Algoritmia y Complejidad

Titulación: Grado en Ingeniería Informática

Curso: 2023-2024

Trabajo: Ejercicio 7.24 del libro de SIPSER

Autores:

Yolanda Sarmiento Rincón

Juan Francisco Sobrino Ramírez

LENGUAJES Y CIENCIAS DE LA COMPUTACIÓN
UNIVERSIDAD DE MÁLAGA

UNIVERSIDAD DE MÁLAGA

Índice

- **Enunciado**
- **Definiciones Previas**
  - Clase P
  - Problema 2SAT
  - Ejemplos 2SAT
- **Resolución**
  - Enfoque
  - Grafo de Implicación
  - Algoritmo
  - Ejemplos
- **Referencias**

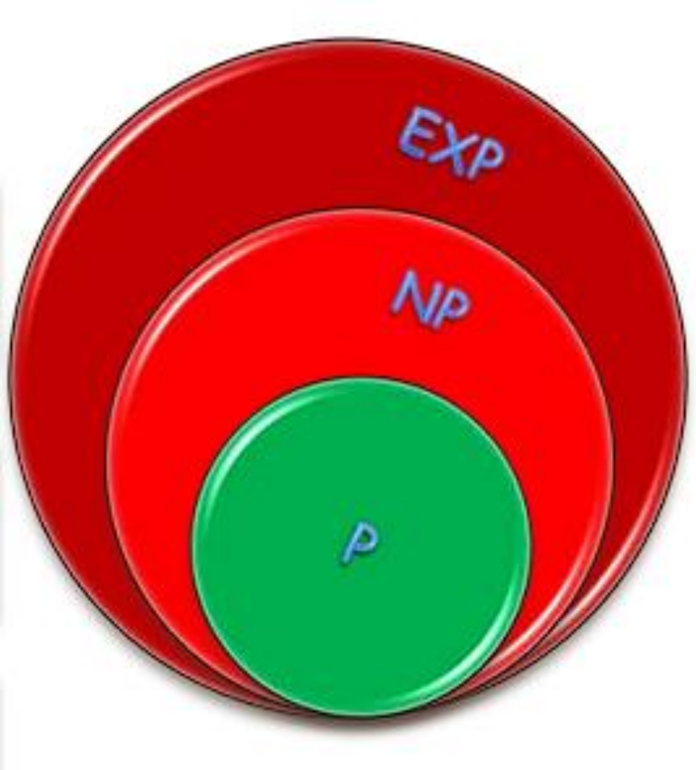
Enunciado

Una 2cnf-fórmula es un AND de cláusulas, dónde cada cláusula es un OR de a lo sumo dos literales. Dado $2SAT = \{\langle\phi\rangle \mid \phi$ es una 2cnf-fórmula satisfacible$\}$. Demuestra que $2SAT \in P$.

Clase P

P es la clase de lenguajes decididos por mTD en poli-tiempo:

$$\mathbf{P} = \bigcup_k TIME(n^k)$$



Problema 2SAT

El problema de decisión de 2SAT es el siguiente:

- **Entrada:** Una fórmula de forma normal conjuntiva con 2 literales por cláusula.
- **Pregunta:** ¿Existe una asignación de las variables que hace que la fórmula sea verdadera?

Ejemplos 2SAT

- **Ejemplo Válido:**

$\Phi_1 = (x_1 \vee \neg x_2) \wedge (\neg x_1 \vee x_2) \wedge (x_1 \vee x_2)$

Para $x_1 = 1$, $x_2 = 1$: $\Phi_1 \in 2SAT$

- **Ejemplo No Válido:**

$\Phi_2 = \Phi_1 \wedge (\neg x_1 \vee \neg x_2)$

Para todas las combinaciones posibles:

$\Phi_2 \notin 2SAT$

Enfoque

Supongamos que la cláusula (*A V B)* es verdadera, podemos expresarla como una implicación:

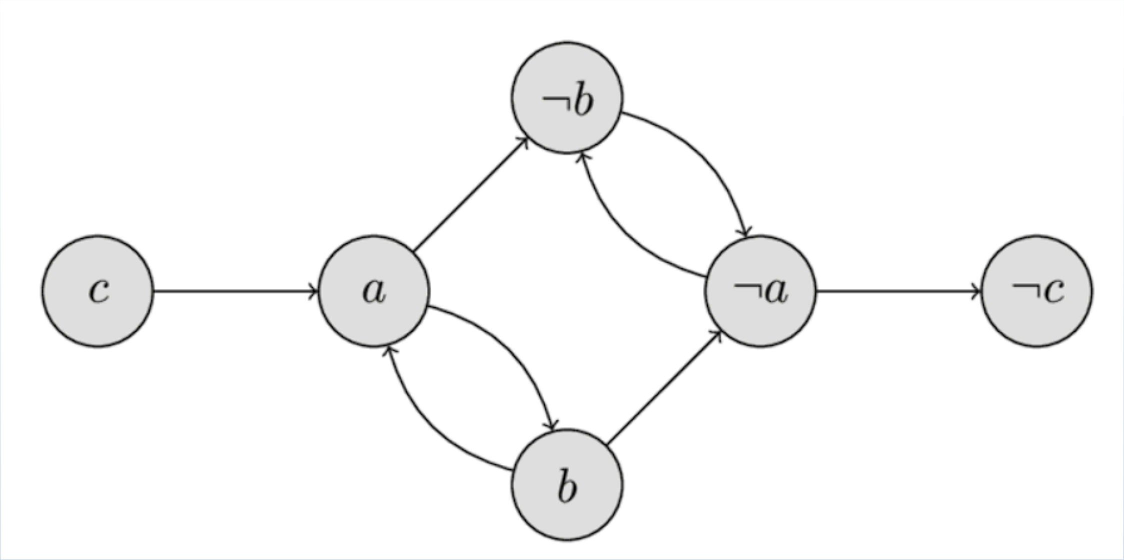
- Si A = 0, B debe ser 1, es decir (¬A → B)
- Si B = 0, A debe ser 1, es decir (¬B → A)

Es decir:

(*A V B)* es equivalente *(¬A → B) ∧ (¬B → A)*

Grafo de Implicación

Podemos construir un grafo dirigido en el que para cada variable *x*, habrá dos nodos. Las aristas corresponderán a las implicaciones.



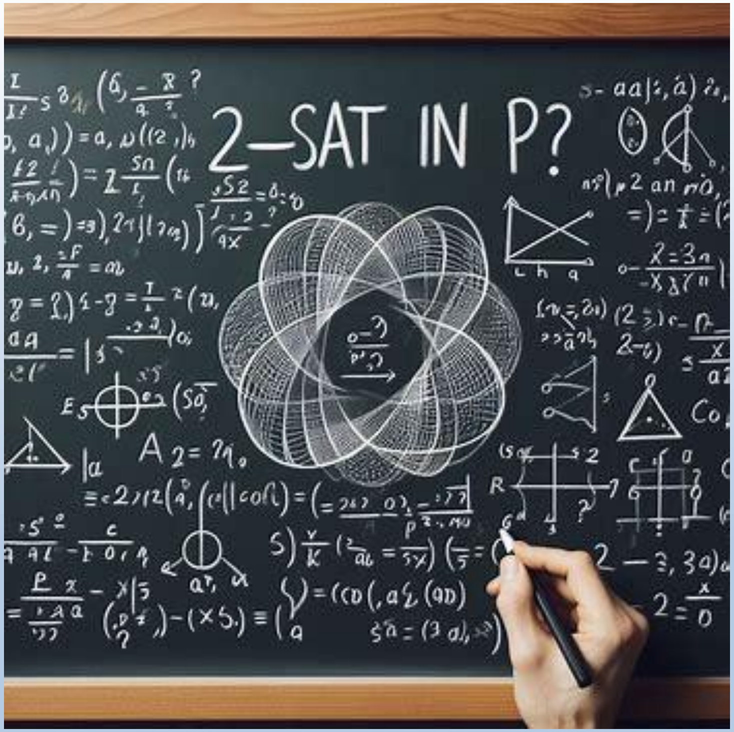
$$(a \vee \neg b) \wedge (\neg a \vee b) \wedge (\neg a \vee \neg b) \wedge (a \vee \neg c)$$

Construcción del Algoritmo

Φ satisfacible $\leftrightarrow$ No existe camino de x_i a $\neg x_i$ y de $\neg x_i$ a x_i simultáneamente

Esta búsqueda emplea tiempo polinómico con algoritmos de búsqueda en profundidad.

Luego 2SAT pertenece a P



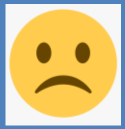
Demostración

Φ satisfacible $\longrightarrow$ No existe camino de x_i a $\neg x_i$ y de $\neg x_i$ a x_i simultáneamente

Si de $x_i \rightarrow^* \neg x_i$

y de $\neg x_i \rightarrow^* x_i$

Entonces: $xi \longleftrightarrow \neg xi$



Φ insatisfacible

Demostración

Φ satisfacible $\longleftarrow$ No existe camino de x_i a $\neg x_i$ y de $\neg x_i$ a x_i simultáneamente

1
- Tomamos x_i y si $x_i \longrightarrow \neg x_i$ seleccionamos $\neg x_i$

2
- Le asignamos verdadero al literal seleccionado y a todos los conectados.*

3
- Eliminamos los nodos asignados

Ejemplo 1

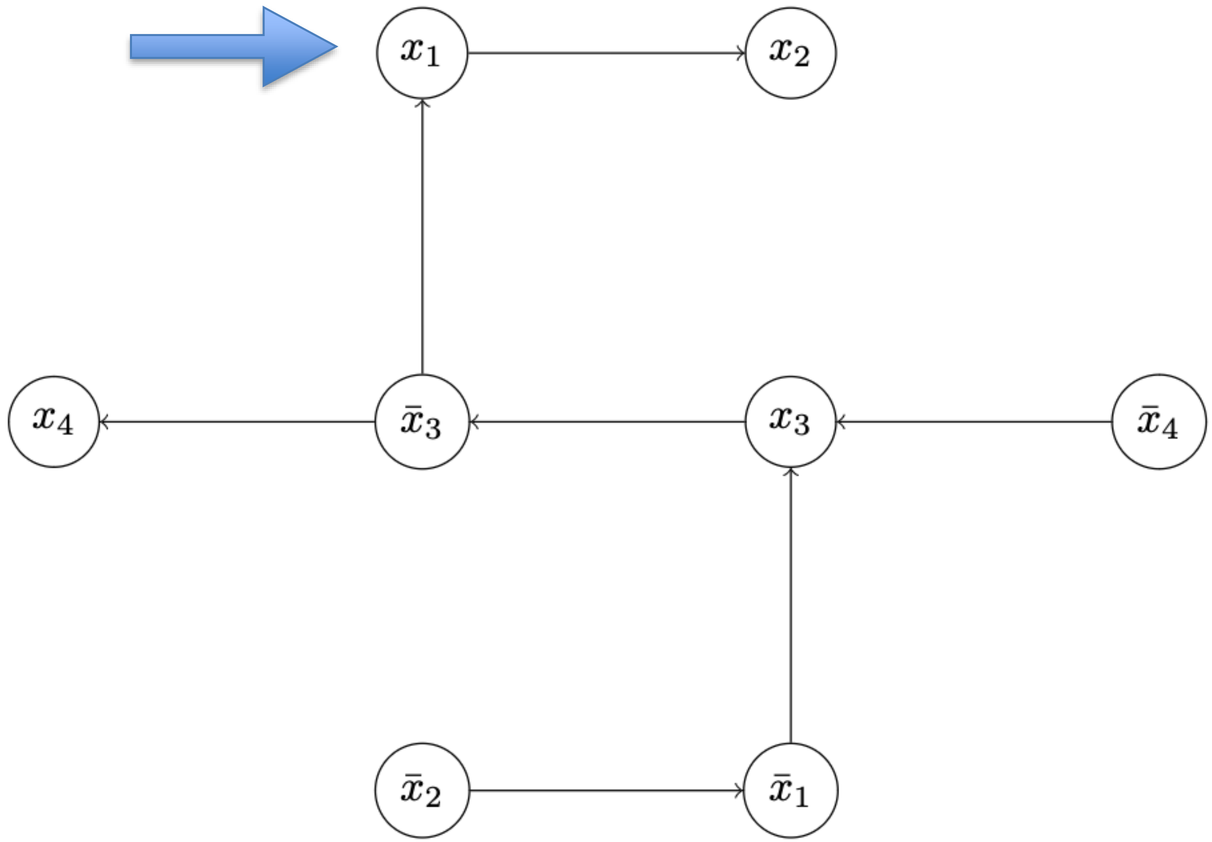
$$\Phi = (\bar{x}_1 \vee x_2) \wedge \bar{x}_3 \wedge (x_1 \vee x_3) \wedge \bar{x}_2$$

$$\begin{tikzcd}
x_1 \arrow[d] & \bar{x}_3 \arrow[l] & x_3 \arrow[l] & \bar{x}_1 \arrow[l] \\
x_2 \arrow[rrr] & & & \bar{x}_2 \arrow[u]
\end{tikzcd}$$

INSATISFACIBLE

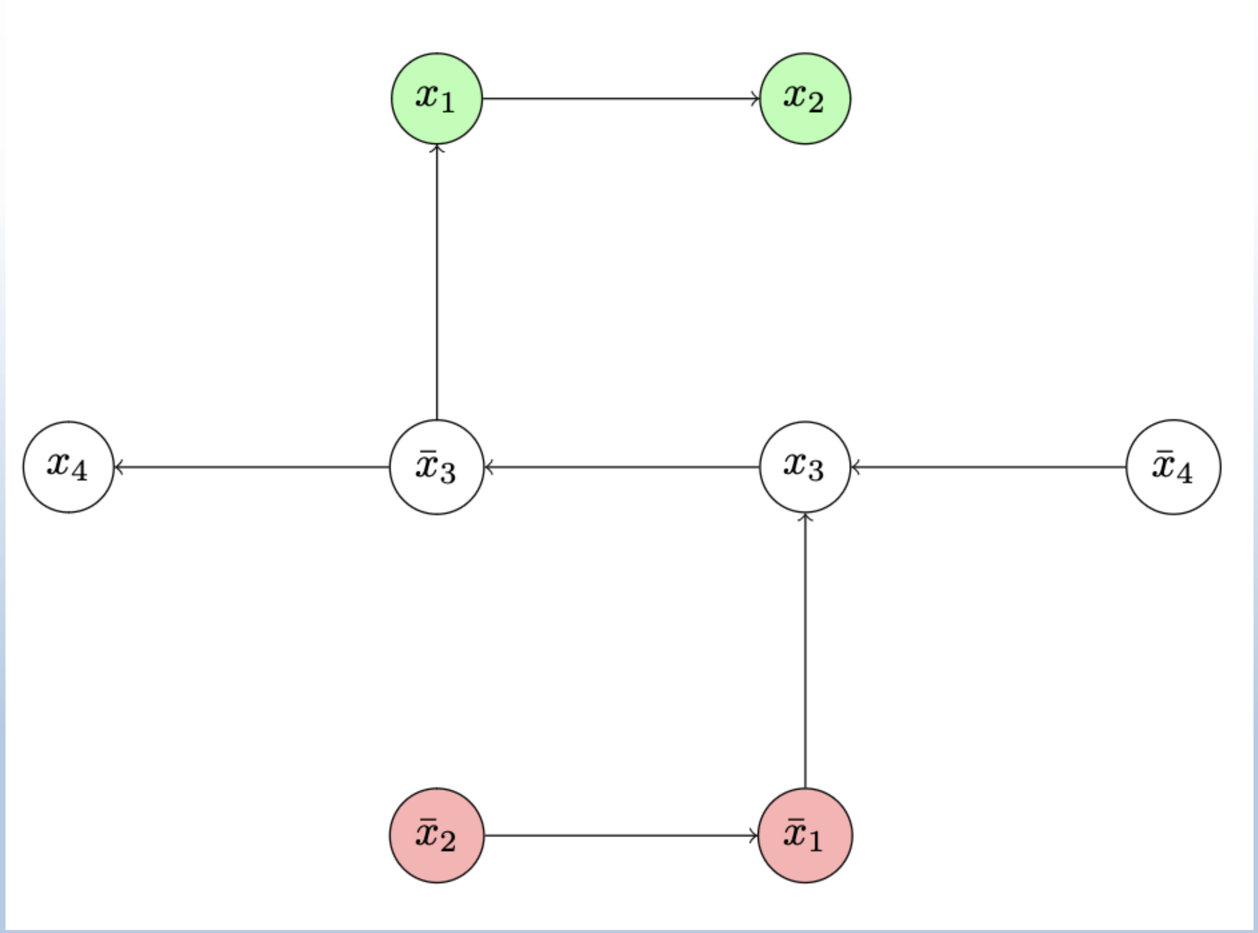
Ejemplo 2

$$\Phi = (\bar{x}_1 \vee x_2) \wedge \bar{x}_3 \wedge (x_1 \vee x_3) \wedge (x_3 \vee x_4)$$



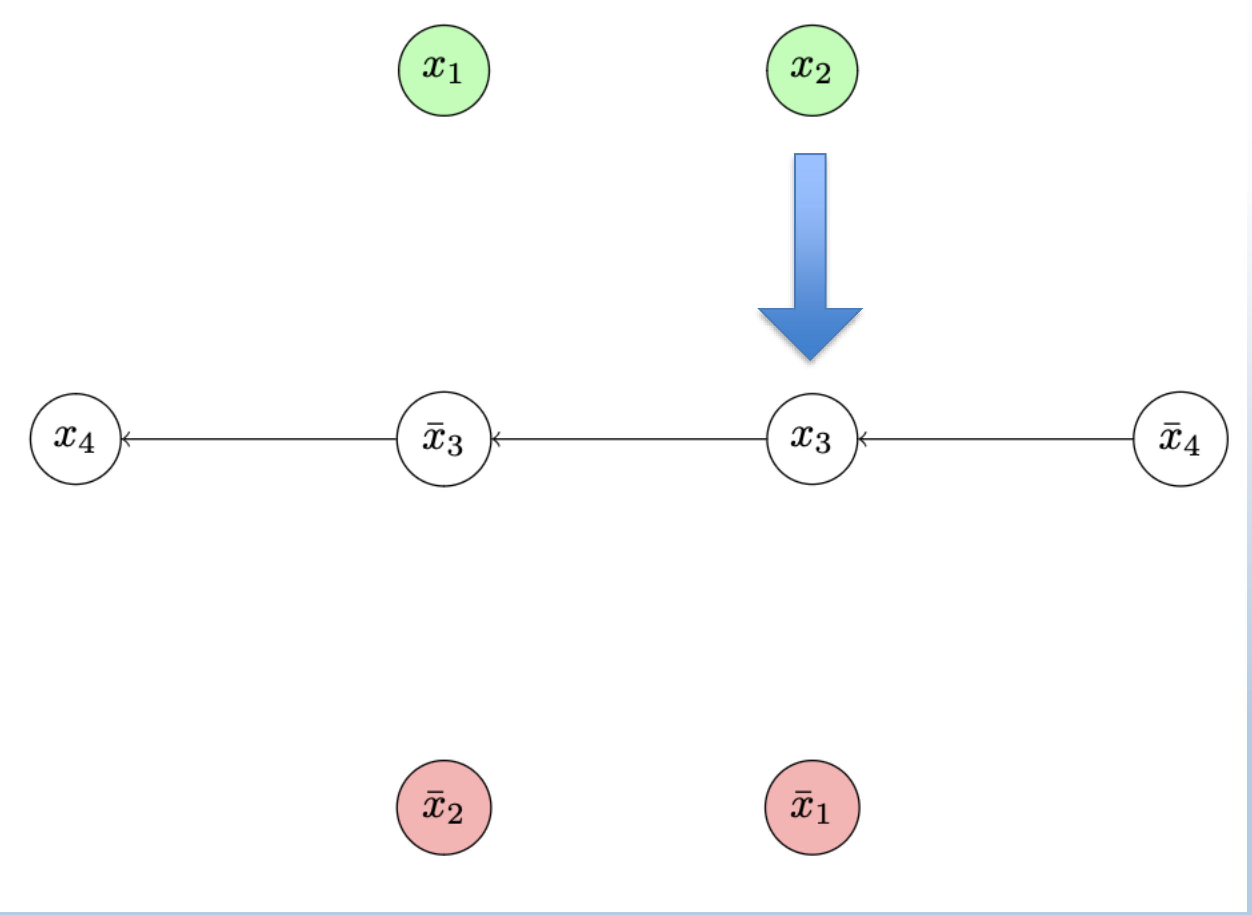
Ejemplo 2

$$\Phi = (\bar{x}_1 \vee x_2) \wedge \bar{x}_3 \wedge (x_1 \vee x_3) \wedge (x_3 \vee x_4)$$



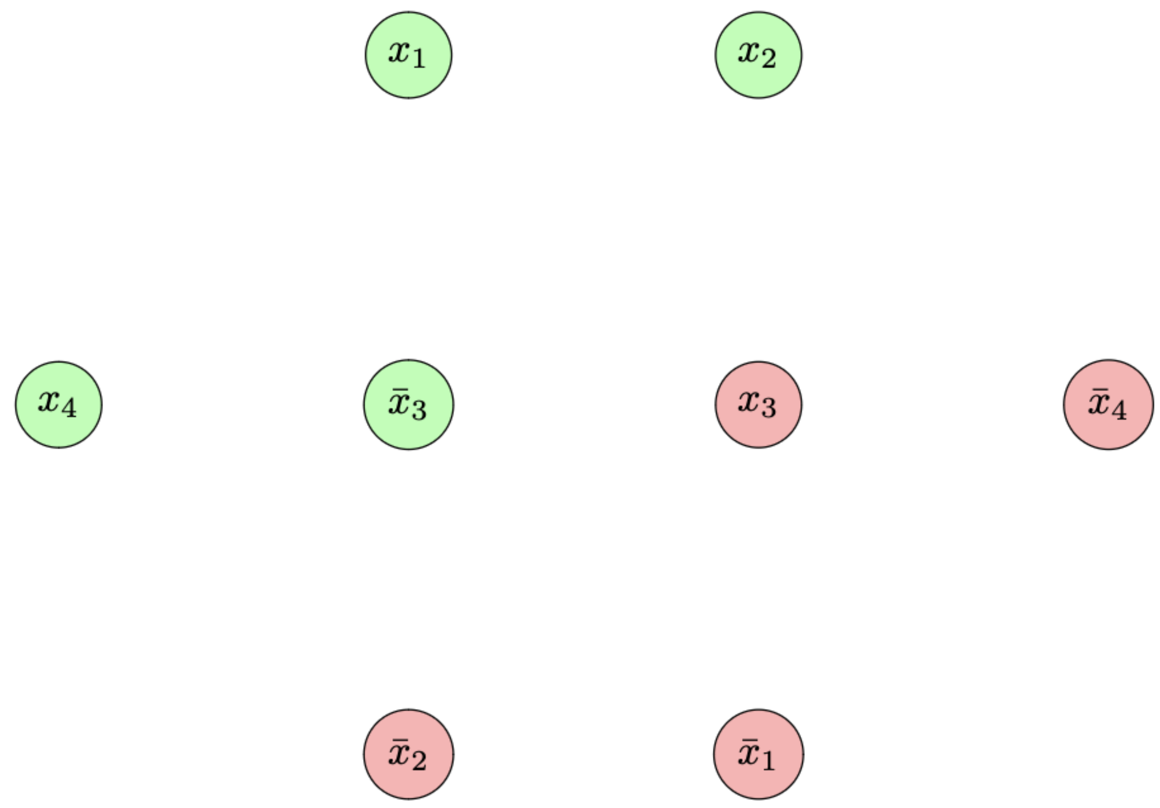
Ejemplo 2

$$\Phi = (\bar{x}_1 \vee x_2) \wedge \bar{x}_3 \wedge (x_1 \vee x_3) \wedge (x_3 \vee x_4)$$



Ejemplo 2

$$\Phi = (\bar{x}_1 \vee x_2) \wedge \bar{x}_3 \wedge (x_1 \vee x_3) \wedge (x_3 \vee x_4)$$



SATISFACIBLE

Referencias

- https://people.seas.harvard.edu/~cs125/fall14/lec19.pdf
- https://www.geeksforgeeks.org/
- https://cstheory.stackexchange.com/questions/
- https://en.wikipedia.org/wiki/2-satisfiability
- https://math.stackexchange.com/questions/