{(n - 1)(n - 2) - 2(n - 1)}{2} \\ &= \frac{n^2 - 3n + 2 - 2n + 2}{2} \\ &= \frac{n^2 - 5n + 4}{2} \end{aligned}$$

Ahora,

$$\begin{aligned}
 \frac{n^2 - 5n + 4}{2} &= \frac{4 \left[ \frac{n^2}{4} - \frac{5n}{4} + 1 \right]}{2} \\
 &= 4 \cdot \frac{\left[ \frac{n^2}{4} - \frac{5n}{4} + 1 \right]}{2} \\
 &= 4 \cdot \frac{\left[ \frac{n^2 - 5n}{4} + 1 \right]}{2} \\
 &= 4 \cdot \frac{\left[ \frac{n^2 - 5n + 4}{4} \right]}{2} \\
 &= 4 \cdot \frac{n^2 - 5n + 4}{8}
 \end{aligned}$$

Por lo tanto, tenemos lo siguiente:

$$|E_G| = 4 \cdot \left[ \frac{n^2 - 5n + 4}{8} \right] \longrightarrow |V_G| - 1 = 4 \cdot \left[ \frac{n^2 - 5n + 4}{8} \right]$$

Despejando  $|V_G|$ , obtenemos:

$$|V_G| - 1 = 4 \cdot \left[ \frac{n^2 - 5n + 4}{8} \right] \longrightarrow |V_G| = 4 \cdot \left[ \frac{n^2 - 5n + 4}{8} \right] + 1$$

Sea  $k = \left[ \frac{n^2 - 5n + 4}{8} \right]$ . Entonces:

$$|V_G| = 4 \cdot k + 1$$

Por lo tanto, llegamos a que  $|E_G| = |E_{\bar{G}}|$  si  $|V_G| = 4 \cdot k + 1$ .

□

2. Un *orden topológico* de una digráfica  $D$  es un orden lineal de sus vértices tal que para cada flecha  $a$  de  $D$ , la cola de  $a$  precede a su cabeza en el orden.

- (a) Demuestre que toda digráfica acíclica tiene al menos una fuente (vértice de ingrado 0) y un sumidero (vértice de exgrado 0).

**Demostración:** Procedamos por reducción al absurdo. Sea  $D$  una digráfica acíclica con  $\delta^+ > 0$  y  $\delta^- > 0$ , esto es que, para cada  $v \in V_D$  hay una flecha que le “pega”<sup>1</sup> a  $v$  y otra que “sale”<sup>2</sup> de  $v$ . Tomemos la trayectoria  $\vec{T}$  más larga en  $D$  y sea  $x \in V_D$  el último vértice de  $\vec{T}$ , luego en  $x$  sale una arista hacia algún otro vértice en  $\vec{T}$  [pues si saliera hacia algún otro vértice que no este en  $\vec{T}$ , llegaríamos a que  $\vec{T}$  no es de longitud máxima!!], así  $\vec{T}xy$  claramente contiene un ciclo, esto implica que  $D$  contiene un ciclo!!, he aquí una contradicción de suponer que  $D$  no contiene ciclos.

∴ Si  $D$  es acíclica tiene al menos una fuente y un sumidero.

□

- (b) Deduzca que una digráfica admite un orden topológico si y sólo si es acíclica.

<sup>1</sup>Una arista incide en  $v$  y  $v$  es la cabeza.

<sup>2</sup>Una arista que inicia en  $v$  con dirección a otro vértice.

**Demostración:** Para este inciso analicemos 2 posibles casos:

$\Rightarrow$ ) Procedamos por reducción al absurdo. Sea  $D$  una digráfica tal que admite un orden topológico. Supongamos que  $D$  contiene al menos un ciclo  $C$ , entonces existe un  $x \in V_D$  tal que  $\{x\} \subset C$  y  $x$  es un vértice inicial y final en  $C$ , luego existe  $y \in V_D : \{y\} \subset C$  tal que  $\vec{yx}$  es una arista, por tanto  $y < x$  [esto es que  $y$  precede a  $x$  en el orden]. Nótese que hay una trayectoria  $\vec{T}$  que va de  $x$  a  $y$  en  $C$ , así  $x < y$ !! [esto es que  $x$  precede a  $y$  en el orden], he aquí una contradicción de suponer que  $D$  admite un orden topológico.

$\therefore$  Si  $D$  admite un orden topológico  $\Rightarrow D$  es acíclica.

$\Leftarrow$ ) Por el inciso (a) sabemos que  $D$  tiene al menos una *fuentes* y un *sumidero*, tomemos una componente conexa en  $D$  y veamos que si los vértices  $x$  es *fuentes* e  $y$  es *sumidero*, entonces la trayectoria de  $x$  a  $y$  es un orden topológico, si hay más de una *fuentes* o más de un *sumidero*, cada trayectoria entre una *fuentes* y un *sumidero* es un orden topológico [pues de no serlo, dos flechas distintas provenientes de una misma fuente incidirían en algún vértice en común, lo que implicaría que  $D$  contiene un ciclo!!], así la componente conexa admite un orden topológico y esto pasa para cualquier componente conexa en  $D$ .

$\therefore$  Si  $D$  es acíclica  $\Rightarrow D$  admite un orden topológico.

$\therefore$  Una digráfica admite un orden topológico si y sólo si es acíclica.  $\square$

(c) Exhiba un algoritmo de tiempo a lo más cuadrático para encontrar un orden topológico en una digráfica acíclica.

A continuación se muestra el algoritmo<sup>3</sup> requerido:

---

**1: TopologicalOrder( $D; D$ )**

---

**Input:** Una digráfica  $D$  acíclica.

**Output:** Un orden topológico admitido en  $D$  basado en números.

---

```

1 for  $v \in V_D$  do
2   if  $d^-(v) = 0$  then
3      $v \leftarrow 0$ ;
4   end
5 end
6 for  $v \in V_D$  do
7   if  $v \neq 0$  then
8     temp  $\leftarrow 0$ ;
9     for  $u \in V_D : u$  es antecesor de  $v$  en  $D$  and  $u \neq \text{null}$  do
10      if temp  $< u$  then
11        temp  $\leftarrow u$ 
12      end
13    end
14     $v \leftarrow \text{temp} + 1$ ;
15    for  $u \in V_D : u$  es sucesor de  $v$  en  $D$  and  $u \neq \text{null}$  do
16      if  $u < v$  then
17         $u \leftarrow v + 1$ 
18      end
19    end
20  end
21 end
22 return  $D$ 
```

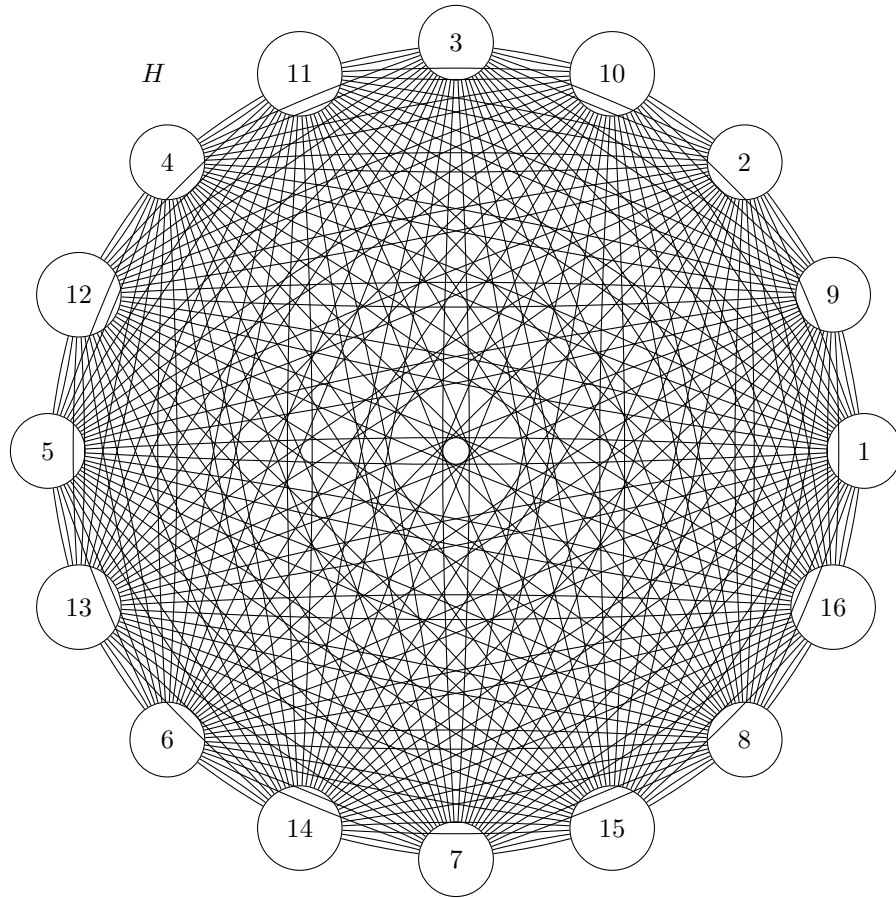
---

<sup>3</sup>Tome en cuenta que suponemos que  $D$  se pasa como parámetro con valores nulos en sus vértices.

3. Demuestre que cada uno de los siguientes problemas está en la clase  $NP$  exhibiendo un certificado y un algoritmo de tiempo polinomial para verificar el certificado (escriba el algoritmo utilizando pseudo código como el visto en clase; sólo está permitido el uso de las estructuras de control **if**, **while** y **for**). Demuestre que su algoritmo usa tiempo polinomial.

- (a) HAMILTON CYCLE.
- (b) VERTEX COVER.
- (c) COLOURING.
- (d) DOMINATING SET.

**Solución de (a):** A continuación se da un certificado para una gráfica que contiene un ciclo hamiltoniano:



y  $S = (v_0, v_1, \dots, v_n, v_0) = (3, 10, 2, 9, 1, 16, 8, 15, 7, 14, 6, 13, 5, 12, 4, 11, 3)$  una colección que contiene los vértices en sucesión tal que esta sucesión forma un ciclo hamiltoniano en  $H$ . Así, nuestro algoritmo es el siguiente:

**2:** HamiltonCycle( $\langle H, S \rangle$ ; true/false)

**Input:** Una gráfica  $H$  y una colección  $S$  que contiene a la sucesión de vértices que representará el ciclo hamiltoniano en  $H$ .

**Output:** TRUE o FALSE dependiendo si  $S$  es un ciclo hamiltoniano en  $H$ .

```

1 if  $|V_G| \neq |S|$  then
2   | return false;
3 end
4 for  $v \in S$  do
5   | siguiente=0;
6   | while siguiente <  $|V_H|$  do
7     |    $u \leftarrow S(\text{siguiente})$ ;
8     |   siguiente  $\leftarrow$  siguiente + 1;
9     |   if  $vu \notin E_H$  then
10    |     | return false;
11    |   end
12  | end
13 end
14 if  $S(0) \neq S(|V_H| - 1)$  then
15   | return false;
16 end
17 for  $v \in S$  do
18   | decision  $\leftarrow$  false;
19   | for  $u \in V_G$  do
20     |   if  $u = v$  then
21     |     | decision  $\leftarrow$  true;
22     |   end
23   | end
24   | if decision = false then
25     |   return decision;
26   | end
27 end
28 return true;
```

Este algoritmo verifica si la entrada (input) es un sí-certificado o no.

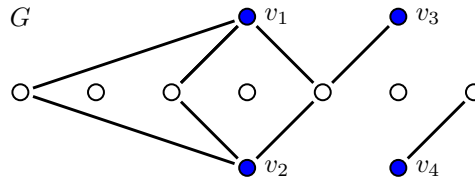
Obs. Tomar a  $S$  como una colección da flexibilidad en su implementación, es por eso que  $S(\text{int siguiente})$  está entre paréntesis.

Ahora, analicemos la complejidad del algoritmo HamiltonCycle, veamos que en la línea 1 tenemos una instrucción iterativa que por definición de  $S$  tenemos que se iterará  $|V_G| - 1$  veces, luego en la línea 6 hay otro ciclo tal que se itera  $|V_G|$  veces, así por la regla de la multiplicación de complejidades tenemos una complejidad, hasta el momento, contenida en  $\mathcal{O}(|V_G|^2)$ . Luego, por la línea 17 sabemos que hay una instrucción iterativa no anidada en las instrucciones anteriores que se iterará  $|V_G|$  veces, y por la línea 19 sabemos que hay un ciclo que se itera  $|V_G|$  veces y además está anidada en el ciclo de la línea 17, luego la complejidad de HamiltonCycle esta contenida en  $\mathcal{O}(|V_G|^2 + |V_G|^2)$  y por la regla de suma de complejidades tenemos que

$$\mathcal{O}(|V_G|^2 + |V_G|^2) = \mathcal{O}(|V_G|^2)$$

Como la complejidad de HamiltonCycle está en un tiempo cuadrático y por la definición de problemas en la clase  $NP$ , tenemos que HamiltonCycle está en la clase  $NP$ .  $\square$

**Solución de (b):** A continuación se da un certificado para una gráfica que contiene un cobertura de vértices:



con  $S = (v_1, v_2, v_3, v_4)$  una cubierta de vértices en  $G$ . Así nuestro algoritmo sería el que a continuación se muestra:

---

**3:** VertexCover( $\langle G, S \rangle$ ; true/false)

---

**Input:** Una gráfica  $G$  y una colección  $S$  que contiene a la sucesión que conforma la cobertura de vértices en  $G$ .

**Output:** TRUE o FALSE dependiendo si  $S$  es una cobertura de vértices en  $G$ .

---

```

1 contador  $\leftarrow$  0;
2 for  $v \in S$  do
3   decision  $\leftarrow$  false;
4   for  $u \in V_G$  do
5     if  $u = v$  then
6       decision  $\leftarrow$  true;
7     end
8   end
9   if decision = false then
10    return decision;
11  end
12  for  $u \in V_G$  do
13    if  $vu \in E_G$  then
14      contador  $\leftarrow$  contador + 1;
15    end
16  end
17 end
18 if contador  $\neq$   $|E_G|$  then
19   return false;
20 end
21 return true;
```

---

Este algoritmo verifica que la entrada (input) sea un sí-certificado, *i.e.*, que  $S$  sea una cobertura de vértices sobre  $G$ . Esto es, que la unión de todas las aristas incidentes en los vértices de  $S$  contengan todas las aristas de  $G$ .

Ahora analicemos la complejidad del algoritmo VertexCover, veamos que en la línea 2 hay una instrucción iterativa de complejidad en  $\mathcal{O}(|V_G|)$ , en la cual están anidadas las instrucciones de las líneas 4 y 12, por regla de la suma de complejidades y como estas instrucciones<sup>4</sup> tienen la misma complejidad (contenida en  $\mathcal{O}(|V_G|)$ ), es indistinto con cual nos quedemos, así por la regla del producto tenemos que la complejidad de VertexCover es  $\mathcal{O}(|V|^2)$ .

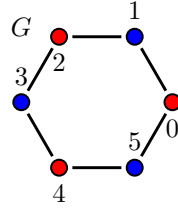
Como la complejidad del algoritmo VertexCover es  $\mathcal{O}(|V|^2)$ , podemos concluir que se ejecuta en tiempo polinomial, luego por la definición de problemas en la clase  $NP$ , tenemos que VertexCover está contenido en esta misma.  $\square$

---

<sup>4</sup>Línea 4 y línea 12.



**Solución de (c):** A continuación se muestra un certificado para una gráfica que admite una coloración:



con  $S = (1R, 3R, 5R, 0A, 2A, 4A)$  una coloración de vértices en  $G$ . Luego, nuestro algoritmo sería el que a continuación se muestra:

---

**4: Colouring**( $\langle G, S \rangle$ ; true/false)

---

**Input:** Una gráfica  $G$  y una colección  $S$  que contiene a la sucesión que conforma la coloración de vértices en  $G$ .

**Output:** TRUE o FALSE dependiendo si  $S$  es una coloración de vértices en  $G$ .

```

1 if  $|S| \neq |V_G|$  then
2   return false;
3 end
4 for  $v \in S$  do
5   decision  $\leftarrow$  false;
6   for  $u \in V_G$  do
7     if  $v = u$  then
8       decision  $\leftarrow$  true;
9     end
10  end
11  if decision = false then
12    return decision;
13  end
14  for  $u \in S$  do
15    if  $v$  tiene el mismo color que  $u$  then
16      if  $uv \in E_G$  then
17        return false;
18      end
19    end
20  end
21 end
22 return true;
```

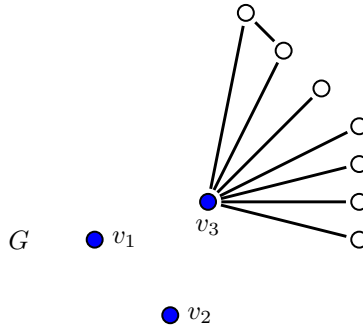
---

Este algoritmo verifica si la entrada (input) es un sí-certificado o no, *i.e.*, si  $S$  es una coloración para  $G$ . Esto es, que a cada vértice de la gráfica le asigne un color de tal manera que dos vértices vecinos no tienen el mismo color.

Analizando la complejidad del algoritmo Colouring tenemos que por la línea 4 hay una instrucción iterativa que tiene complejidad en  $\mathcal{O}(|V|)$ , luego los ciclos de las líneas 6 y 14 están anidados en el de la línea 4, usando la regla de la suma (y como ambos tienen la misma complejidad) tenemos que la complejidad de Colouring es  $\mathcal{O}(|V_G|^2)$ .

Como Colouring tiene complejidad polinomial, se sigue que Colouring esta en la clase  $NP$ .  $\square$

**Solución de (d):** A continuación se muestra un conjunto dominante como certificado para una gráfica  $G$ :



con  $S = (v_1, v_2, v_3)$  un conjunto dominante de vértices en  $G$ . Luego, nuestro algoritmo sería el que a continuación se muestra:

---

**5:** DominatingSet( $\langle G, S \rangle$ ; true/false)

---

**Input:** Una gráfica  $G$  y una colección  $S$  que contiene a la sucesión que conforma un conjunto dominante de vértices en  $G$ .

**Output:** TRUE o FALSE dependiendo si  $S$  es un conjunto dominante de vértices en  $G$ .

---

```

1  $C \leftarrow \emptyset$ ;
2 for  $v \in S$  do
3   decision  $\leftarrow$  false;
4   for  $u \in V_G$  do
5     if  $v = u$  then
6       decision  $\leftarrow$  true;
7     end
8   end
9   if decision = false then
10    return false;
11  end
12   $C \leftarrow v$ ;
13  for  $u \in V_G$  do
14    if  $uv \in E_G$  then
15       $C \leftarrow u$ ;
16    end
17  end
18 end
19 for  $v \in V_G$  do
20   decision  $\leftarrow$  false;
21   for  $u \in C$  do
22     if  $v = u$  then
23       decision  $\leftarrow$  true;
24     end
25   end
26 end
27 return decision;

```

---

Este algoritmo verifica si la entrada (input) es un sí-certificado o no, *i.e.*, si  $S$  es un conjunto dominante para  $G$ . Esto es, que la unión de los vecinos de los vértices de  $S$ , más los vértices que están en  $S$ , son todos los vértices en  $G$ .

Por la línea 2 sabemos que hay una instrucción iterativa que contiene anidadas a los ciclos de las

líneas 4 y 13, en el peor de los casos el ciclo de la línea 2 se iterará  $|V_G|$  veces y esto producto con  $|V_G|$  que son las veces que se iteran los ciclos de las líneas 4 y 13, esto nos da una complejidad en  $\mathcal{O}(|V_G|)$ . Luego los ciclos anidados de las líneas 19 y 21 nos dan una complejidad en  $\mathcal{O}(|V_G|^2)$ , así concluimos que DominatingSet tiene una complejidad en  $\mathcal{O}(|V_G|^2)$ .

Como DominatingSet tiene complejidad cuadrática, entonces tiene complejidad polinomial y por tanto DominatingSet se encuentra en la clase de problemas  $NP$ .  $\square$

### Puntos extra

1. Demuestre que toda digráfica sin lazos admite una descomposición en dos digráficas acíclicas, es decir, que existen  $D_1$  y  $D_2$  subdigráficas de  $D$ , acíclicas y tales que  $D_1 \cup D_2 = D$  y  $A_{D_1} \cap A_{D_2} = \emptyset$ .
2. Un torneo es una digráfica en la que entre cualesquiera dos vértices existe una única flecha. Demuestre que todo torneo es fuertemente conexo o puede transformarse en un torneo fuertemente conexo al reorientar exactamente una flecha.
3. Demuestre que una digráfica es fuertemente conexa si y sólo si contiene un camino cerrado generador.
4. Demuestre que si  $l, m$  y  $n$  son enteros con  $0 < l \leq m \leq n$ , entonces existe una gráfica simple  $G$  con  $\kappa = l$ ,  $\kappa' = m$  y  $\delta = n$ .

**Demostración:** Sean  $l, m, n$  perteneciente a los Enteros y  $G$  una gráfica con  $\kappa=l$ ,  $\kappa'=m$  y  $\delta=n$ , tenemos que  $0 < \kappa$  ya que una gráfica no puede tener conexidad menor que 0  $\rightarrow$  por proposición demostrada en clase esta gráfica tendrá la desigualdad  $0 < \kappa \leq \kappa' \leq \delta$  sustituyendo los valores  $0 < l \leq m \leq n$

Por lo tanto existe la gráfica (ya que la proposición demostrada en clase era un para todo y el paratodo implica el existe)  $\square$