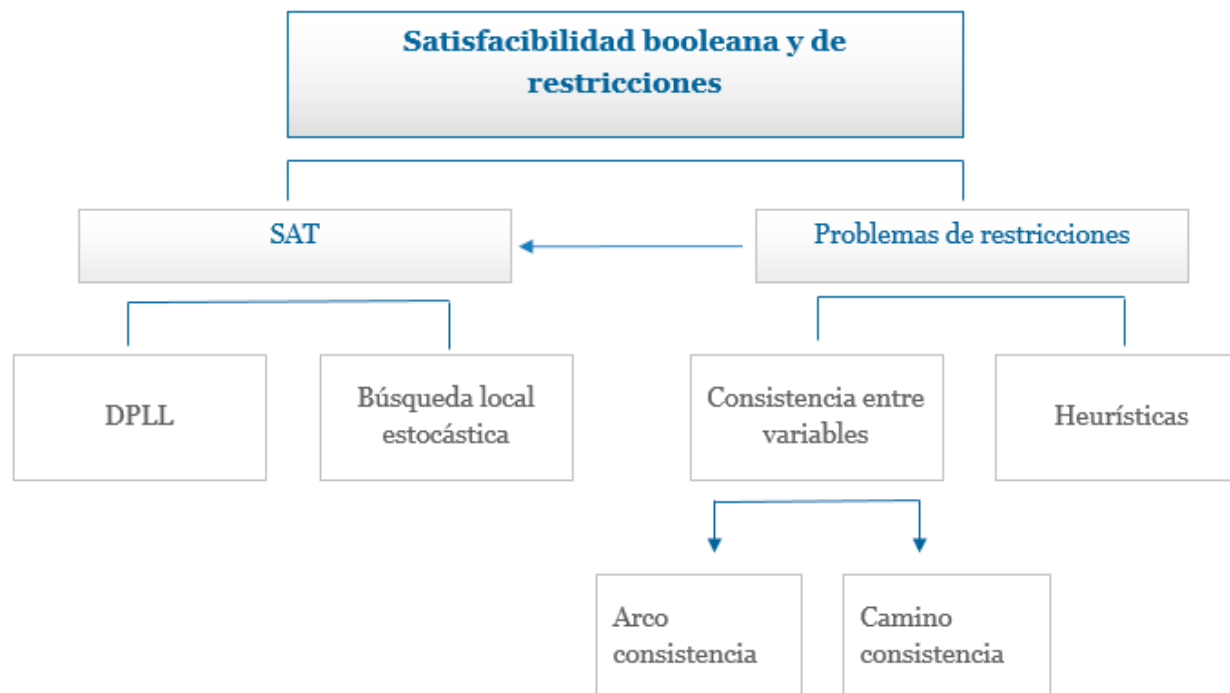


## Satisfacibilidad booleana y de restricciones

# ¿Cómo estudiar este tema?



## » TEMA 3. SATISFACIBILIDAD BOOLEANA Y DE RESTRICCIONES

IDEAS CLAVE	LO + RECOMENDADO	+ INFORMACIÓN
<p>¿Cómo estudiar este tema?</p> <p>Satisfacibilidad booleana</p> <p>Problemas de satisfacción de restricciones</p>	<p><b>Lecciones magistrales</b></p> <p><b>TV</b> Satisfacibilidad booleana y de restricciones</p> <p><b>No dejes de leer...</b></p> <p>Problemas de satisfacción de restricciones (CSP)</p> <p><b>No dejes de ver...</b></p> <p><b>TV</b> Videos de Wheeler Ruml sobre CSPs</p>	<p><b>A fondo</b></p> <p>Constraint Processing</p> <p>Estrategias de Sudoku</p> <p><b>Bibliografía</b></p>

# Objetivos de la clase

- Definir un problema de satisfacibilidad booleana / de restricciones.
- Reconocer cuando un problema se puede resolver mediante satisfacibilidad booleana / de restricciones.
- Representar problemas de satisfacibilidad booleana / de restricciones.
- Resolver un problema sencillo de satisfacibilidad booleana mediante el método DPLL.
- Identificar algoritmos que resuelven problemas de satisfacción de restricciones.

# Satisfacibilidad Booleana

El **problema de satisfacibilidad booleana** (SAT) consiste en hallar una asignación a una serie de variables booleanas para satisfacer una fórmula lógica.

## **SAT:** Formalización en lógica proposicional

Operadores permitidos:

- Conjunciones (*AND* lógico,  $\wedge$ )
- Disyunciones (*OR* lógico,  $\vee$ )
- Negaciones ( $\neg$ )
- Paréntesis

# Satisfacibilidad Booleana

## Forma Normal Conjuntiva (CNF):

- ☐ Representar una fórmula lógica como una **conjunción de disyunciones de literales**.
- ☐ Un literal de una variable es bien la variable afirmada o bien negada.
- ☐ Una disyunción de literales se denomina **cláusula**.

*Toda fórmula expresada en lógica proposicional puede pasarse a CNF usando transformaciones booleanas convencionales.*

**Problema muy estructurado** al que pueden reducirse otros problemas  
→ Un único resolutor SAT puede resolver diferentes problemas

# Satisfacibilidad Booleana

**EJEMPLO I:** ¿Es satisfacible la siguiente expresión booleana?  $p \wedge (q \vee \neg p)$

A cada asignación de variables que aparecen en la fórmula se le denomina modelo  $\rightarrow$  el objetivo es encontrar un modelo  $M$  que satisfaga la fórmula:  $M \models F$

$p$	$q$	$p \wedge (q \vee \neg p)$
0	0	0
0	1	0
1	0	0
1	1	1

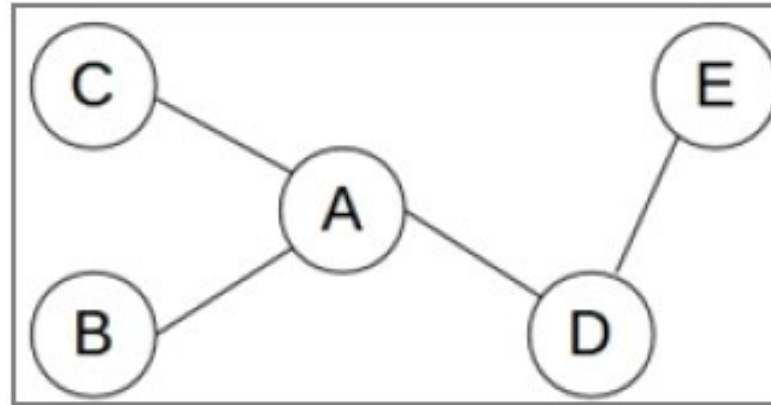
# Satisfacibilidad Booleana

**EJEMPLO 2:** ¿Es satisfacible la siguiente expresión booleana?

$$p \wedge (\neg p \vee q) \wedge \neg q$$

$p$	$q$	$\neg p$	$\neg q$	$\neg p \vee q$	$p \wedge (\neg p \vee q) \wedge \neg q$
0	0	1	1	1	0
0	1	1	0	1	0
1	0	0	1	0	0
1	1	0	0	1	0

# Satisfacibilidad Booleana



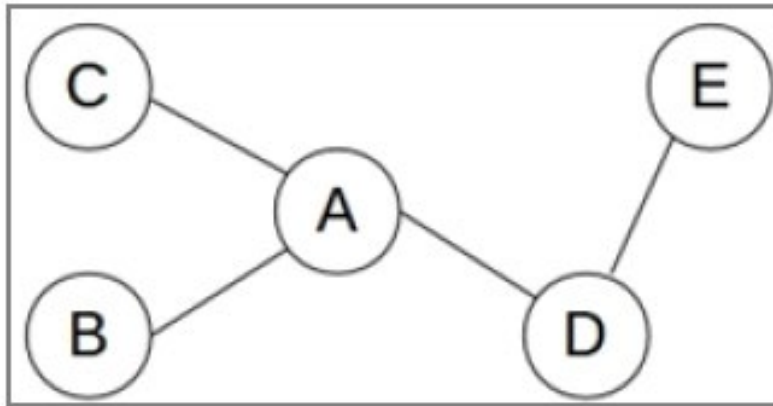
- ☐ Las zonas **A-B-C-D-E** tienen que estar vigiladas por guardias
- ☐ Una zona está vigilada si un guardia está en dicha zona o en una adyacente

$$F = (a \vee b \vee c \vee d) \wedge (a \vee b) \wedge (a \vee c) \wedge (a \vee d \vee e) \wedge (d \vee e)$$

*¿Cuál es la solución si sólo hay dos guardias?*



# Satisfacibilidad Booleana



- ❑ Las zonas **A-B-C-D-E** tienen que estar vigiladas por guardias
- ❑ Una zona está vigilada si un guardia está en dicha zona o en una adyacente

$$F = (a \vee b \vee c \vee d) \wedge (a \vee b) \wedge (a \vee c) \wedge (a \vee d \vee e) \wedge (d \vee e)$$

**¿Cuál es la solución si sólo hay dos guardias?** Se añaden a F cláusulas tal que se evite que 3, 4 ó 5 zonas tengan asignadas guardias de seguridad a la vez.

$$(\neg a \vee \neg b \vee \neg c \vee \neg d \vee \neg e)$$

Modela que al menos 1 zona no está vigilada de las 5 (no hay cinco guardias)

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

Modela que no haya cuatro guardias asignados a cuatro zonas

# Satisfacibilidad Booleana: DPLL

DPLL es un algoritmo para resolver el problema de SAT con una búsqueda en profundidad.

Características del problema SAT:

- ☐ No es relevante que la solución sea óptima
- ☐ Construir la solución **asignando valores incrementalmente** es beneficioso porque si una cláusula contiene:
  - el literal elegido, se puede eliminar la cláusula porque ya está satisfecha.
  - la negación de un literal, se puede eliminar dicho literal pues la satisfacibilidad de la cláusula depende del resto de literales.
- ☐ Se tiene un **límite en la profundidad de la solución**

# Satisfacibilidad Booleana: DPLL

Dado que es beneficioso resolver el problema SAT asignando valores incrementalmente...la solución puede verse como una secuencia de asignaciones de valores

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

$a=1$

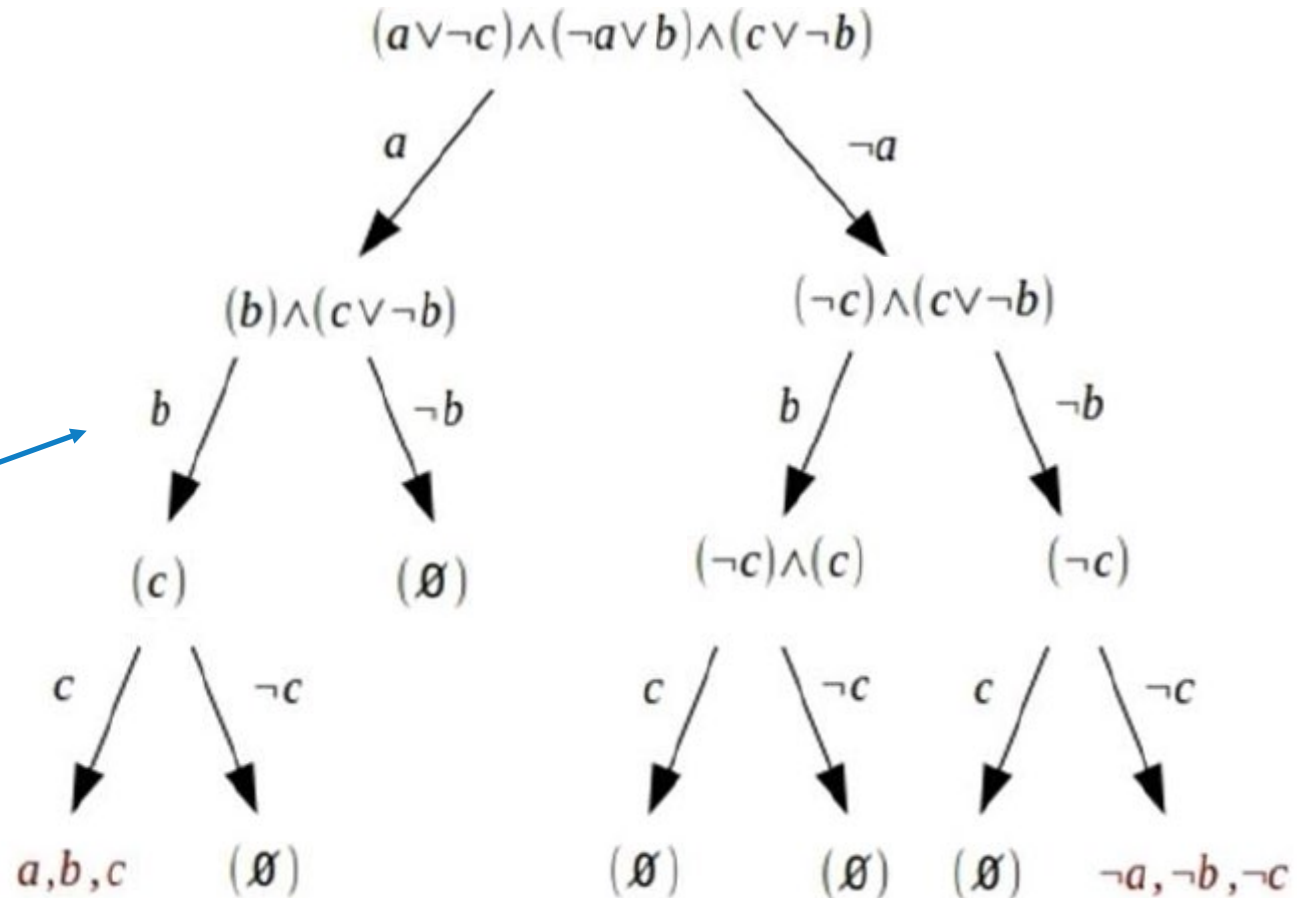
$$F = (b \vee c)$$



# Satisfacibilidad Booleana: DPLL

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

**Resolución unitaria:** si en un nodo hay una o más cláusulas unitarias, se eligen directamente los literales que los componen.



# Satisfacibilidad Booleana: DPLL

En cada nodo del árbol se decide:

- ☐ Qué variable asignar
- ☐ Qué valor dar a la variable

¡Estas decisiones tienen un impacto significativo en el número de ramas que hay que explorar!

Podemos utilizar heurísticas para estimar qué nodo es el que lleva a una solución con mayor probabilidad:

- ☐ Elegir la variable y el literal que más simplifiquen F
- ☐ Elegir la variable y el literal que satisfaga cláusulas difíciles de resolver

# Satisfacibilidad Booleana: búsqueda local estocástica

- ❑ La **búsqueda local estocástica** consiste en ir cambiando la asignación de valores de las variables hasta encontrar una solución, partiendo de un modelo inicial no válido generado aleatoriamente.
- ❑ El cambio se realiza entre asignaciones vecinas (a las que se puede llegar con un único cambio).
- ❑ La aleatoriedad permite tener cierta probabilidad de escapar de regiones poco prometedoras.

**GSAT:** Cambia una variable al azar o elige el cambio que minimice el nº de cláusulas sin satisfacer, alternando con una probabilidad.

**WalkSAT:** Elige una cláusula sin satisfacer al azar y cambia el valor de una de las variables que contiene para satisfacerla (de la variable que permita satisfacer más cláusulas o escogida al azar, alternando).

# Problemas de Satisfacción de Restricciones

- ❑ CSP – Constraint Satisfaction Problem
- ❑ Las variables tienen un número finito de valores
- ❑ Las restricciones no se limitan a disyunciones lógicas
- ❑ Se representan con tuplas  $\langle X, D, C \rangle$ 
  - $X$ : variables del problema
  - $D$ : dominio (valores posibles de las variables)
  - $C$ : restricciones del problema

# Problemas de satisfacción de restricciones

Para que una asignación sea solución debe ser consistente y completa

Consistente



No viola ninguna restricción de C

Completa

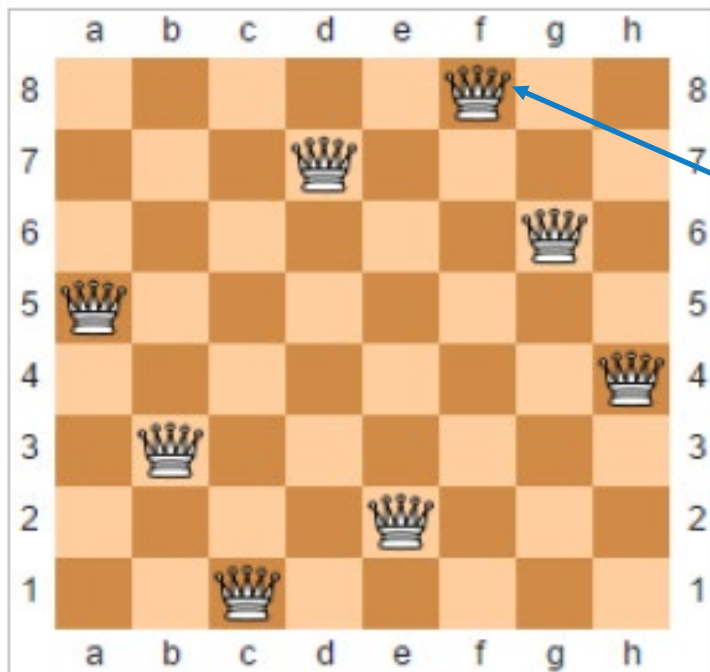


Todas las variables tienen un valor asignado



# Problemas de satisfacción de restricciones

**Ejemplo:** *Juego de las 8-Reinas* → Hay que colocar 8 reinas en un tablero de ajedrez de tal forma que ninguna reina amenace a otra reina.



- Dos variables por reina con dominio  $\{1..8\}$ , representando fila y columna de la posición de la reina

$X_{1f}=8$  y  $X_{1c}=6$

- Restricciones:
  - La primera y la segunda reina no pueden estar en la misma columna:
    - $R_{c12} = \{(1,2), (1,3), \dots, (1,8), (2,1), (2,3), \dots, (8,7)\}$  (*explícita*)
    - $R_{1c} \neq R_{2c}$
  - Dos reinas no pueden estar en la misma diagonal:  
 $|X_{1f} - X_{2f}| \neq |X_{1c} - X_{2c}|$

# Problema de satisfacción de restricciones

**Ejemplo:** *Sudoku* → Hay que rellenar cuadrícula con 9x9 casillas con números de 1 a 9 una, partiendo de algunos números ya dispuestos tal que no se repitan los números en una misma fila, columna o subcuadrícula.

	1	2	3	4	5	6	7	8	9
A	5	3			7				
B	6			1	9	5			
C		9	8					6	
D	8				6				3
E	4			8		3			1
F	7				2				6
G		6					2	8	
H				4	1	9			5
I					8			7	9

- 81 variables (A1, A2....I9)
- Dominio {1, 2, 3, 4, 5, 6, 7, 8, 9}
- Los números ya dispuestos tienen un único valor de dominio, el inicial.
- Utilizamos 27 restricciones *AllDiff* (fuerzan a que las variables afectadas sean diferentes):
  - $AllDiff(A1, A2, A3, A4, A5, A6, A7, A8, A9)$
  - $AllDiff(A1, A2, A3, B1, B2, B3, C1, C2, C3)$
  - Etc..

Fuente:

<http://commons.wikimedia.org/wiki/File:Sudoku-by-L2G-20050714.gif>

# Problemas de satisfacción de restricciones

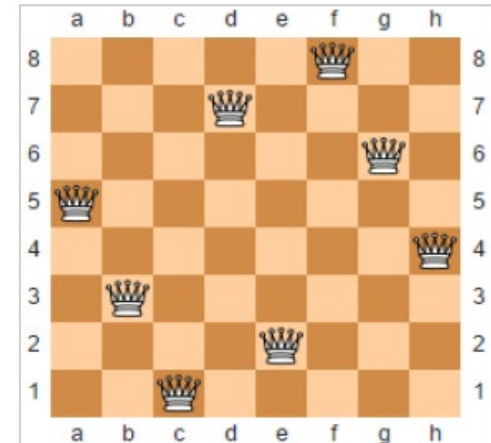
## ALGORITMOS

### Búsqueda GLOBAL

- ❑ Algoritmo de búsqueda en profundidad
- ❑ Profundidad acotada por el número de variables

### Búsqueda LOCAL

- ❑ Similar a SAT: asignaciones al azar



## HEURÍSTICAS

- ☐ Simplificar el problema
- ☐ Satisfacer las restricciones más difíciles
- ☐ Algunas populares:
  - Variable más restrictiva
  - Valores restantes mínimos
  - Valor menos restrictivo

**Desafío 4:** Referente al tema 3, contesta brevemente (no más de 100 palabras) a las siguientes cuestiones ¿Qué es la forma normal conjuntiva? ¿en qué tipo de problemas se puede utilizar? ¿en qué consisten estos problemas?

Es una representación de una fórmula lógica como una conjunción de disyunciones de literales. Se puede utilizar en problemas de satisfacibilidad booleana que consisten en hallar una asignación a una serie de variables booleanas para satisfacer una fórmula lógica.

Desafío 7: Referente al tema 4, en menos de 100 palabras responde a las siguientes cuestiones: ¿Qué quiere decir que la planificación automática sea independiente del dominio? ¿Qué ventaja tiene esta independencia del dominio?

Formulario con el desafío:

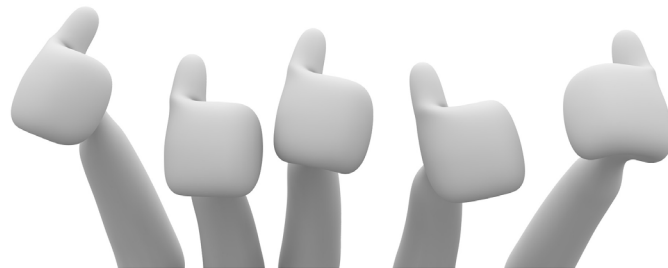
<https://forms.office.com/r/qhFf01NkmT>

# ¿Dudas?



¡Muchas gracias por vuestra atención!

¡Feliz y provechosa semana!







[www.unir.net](http://www.unir.net)