SAT [Solving]

Razonamiento Automático

Máster en Lógica, Computación e IA

Jesús Giráldez Crú

jgiraldez@ugr.es

El Problema SAT

El problema **SAT** [16] consiste en **decidir** si una **fórmula proposicional** F es o no es satisfactible:

Existe una asignación de $var(F)$ que **satisface** F?
(esa asignación se llama "*modelo*")
$\exists \sigma : var(F) \rightarrow \{0,1\}^{|var(F)|} \mid H(F_{|\sigma}) = 1$?

Fórmula	Modelos	SAT?
$x \vee y$	$\{x, y\}, \{x, \bar{y}\}, \{\bar{x}, y\}$	SAT
$x \wedge y$	$\{x, y\}$	SAT
$(x \vee y) \wedge z$	$\{x, y, z\}, \{x, \bar{y}, z\}, \{\bar{x}, y, z\}$	SAT
$(x \vee y) \wedge z \wedge \neg z$	-	UNSAT

El Problema SAT

El problema **SAT** [16] consiste en **decidir** si una **fórmula proposicional** F es o no es satisfactible:

Existe una asignación de $var(F)$ que **satisface** F?
(esa asignación se llama "*modelo*")
$\exists \sigma : var(F) \rightarrow \{0,1\}^{|var(F)|} \mid H(F_{|\sigma}) = 1$?

Fórmula	Modelos	SAT?
$x \vee y$	$\{x, y\}, \{x, \bar{y}\}, \{\bar{x}, y\}$	SAT
$x \wedge y$	$\{x, y\}$	SAT
$(x \vee y) \wedge z$	$\{x, y, z\}, \{x, \bar{y}, z\}, \{\bar{x}, y, z\}$	SAT
$(x \vee y) \wedge z \wedge \neg z$	-	UNSAT

El Problema SAT

El problema **SAT** [16] consiste en **decidir** si una **fórmula proposicional** F es o no es satisfactible:

Existe una asignación de $var(F)$ que **satisface** F?
(esa asignación se llama "*modelo*")
$\exists \sigma : var(F) \rightarrow \{0,1\}^{|var(F)|} \mid H(F_{|\sigma}) = 1$?

Fórmula	Modelos	SAT?
$x \vee y$	$\{x, y\}, \{x, \bar{y}\}, \{\bar{x}, y\}$	SAT
$x \wedge y$	$\{x, y\}$	SAT
$(x \vee y) \wedge z$	$\{x, y, z\}, \{x, \bar{y}, z\}, \{\bar{x}, y, z\}$	SAT
$(x \vee y) \wedge z \wedge \neg z$	-	UNSAT

SAT: Ejemplo

$$F = c_1 \wedge c_2 \wedge c_3 \wedge c_4 =$$
$$= (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$

F es SAT?

x_1	x_2	x_3	$H(c_1)$	$H(c_2)$	$H(c_3)$	$H(c_4)$	$H(F)$
0	0	0					
0	0	1					
0	1	0					
0	1	1					
1	0	0					
1	0	1					
1	1	0					
1	1	1					

SAT: Ejemplo

$$F = c_1 \wedge c_2 \wedge c_3 \wedge c_4 =$$
$$= (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$

F es SAT?

x_1	x_2	x_3	$H(c_1)$	$H(c_2)$	$H(c_3)$	$H(c_4)$	$H(F)$
0	0	0	0				
0	0	1	1				
0	1	0	1				
0	1	1	1				
1	0	0	1				
1	0	1	1				
1	1	0	1				
1	1	1	1				

SAT: Ejemplo

$$F = c_1 \wedge c_2 \wedge c_3 \wedge c_4 =$$
$$= (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$

F es SAT?

x_1	x_2	x_3	$H(c_1)$	$H(c_2)$	$H(c_3)$	$H(c_4)$	$H(F)$
0	0	0	0	1			
0	0	1	1	1			
0	1	0	1	1			
0	1	1	1	1			
1	0	0	1	0			
1	0	1	1	0			
1	1	0	1	1			
1	1	1	1	1			

SAT: Ejemplo

$$F = c_1 \wedge c_2 \wedge c_3 \wedge c_4 =$$
$$= (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$

F es SAT?

x_1	x_2	x_3	$H(c_1)$	$H(c_2)$	$H(c_3)$	$H(c_4)$	$H(F)$
0	0	0	0	1	1		
0	0	1	1	1	1		
0	1	0	1	1	0		
0	1	1	1	1	1		
1	0	0	1	0	1		
1	0	1	1	0	1		
1	1	0	1	1	0		
1	1	1	1	1	1		

SAT: Ejemplo

$$F = c_1 \wedge c_2 \wedge c_3 \wedge c_4 =$$
$$= (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$

F es SAT?

x_1	x_2	x_3	$H(c_1)$	$H(c_2)$	$H(c_3)$	$H(c_4)$	$H(F)$
0	0	0	0	1	1	1	
0	0	1	1	1	1	0	
0	1	0	1	1	0	1	
0	1	1	1	1	1	0	
1	0	0	1	0	1	1	
1	0	1	1	0	1	0	
1	1	0	1	1	0	1	
1	1	1	1	1	1	0	

SAT: Ejemplo

$$F = c_1 \wedge c_2 \wedge c_3 \wedge c_4 =$$
$$= (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$

F es SAT?

x_1	x_2	x_3	$H(c_1)$	$H(c_2)$	$H(c_3)$	$H(c_4)$	$H(F)$
0	0	0	0	1	1	1	**0**
0	0	1	1	1	1	0	**0**
0	1	0	1	1	0	1	**0**
0	1	1	1	1	1	0	**0**
1	0	0	1	0	1	1	**0**
1	0	1	1	0	1	0	**0**
1	1	0	1	1	0	1	**0**
1	1	1	1	1	1	0	**0**

F es **UNSAT**

Ejemplo: *Vertex Coloring*

Un grafo G(V,A) tiene una *N*-coloración?

Ejemplo: *Vertex Coloring*

Un grafo G(V,A) tiene una N-coloración?

1. Codifica en lógica proposicional:

$$
\begin{array}{ll}
F = & \\
(x_{i1} \vee x_{i2} \vee \cdots \vee x_{iN}) \wedge & \forall i \in V \\
(x_{ij} \rightarrow \neg x_{ij'}) \wedge & \forall i \in V, 1 \leq j, j' \leq N \\
(x_{i1} \rightarrow \neg x_{j1}) \wedge \cdots \wedge (x_{iN} \rightarrow \neg x_{jN}) & \forall i \in V, \langle i,j \rangle \in A
\end{array}
$$

2. Resuelve con SAT:

   $SAT(F)$?

3. [*Opcional*] Recodifica el modelo (si F es SAT)

$$IA \overset{?}{\subseteq} SAT$$

$$\text{IA} \supseteq \text{SAT}$$

- SAT es un problema central en **Ciencias de la Computación** (CS) e **Inteligencia Artificial** (AI)
  - SAT es el primer problema NP-completo: **complejidad computacional**
  - SAT es extensivamente usado en muchas **aplicaciones de AI**

- **SAT** como **área clásica de investigación** en CS/AI:
  - Satisfacción y Optimización de Restricciones
  - Búsqueda completa (en árboles) + propagación de restricciones
  - Búsqueda local (estocástica)
  - ...

Decisión (=Satisfacción) $\approx$ Optimización

- Si $\exists$ `optimiza(problema)`:

  $\texttt{decide(problema)} = \{$
  $\quad\quad \mathit{optimiza}(\mathit{problema}) \wedge \mathit{coste} == 0;$
  $\}$

Decisión (=Satisfacción) $\approx$ Optimización

- Si $\exists$ `optimiza(problema)`:

  `decide(problema)` $= \{$
  $\quad$ *optimiza*(*problema*) $\wedge$ *coste* $==$ 0;
  $\}$

- Si $\exists$ `decide(problema)`:

  `optimiza(problema)` $= \{$
  $\quad$ *coste* $= \infty$;
  $\quad$ `do` $\{$
  $\quad\quad$ *optimo* $=$ *coste*;
  $\quad$ $\}$ `while`(*decide*(*problema* $\wedge$ *coste* $<$ *optimo*));
  $\}$

Agenda

Preliminares (1)

- **Variables** Booleanas/Proposicionales
  - Sólo toman dos valores: 0/1, `true`/`false`, ...
  - $x, y, z, \ldots$
- **Literales**
  - Una variables o su negación
  - $x, \neg x, \ldots$
- **Cláusulas**
  - Disjunción de literales
  - $(x \vee \neg y \vee \neg z)$
- Fórmula en Forma Normal Conjuntiva (**CNF**)
  - Conjunción de cláusulas
  - $(x) \wedge (x \vee \neg y \vee \neg z) \wedge (\neg x \vee y \vee z) \wedge (\neg x \vee \neg y \vee \neg z)$
- k-CNF
  - Todas las cláusulas tienen (como máximo) k literales
  - 2SAT, 3SAT, ...

Preliminares (2)

- **Resolver SAT** sobre fórmula en CNF
  - Satisfacer (al menos) un literal de cada cláusula
  - $(x) \wedge (x \vee \neg y \vee \neg z) \wedge (\neg x \vee y \vee z) \wedge (\neg x \vee \neg y \vee \neg z)$
  - $(x) \wedge (x \vee \neg y \vee \neg z) \wedge (\neg x \vee y \vee z) \wedge (\neg x \vee \neg y \vee \neg z)$
  - *Traducción* a CNF en PTIME
- Solución: **SAT** (+**modelo**) o **UNSAT** (+**prueba**)
- **Modelo**
  - Asignación de $var(F)$ que satisface F
- **Prueba**
  - Conjunto de pasos para obtener la cláusula vacía
  - Una cláusula vacía (sin literales) no se puede satisfacer!

NP-completitud

- **SAT**: primer problema **NP-completo**[1] [23]: NP y NP-hard
  - **Verificación** de soluciones **eficiente**:
    - Comprobar si cada cláusula es satisfecha: $\mathcal{O}(\#clauses)$
  - Y **encontrarlas**?
    - Fuerza bruta: $\Theta(2^{\#var})$
    - Branch and bound: $\mathcal{O}(exp(\#var))$
    - $\exists$ algoritmo **eficiente**?
- $P \stackrel{?}{=} NP$ [66]
  - Clay Mathematics Institute: Uno de los siete **problemas matemáticos del milenio** [21]
  - Premio por resolver alguno: **1.000.000 USD**
- Si SAT se resuelve eficientemente, **cualquier problema NP-completo se resuelve eficientemente**: *vertex coloring*, mochila (*knapsack*), ciclo hamiltoniano, viajante viajero (*TSP*), clique, ... [33]

[1] Sólo en el caso $(2+\epsilon)$SAT

Resolución (1)

- Regla de **resolución**:

$$\frac{x \vee C \qquad \neg x \vee D}{C \vee D}$$

- Resolución es un **sistema de pruebas** (*proof system*) [17]
- **Refutación** por resolución: $F \vdash_r \bot$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{C_a \qquad C_b}{C_1' \qquad C_c}}{C_2' \qquad C_d}}{\ddots \qquad C_i'}}{C_j' \qquad C_e}}{\ddots \qquad C_k}$$
$$\bot$$

Resolución (2)

- Podemos analizar *todas* las posibles **refutaciones** de una fórmula y elegir *la mejor*:
  - **Longitud**: número de pasos de resolución
  - **Anchura**: tamaño de la cláusula más larga
  - **Espacio**: número de cláusulas en memoria

- La **longitud** de la refutación más corta es una **cota inferior del tiempo** de un algoritmo[2]
  - Existen fórmulas que requieren refutaciones de longitud exponencial [38, 55, 54, 53]
- La refutación más eficiente en **espacio** es una **cota inferior del espacio** de un algoritmo[2]
  - Existen fórmulas con *tradeoffs* entre longitud y espacio [11, 53]
- **Anchura** también relacionada con longitud (y espacio) [12, 53]

[2] Algoritmos que resuelven SAT implementando resolución

Otros sistemas de prueba

- Polynomial Calculus (PC) [22, 1]:

$$(x \vee y \vee \neg z) \quad \rightsquigarrow \quad xy\bar{z} = 0$$

$$F \overset{?}{\vdash}_{PC} (0 = 1)$$

- Cutting Planes (CP) [24]:

$$(x \vee y \vee \neg z) \quad \rightsquigarrow \quad x + y + (1 - z) > 0$$

$$F \overset{?}{\vdash}_{CP} (0 \geq 1)$$

- ...

Ejemplo: Pigeon Hole Principle (PHP)

- **PHP_k^n** : n palomas en k huecos (sin repetición)
- PHP_n^{n+1} es (obviamente) UNSAT
- La **teoría** nos dice:
  - PHP requiere refutación de **resolución** de longitud **exponencial** [38, 55, 54, 53]
  - Se puede resolver **eficientemente** con **CP** [24]
- En la **práctica**:
  - Algoritmos que implementan resolución no pueden resolver PHP con decenas de variables en varios días
  - Algoritmos que implementan CP pueden resolver PHP con millones de variables en pocos segundos

Proof Complexity

- Los algoritmos que resuelven SAT[3] implementan un sistema de pruebas
- Estudiando la prueba que un algoritmo genera podemos tener información sobre la complejidad del algoritmo
- Esta rama de investigación es **proof complexity**

- (Algunas) **Preguntas relevantes** en *proof complexity* [53]:
  - Una (familia de) fórmula(s) se puede resolver eficientemente?
  - Una implementación (basada en cierto sistema de pruebas) es eficiente?
  - ...

[3] Específicamente, que demuestran UNSAT

Instancias SAT industriales

- Algoritmos competitivos implementan **resolución**
  - $\nexists$ implementación eficiente de CP u otros [31, 65]
- $\rightsquigarrow$ **SAT**: problema **duro en el peor caso**... a veces incluso en **en el caso medio** (eg, random k-CNF) [20]
  - Fórmulas pequeñas irresolubles en la práctica

- Por contra, **ciertos problemas** (varios órdenes de magnitud más grandes) se pueden resolver bastante **rápido**
- Representan problemas de **muy diferentes dominios**, pero todos ellos son dominios "**industriales**" o "**del mundo real**"
- Curiosamente todos ellos tienen *rendimientos similares*. **¿Por qué?**

Estructura de las instancias industriales

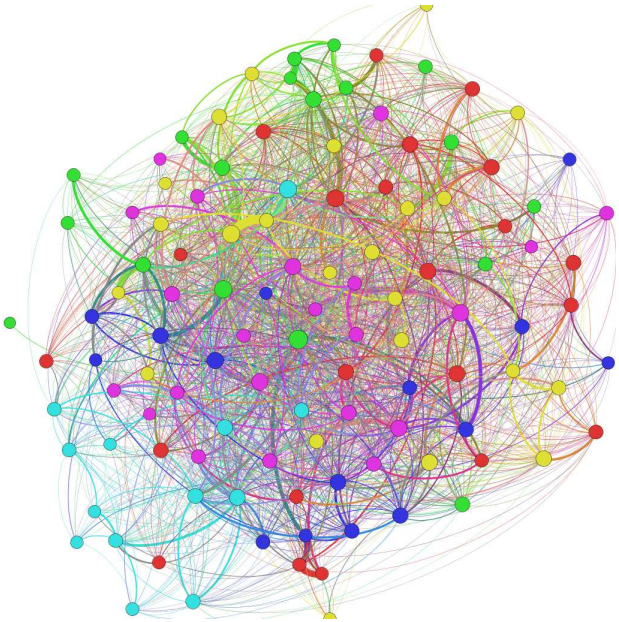
- **Posible explicación**: las instancias industriales tienen una **estructura "oculta"** que es explotada por los SAT solvers
- Estructura bien diferenciada de la (no) estructura de las instancias aleatorias

- Perspectiva teórica: propiedades que explican características de la refutación [30]
- Perspectiva aplicada: propiedades que explican la efectividad de los algoritmos [9, 45]

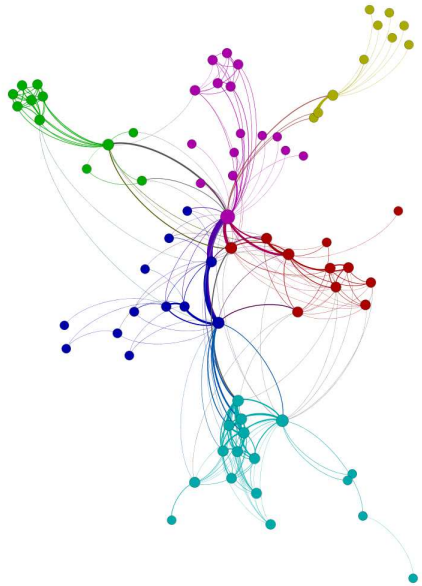
- **Posible estrategia**: Representar fórmulas SAT como **grafos**

Instancias aleatorias vs Instancias industriales

Grafo "*aleatorio*"



Grafo con "*estructura*"



Preguntas Abiertas

- ¿Qué **características** comunes tienen las instancias **SAT industriales**?
  - Alta variabilidad en las ocurrencias de variables (distribuciones *power-law*) [5]
  - Estructura de comunidades [8]
  - Dimensión fractal [2]
  - ...
- ¿Estas características **explican el rendimiento** de los algoritmos?
  - Modelos aleatorios *pseudo-industriales* [35, 34, 36, 6]
  - Soluciones *ad-hoc* [14]
  - Algoritmos portfolios [4, 3]
  - ...
- ¿Cómo podemos **mejorar** estos **algoritmos**? [9]

Aplicaciones de IA basadas en SAT

- Numerosas **aplicaciones de IA** que usan internamente un **SAT solver** como tecnología clave:
  - Spatial-temporal reasoning in IA systems
  - Controllers for sensory-based robots
  - Process concurrency and synchronization (concurrency)
  - Geometric coherence in Computer Graphics
  - Constraints databases
  - Sequence alignment in Bioinformatics
  - Optimization in Operation Research
  - Production planning
  - Staff scheduling
  - Resource allocation
  - Circuit design
  - Option trading
  - DNA sequencing
  - Software verification
  - Cryptography
  - ...

Ejemplo1: Bounded Model Checking (BMC) [15]

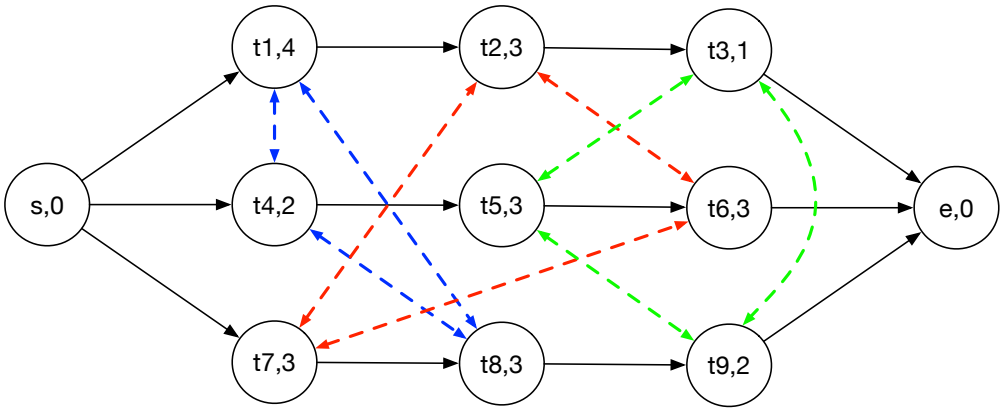
- Sistema de transiciones: $K = (V, S, I, T)$
- **BMC**: algoritmo de búsqueda de errores (*bug-finding*)

$$I(V_0) \wedge T(V_0, V_1) \wedge \cdots \wedge T(V_{k-1}, V_k) \wedge Bad(V_k)$$

- *Desenrrollar* la relación de transición un número de iteraciones k (**unrolling depth**)
- Resolver iterativamente SAT(BMC_k) para k creciente
- Si BMC_k es satisfactible, hay un *bug* (en k iteraciones)

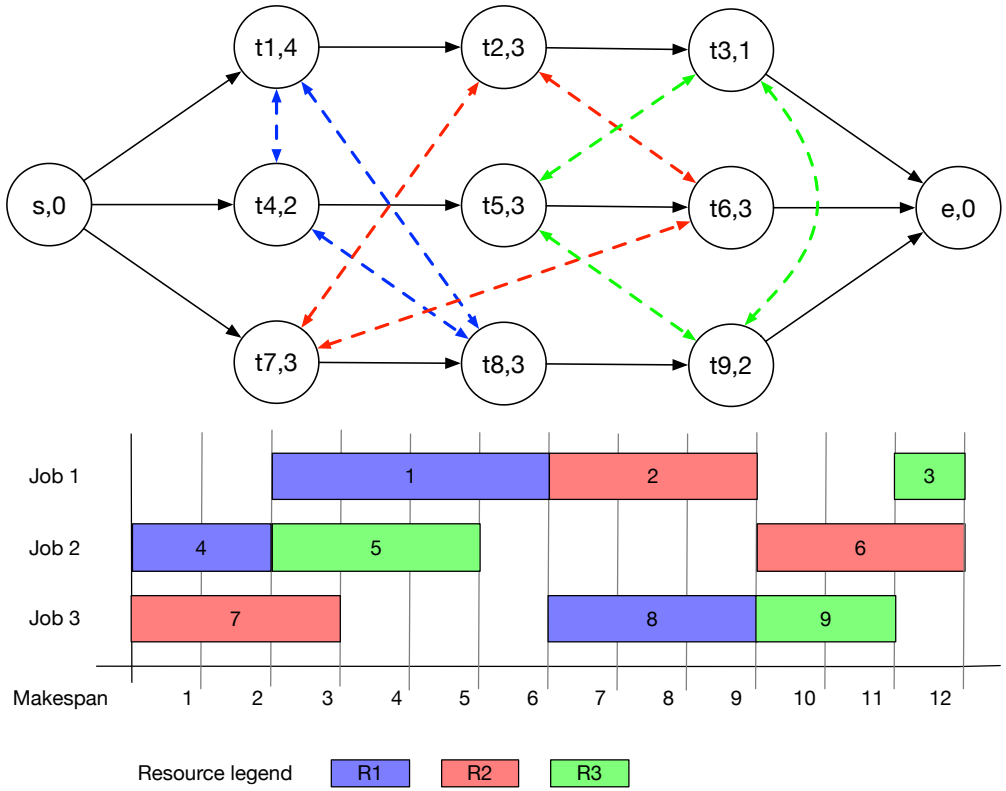
Ejemplo2: Job-Shop Scheduling Problem [56]

- Organizar las **tareas** de distintos **trabajos**, respetando **restricciones de prioridad y recursos**, **minimizando el makespan**.



Ejemplo2: Job-Shop Scheduling Problem [56]

- Organizar las **tareas** de distintos **trabajos**, respetando **restricciones de prioridad y recursos**, **minimizando el makespan**.



Métodos Completos e Incompletos

- **Métodos completos**: resuelven SAT y demuestran UNSAT (**garantizado**) [25]
  - DP: Davis-Putnam [27]
  - DPLL: Davis-Putnam-Logemann-Loveland [26]
  - CDCL: Conflict-Driven Clause Learning [62]
  - Look-ahead [39]

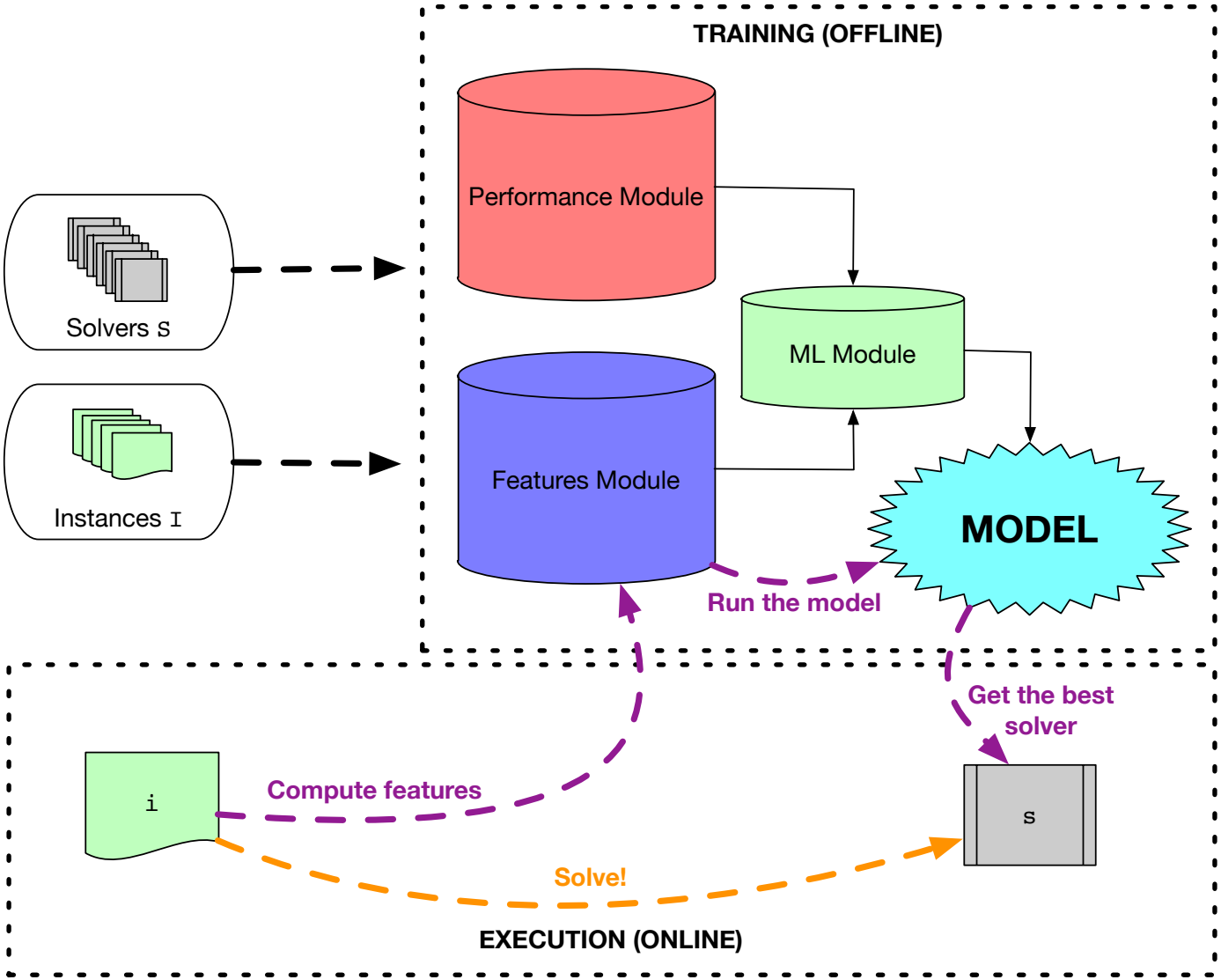
- **Métodos incompletos**: pueden resolver SAT (**no garantizado**) [46]
  - SLS: Stochastic Local Search
    - GSAT [58]
    - WalkSAT [57]
    - Variantes [51, 19, 32, 64, 42]
  - SP: Survey Propagation [18]
  - NeuroSAT (neural networks) [60, 59, 61]

Portfolios y Autoconfiguradores

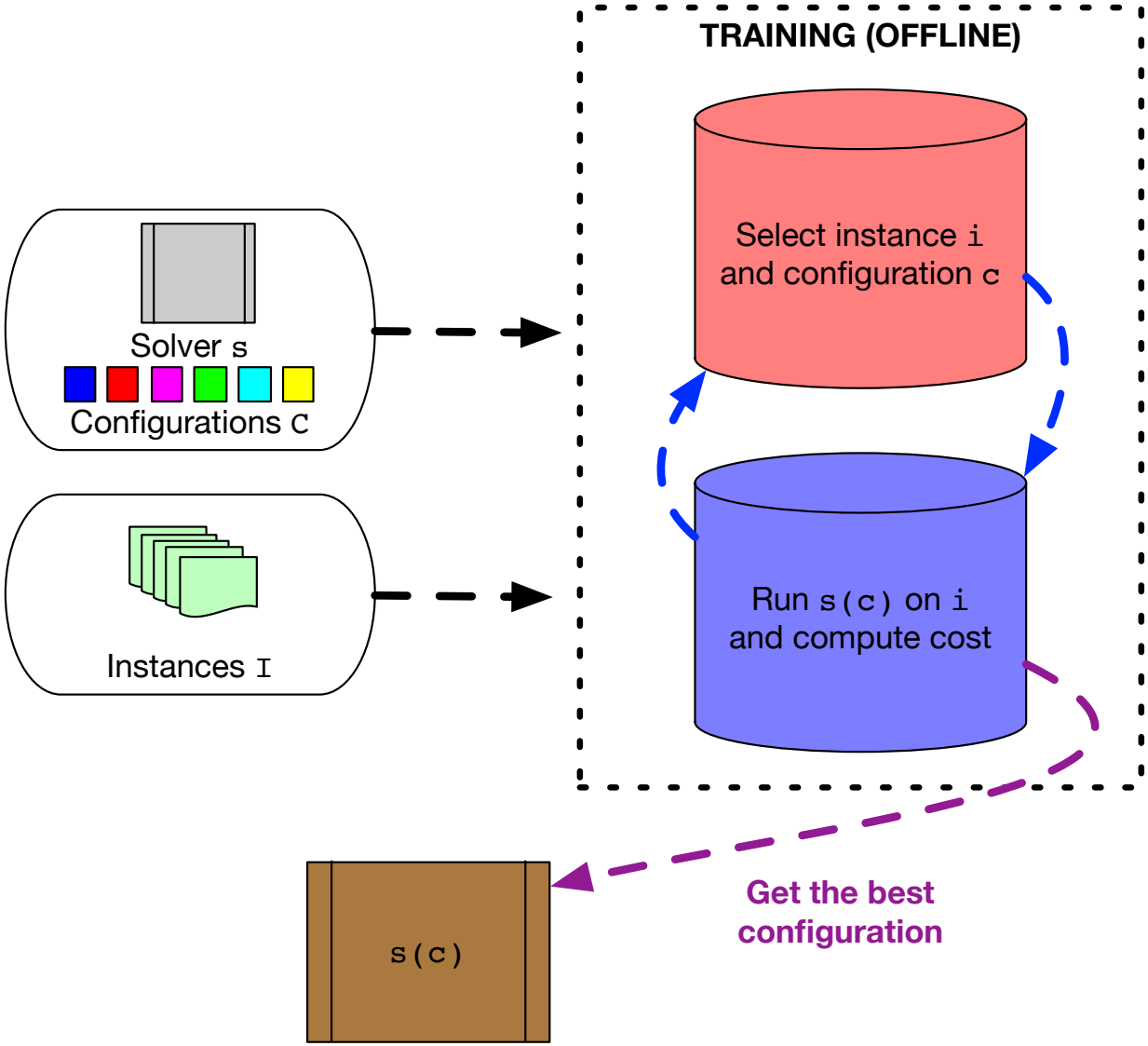
- **Portfolios** [68, 67, 44, 48]
  - Muchos algoritmos para resolver SAT
  - Para cada instancia, cada algoritmo tiene diferentes rendimientos
  - Para una instancia dada, ¿se puede aprender cuál es el mejor algoritmo para resolverla?
  - Sí, usando ML!

- **Autoconfiguradores** [41, 10, 28, 40]
  - Cada algoritmo tiene muchas parametrizaciones
  - Para cada instancia, cada parametrización tiene diferentes rendimientos
  - Para una instancia dada, ¿se puede aprender cuál es la mejor parametrización para resolverla?
  - Sí, usando ML!

Algoritmos Portfolio



Algoritmos de Autoconfiguración



DPLL

- **DPLL**: Davis-Putnam-Logemann-Loveland [26]

- Origen: algoritmo **DP** [27]
  - Resolver F equivalente a resolver $F' = F_{|x} \vee F_{|\neg x}$
  - F' no está en CNF
  - La traducción a CNF no es eficiente en espacio: *explosión en memoria!*
  - Mejor usar **búsqueda** $\rightsquigarrow$ **DPLL**

- Consiste en:
  - **Propagación unitaria** (**UP**)
  - **Búsqueda** en profundidad (con poda)

Propagación Unitaria (UP)

1. Itera para cada cláusula:
   - Si todos sus literales menos uno están asignados, y la cláusula no está satisfecha:
     - **UP**: Asigna el literal restante para satisfacer la cláusula
   - Si todos sus literales están asignados, y la cláusula no está satisfecha:
     - Devuelve `conflicto`
2. Repite (1) hasta que no haya asignaciones
3. Devuelve la fórmula resultante F' y la asignación (parcial) σ

UP: Ejemplo 1

$$F = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_2 \vee \neg x_3) \wedge (\neg x_3)$$

UP: Ejemplo 1

$$F = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_2 \vee \neg x_3) \wedge (\neg x_3)$$

$$F = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_2 \vee \neg x_3) \wedge (\neg x_3)$$
$$\sigma = \{ x_3 = 0 \}$$

UP: Ejemplo 1

$$F = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_2 \vee \neg x_3) \wedge (\neg x_3)$$

$$F = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_2 \vee \neg x_3) \wedge (\neg x_3)$$
$$\sigma = \{ x_3 = 0 \}$$

$$F = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_2 \vee \neg x_3) \wedge (\neg x_3)$$
$$\sigma = \{ x_3 = 0 \}$$

UP: Ejemplo 1

$$F = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_2 \vee \neg x_3) \wedge (\neg x_3)$$

$$F = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_2 \vee \neg x_3) \wedge (\neg x_3)$$
$$\sigma = \{ x_3 = 0 \}$$

$$F = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_2 \vee \neg x_3) \wedge (\neg x_3)$$
$$\sigma = \{ x_3 = 0 \}$$

$$F' = (x_1 \vee x_2) \wedge (\neg x_1 \vee x_2)$$
$$\sigma = \{ x_3 = 0 \}$$

UP: Ejemplo 2

$$F = (x_1 \vee x_2 \vee x_3 \vee x_4) \wedge (\neg x_1 \vee x_2 \vee \neg x_4) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$

UP: Ejemplo 2

$$F = (x_1 \vee x_2 \vee x_3 \vee x_4) \wedge (\neg x_1 \vee x_2 \vee \neg x_4) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$

$$F = (x_1 \vee x_2 \vee x_3 \vee x_4) \wedge (\neg x_1 \vee x_2 \vee \neg x_4) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$

$$\sigma = \{ x_3 = 0 \}$$

UP: Ejemplo 2

$$F = (x_1 \vee x_2 \vee x_3 \vee x_4) \wedge (\neg x_1 \vee x_2 \vee \neg x_4) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$

$$F = (x_1 \vee x_2 \vee x_3 \vee x_4) \wedge (\neg x_1 \vee x_2 \vee \neg x_4) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$
$$\sigma = \{ x_3 = 0 \}$$

$$F = (x_1 \vee x_2 \vee x_3 \vee x_4) \wedge (\neg x_1 \vee x_2 \vee \neg x_4) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$
$$\sigma = \{ x_3 = 0 , x_2 = 0 \}$$

UP: Ejemplo 2

$$F = (x_1 \vee x_2 \vee x_3 \vee x_4) \wedge (\neg x_1 \vee x_2 \vee \neg x_4) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$

$$F = (x_1 \vee x_2 \vee x_3 \vee x_4) \wedge (\neg x_1 \vee x_2 \vee \neg x_4) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$
$$\sigma = \{ x_3 = 0 \}$$

$$F = (x_1 \vee x_2 \vee x_3 \vee x_4) \wedge (\neg x_1 \vee x_2 \vee \neg x_4) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$
$$\sigma = \{ x_3 = 0, x_2 = 0 \}$$

$$F = (x_1 \vee x_2 \vee x_3 \vee x_4) \wedge (\neg x_1 \vee x_2 \vee \neg x_4) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$
$$\sigma = \{ x_3 = 0, x_2 = 0 \}$$

UP: Ejemplo 2

$$F = (x_1 \vee x_2 \vee x_3 \vee x_4) \wedge (\neg x_1 \vee x_2 \vee \neg x_4) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$

$$F = (x_1 \vee x_2 \vee x_3 \vee x_4) \wedge (\neg x_1 \vee x_2 \vee \neg x_4) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$
$$\sigma = \{ x_3 = 0 \}$$

$$F = (x_1 \vee x_2 \vee x_3 \vee x_4) \wedge (\neg x_1 \vee x_2 \vee \neg x_4) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$
$$\sigma = \{ x_3 = 0, x_2 = 0 \}$$

$$F = (x_1 \vee x_2 \vee x_3 \vee x_4) \wedge (\neg x_1 \vee x_2 \vee \neg x_4) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$
$$\sigma = \{ x_3 = 0, x_2 = 0 \}$$

$$F' = (x_1 \vee x_4) \wedge (\neg x_1 \vee \neg x_4)$$
$$\sigma = \{ x_3 = 0, x_2 = 0 \}$$

UP: Ejemplo 3

$$F = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$

UP: Ejemplo 3

$$F = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$

$$F = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$
$$\sigma = \{ x_3 = 0 \}$$

UP: Ejemplo 3

$$F = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$

$$F = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$
$$\sigma = \{ x_3 = 0 \}$$

$$F = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$
$$\sigma = \{ x_3 = 0 , x_2 = 0 \}$$

UP: Ejemplo 3

$$F = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$

$$F = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$
$$\sigma = \{ x_3 = 0 \}$$

$$F = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$
$$\sigma = \{ x_3 = 0, x_2 = 0 \}$$

$$F = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$
$$\sigma = \{ x_3 = 0, x_2 = 0, x_1 = 1 \}$$

UP: Ejemplo 3

$$F = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$

$$F = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$
$$\sigma = \{ x_3 = 0 \}$$

$$F = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$
$$\sigma = \{ x_3 = 0, x_2 = 0 \}$$

$$F = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$
$$\sigma = \{ x_3 = 0, x_2 = 0, x_1 = 1 \}$$

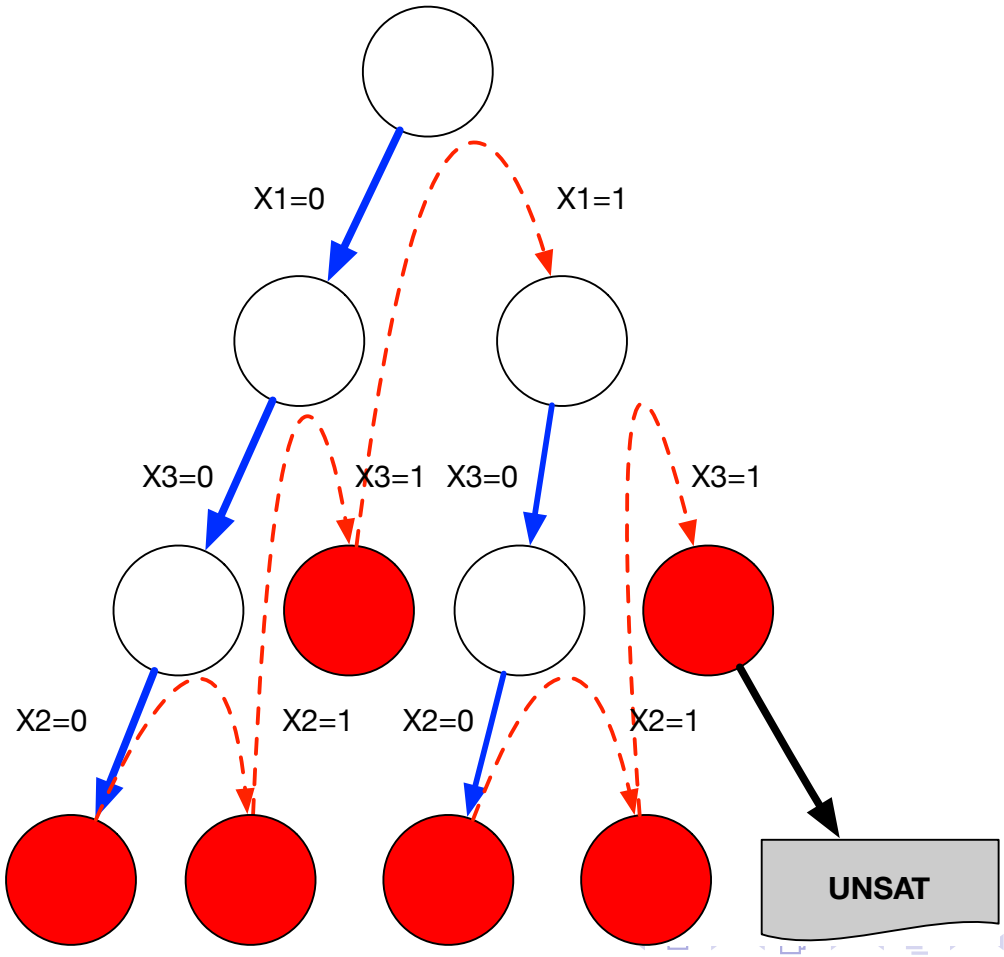
$$F = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3)$$
$$\sigma = \{ x_3 = 0, x_2 = 0, x_1 = 1 \}$$
`conflicto!`

Búsqueda en profundidad

1. Selecciona una variable (heurística)
2. Selecciona una polaridad (heurística)
3. Computa la fórmula resultante tras asignarla
4. Si todas las cláusulas satisfechas, devuelve SAT
5. Si todas las asignaciones exploradas, devuelve UNSAT
6. Si no, repite (1)

Búsqueda en profundidad: Ejemplo

$$F = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2) \wedge (\neg x_1 \vee \neg x_3) \wedge (\neg x_2 \vee \neg x_3) \wedge (x_1 \vee \neg x_2) \wedge (x_2 \vee \neg x_3) \wedge (x_1 \vee \neg x_3) \wedge (\neg x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_3)$$



Heurística y Decisiones

- **Heurística**:
  - ¿Qué variable escoger? ¿Con qué polaridad?
- **Decisión** (*branching variable*):
  literal elegido por la heurística
- **Propagación**: literal asignado en UP
- **Búsqueda + UP**: tras cada decisión, realiza UP
  - También antes de la primera decisión
- **Nivel de decisión** (*dl*): conjunto de literales asignados entre dos decisiones
  - Contiene una decisión y todos los literales propagados
  - En $dl = 0$ sólo hay literales propagados
- **Notación**: <var> $\overset{\{d,UP\}}{=}$ <value> @ <dl>
  - *Ejemplo*: $\{x_1 \overset{UP}{=} 1@0\,,\ x_2 \overset{d}{=} 0@1\,,\ \ldots\}$

Look-ahead SAT solvers [39]

- El **núcleo** es una implementación de **DPLL**
- La mayor **diferencia** está en la **heurística**
- Para elegir la siguiente decisión entre un conjunto de variables, se puede analizar los efectos de elegir cada una de ellas
- En concreto, ¿cómo quedaría la fórmula tras decidir en cada variable?
- Incluso se puede explorar la simulación de varias decisiones...

Conflict Driven Clause Learning (CDCL) [62]

- Basado en **DPLL** [26]: Búsqueda + **UP**
- **Aprendizaje de cláusulas** [63] - - - - - - - - - - → **CL**
- Backtracking no cronológico: **backjumping** [63]
- Heurísticas de actividad: **VSIDS** [52] - - - - - - → **CD**
- Políticas de borrado de cláusulas: **LBD** [13]
- **Restarts** [37]
- Preprocessing [29] e Inprocessing [43]
- Estructura de datos "ligera": two-watched literals [52]

Análisis de conflictos [63]

$$F = \omega_1 \wedge \omega_2 \wedge \omega_3 \wedge \omega_4 \wedge \omega_5 \wedge \omega_6 \wedge \cdots =$$
$$= (x_1 \vee x_7 \vee \neg x_2) \wedge (x_1 \vee \neg x_3) \wedge (x_2 \vee x_3 \vee x_4)$$
$$\wedge (\neg x_4 \vee \neg x_5) \wedge (x_8 \vee \neg x_4 \vee \neg x_6) \wedge (x_5 \vee x_6) \wedge \ldots$$

$$\sigma = \{\}$$

	Conf	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	...
Reason										

Análisis de conflictos [63]

$$F = \omega_1 \wedge \omega_2 \wedge \omega_3 \wedge \omega_4 \wedge \omega_5 \wedge \omega_6 \wedge \cdots =$$
$$= (x_1 \vee x_7 \vee \neg x_2) \wedge (x_1 \vee \neg x_3) \wedge (x_2 \vee x_3 \vee x_4)$$
$$\wedge (\neg x_4 \vee \neg x_5) \wedge (x_8 \vee \neg x_4 \vee \neg x_6) \wedge (x_5 \vee x_6) \wedge \ldots$$

$$\sigma = \{x_8 \stackrel{d}{=} 0@2\}$$

$x_i \stackrel{d}{=} k@1$ $\quad$ $x_8 \stackrel{d}{=} 0@2$

	Conf	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	$\ldots$
Reason									-	

Análisis de conflictos [63]

$$F = \omega_1 \wedge \omega_2 \wedge \omega_3 \wedge \omega_4 \wedge \omega_5 \wedge \omega_6 \wedge \cdots =$$
$$= (x_1 \vee x_7 \vee \neg x_2) \wedge (x_1 \vee \neg x_3) \wedge (x_2 \vee x_3 \vee x_4)$$
$$\wedge (\neg x_4 \vee \neg x_5) \wedge (x_8 \vee \neg x_4 \vee \neg x_6) \wedge (x_5 \vee x_6) \wedge \ldots$$

$$\sigma = \{x_8 \stackrel{d}{=} 0@2 \, , \, x_7 \stackrel{d}{=} 0@3\}$$

$x_7 \stackrel{d}{=} 0@3$

$x_i \stackrel{d}{=} k@1$

$x_8 \stackrel{d}{=} 0@2$

	Conf	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	$\ldots$
Reason								-	-	

Análisis de conflictos [63]

$$F = \omega_1 \wedge \omega_2 \wedge \omega_3 \wedge \omega_4 \wedge \omega_5 \wedge \omega_6 \wedge \cdots =$$
$$= (x_1 \vee x_7 \vee \neg x_2) \wedge (x_1 \vee \neg x_3) \wedge (x_2 \vee x_3 \vee x_4)$$
$$\wedge (\neg x_4 \vee \neg x_5) \wedge (x_8 \vee \neg x_4 \vee \neg x_6) \wedge (x_5 \vee x_6) \wedge \ldots$$

$$\sigma = \left\{ x_8 \stackrel{d}{=} 0@2 \, , \, x_7 \stackrel{d}{=} 0@3 \, , \, x_1 \stackrel{d}{=} 0@5 \right\}$$

$x_7 \stackrel{d}{=} 0@3$

$x_1 \stackrel{d}{=} 0@5$

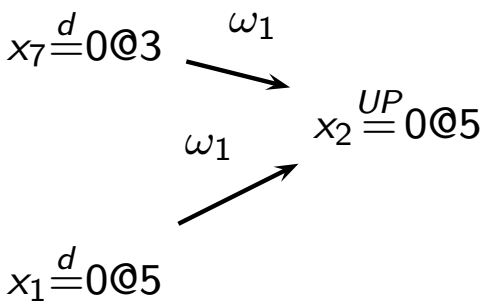
$x_i \stackrel{d}{=} k@1$ $\quad$ $x_j \stackrel{d}{=} k'@4$ $\quad$ $x_8 \stackrel{d}{=} 0@2$

	Conf	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	$\ldots$
Reason		-						-	-	

Análisis de conflictos [63]

$$F = \omega_1 \wedge \omega_2 \wedge \omega_3 \wedge \omega_4 \wedge \omega_5 \wedge \omega_6 \wedge \cdots =$$
$$= (x_1 \vee x_7 \vee \neg x_2) \wedge (x_1 \vee \neg x_3) \wedge (x_2 \vee x_3 \vee x_4)$$
$$\wedge (\neg x_4 \vee \neg x_5) \wedge (x_8 \vee \neg x_4 \vee \neg x_6) \wedge (x_5 \vee x_6) \wedge \ldots$$

$$\sigma = \left\{ x_8 \overset{d}{=} 0@2 \, , \, x_7 \overset{d}{=} 0@3 \, , \, x_1 \overset{d}{=} 0@5 \, , \, x_2 \overset{UP}{=} 0@5 \right\}$$



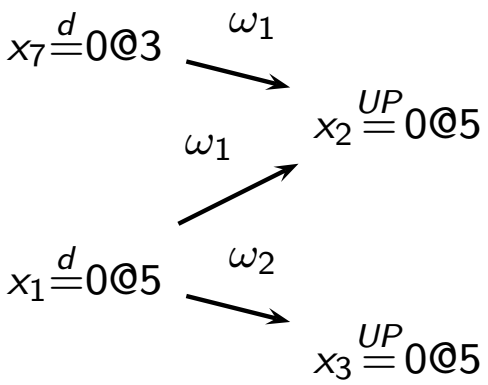
$x_i \overset{d}{=} k@1$ $\quad$ $x_j \overset{d}{=} k'@4$ $\quad$ $x_8 \overset{d}{=} 0@2$

	Conf	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	...
Reason		-	ω_1					-	-	

Análisis de conflictos [63]

$$F = \omega_1 \wedge \omega_2 \wedge \omega_3 \wedge \omega_4 \wedge \omega_5 \wedge \omega_6 \wedge \cdots =$$
$$= (x_1 \vee x_7 \vee \neg x_2) \wedge (x_1 \vee \neg x_3) \wedge (x_2 \vee x_3 \vee x_4)$$
$$\wedge (\neg x_4 \vee \neg x_5) \wedge (x_8 \vee \neg x_4 \vee \neg x_6) \wedge (x_5 \vee x_6) \wedge \ldots$$

$$\sigma = \left\{ x_8 \overset{d}{=} 0@2 \, , \, x_7 \overset{d}{=} 0@3 \, , \, x_1 \overset{d}{=} 0@5 \, , \, x_2 \overset{UP}{=} 0@5 \, , \, x_3 \overset{UP}{=} 0@5 \right\}$$



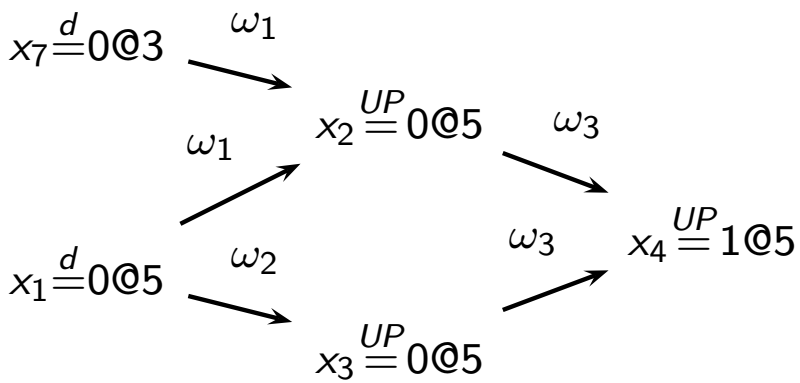
$x_i \overset{d}{=} k@1$ $\quad$ $x_j \overset{d}{=} k'@4$ $\quad$ $x_8 \overset{d}{=} 0@2$

	Conf	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	...
Reason		-	ω_1	ω_2				-	-	

Análisis de conflictos [63]

$$F = \omega_1 \wedge \omega_2 \wedge \omega_3 \wedge \omega_4 \wedge \omega_5 \wedge \omega_6 \wedge \cdots =$$
$$= (x_1 \vee x_7 \vee \neg x_2) \wedge (x_1 \vee \neg x_3) \wedge (x_2 \vee x_3 \vee x_4)$$
$$\wedge (\neg x_4 \vee \neg x_5) \wedge (x_8 \vee \neg x_4 \vee \neg x_6) \wedge (x_5 \vee x_6) \wedge \ldots$$

$$\sigma = \left\{ x_8 \overset{d}{=} 0@2 \, , \, x_7 \overset{d}{=} 0@3 \, , \, x_1 \overset{d}{=} 0@5 \, , \, x_2 \overset{UP}{=} 0@5 \, , \, x_3 \overset{UP}{=} 0@5 \, , \, x_4 \overset{UP}{=} 1@5 \right\}$$



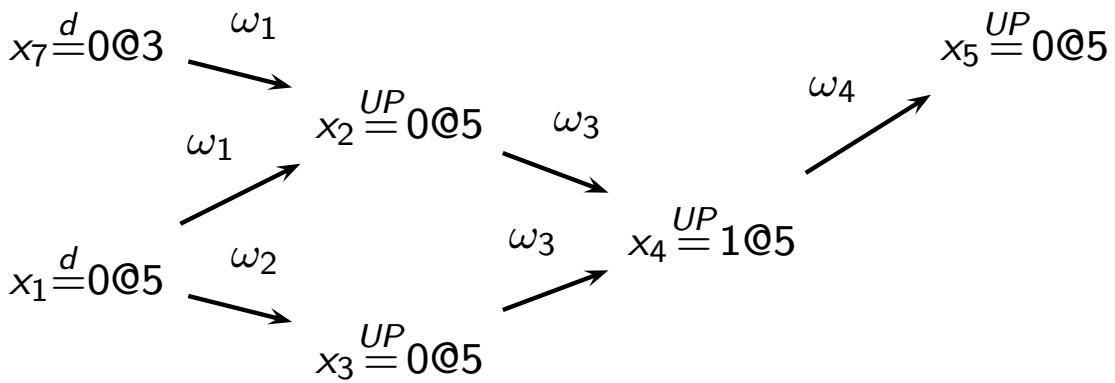
$x_i \overset{d}{=} k@1$ $\quad$ $x_j \overset{d}{=} k'@4$ $\quad$ $x_8 \overset{d}{=} 0@2$

	Conf	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	$\ldots$
Reason		-	ω_1	ω_2	ω_3			-	-	

Análisis de conflictos [63]

$$F = \omega_1 \wedge \omega_2 \wedge \omega_3 \wedge \omega_4 \wedge \omega_5 \wedge \omega_6 \wedge \cdots =$$
$$= (x_1 \vee x_7 \vee \neg x_2) \wedge (x_1 \vee \neg x_3) \wedge (x_2 \vee x_3 \vee x_4)$$
$$\wedge (\neg x_4 \vee \neg x_5) \wedge (x_8 \vee \neg x_4 \vee \neg x_6) \wedge (x_5 \vee x_6) \wedge \ldots$$

$$\sigma = \left\{ x_8 \overset{d}{=} 0@2 \, , \, x_7 \overset{d}{=} 0@3 \, , \, x_1 \overset{d}{=} 0@5 \, , \, x_2 \overset{UP}{=} 0@5 \, , \, x_3 \overset{UP}{=} 0@5 \, , \, x_4 \overset{UP}{=} 1@5 \, , \, x_5 \overset{UP}{=} 0@5 \right\}$$



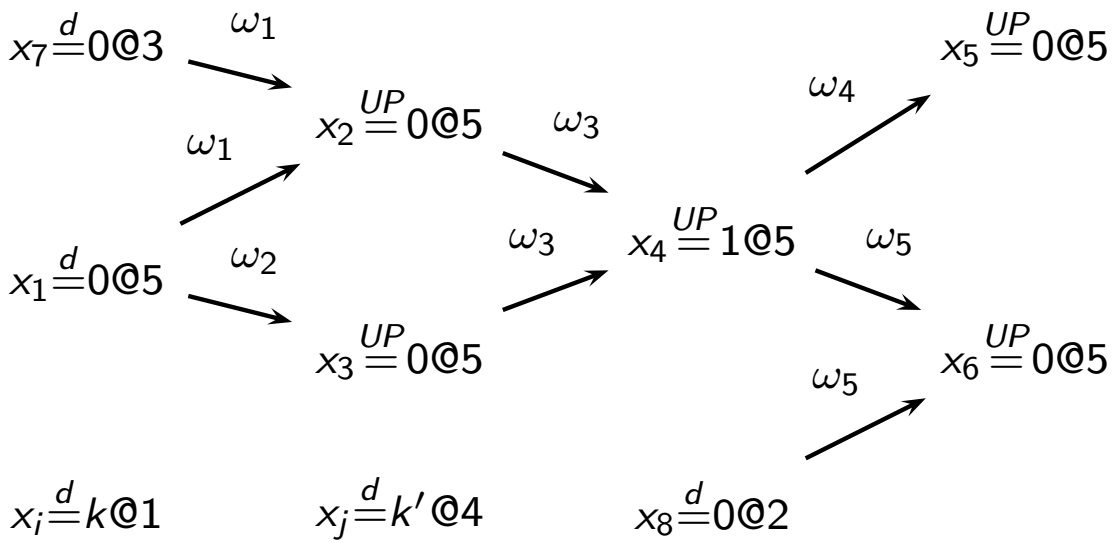
$x_i \overset{d}{=} k@1$ $\quad$ $x_j \overset{d}{=} k'@4$ $\quad$ $x_8 \overset{d}{=} 0@2$

	Conf	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	...
Reason		-	ω_1	ω_2	ω_3	ω_4		-	-	

Análisis de conflictos [63]

$$F = \omega_1 \wedge \omega_2 \wedge \omega_3 \wedge \omega_4 \wedge \omega_5 \wedge \omega_6 \wedge \cdots =$$
$$= (x_1 \vee x_7 \vee \neg x_2) \wedge (x_1 \vee \neg x_3) \wedge (x_2 \vee x_3 \vee x_4)$$
$$\wedge (\neg x_4 \vee \neg x_5) \wedge (x_8 \vee \neg x_4 \vee \neg x_6) \wedge (x_5 \vee x_6) \wedge \ldots$$

$$\sigma = \left\{ x_8 \overset{d}{=} 0@2 \, , \, x_7 \overset{d}{=} 0@3 \, , \, x_1 \overset{d}{=} 0@5 \, , \, x_2 \overset{UP}{=} 0@5 \, , \, x_3 \overset{UP}{=} 0@5 \, , \, x_4 \overset{UP}{=} 1@5 \, , \, x_5 \overset{UP}{=} 0@5 \, , \, x_6 \overset{UP}{=} 0@5 \right\}$$

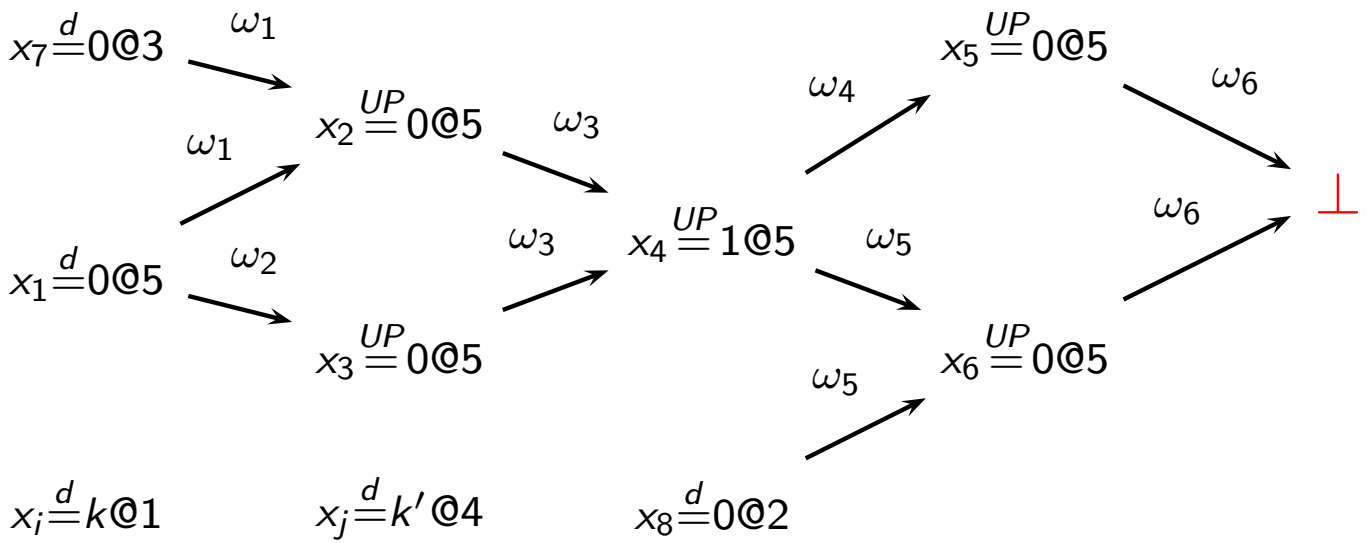


	Conf	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	...
Reason		-	ω_1	ω_2	ω_3	ω_4	ω_5	-	-	

Análisis de conflictos [63]

$$F = \omega_1 \wedge \omega_2 \wedge \omega_3 \wedge \omega_4 \wedge \omega_5 \wedge \omega_6 \wedge \cdots =$$
$$= (x_1 \vee x_7 \vee \neg x_2) \wedge (x_1 \vee \neg x_3) \wedge (x_2 \vee x_3 \vee x_4)$$
$$\wedge (\neg x_4 \vee \neg x_5) \wedge (x_8 \vee \neg x_4 \vee \neg x_6) \wedge (x_5 \vee x_6) \wedge \ldots$$

$$\sigma = \left\{ x_8 \overset{d}{=} 0@2 \, , \, x_7 \overset{d}{=} 0@3 \, , \, x_1 \overset{d}{=} 0@5 \, , \, x_2 \overset{UP}{=} 0@5 \, , \, x_3 \overset{UP}{=} 0@5 \, , \, x_4 \overset{UP}{=} 1@5 \, , \, x_5 \overset{UP}{=} 0@5 \, , \, x_6 \overset{UP}{=} 0@5 \right\}$$



	Conf	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	...
Reason	ω_6	-	ω_1	ω_2	ω_3	ω_4	ω_5	-	-	

Grafo de implicaciones del conflicto

- Se construye **a partir del conflicto hacia atrás**:
  1. Comenzar con una pila/lista vacía
  2. Apilar literales de la cláusula en conflicto
  3. Mientras la pila no esté vacía:
     - 3.1 Desapilar literal
     - 3.2 Si se asignó por UP, apilar literales de su *reason*
     - 3.3 (Si se asignó por decisión, `continue`)

- Cada vez que (3.2): resolución. Por eso **CDCL implementa resolución** (resolvente: consecuencia lógica de la fórmula)
- Si se conserva (se **aprende**), se evita repetir el mismo conflicto $\rightsquigarrow$ pruning!
- Pero muchas posibles cláusulas aprendidas en cada conflicto $\rightsquigarrow$ explosión en memoria
  Entonces aprender sólo 1. ¿Cuál?
- **1-UIP** [63]: primera cláusula que contiene **un único literal del nivel de decisión del conflicto**

Grafo de implicaciones del conflicto

- Se construye **a partir del conflicto hacia atrás**:
  1. Comenzar con una pila/lista vacía
  2. Apilar literales de la cláusula en conflicto
  3. Mientras la pila no esté vacía:
     - 3.1 Desapilar literal
     - 3.2 Si se asignó por UP, apilar literales de su *reason*
     - 3.3 (Si se asignó por decisión, `continue`)

- Cada vez que (3.2): resolución. Por eso **CDCL implementa resolución** (resolvente: consecuencia lógica de la fórmula)
- Si se conserva (se **aprende**), se evita repetir el mismo conflicto $\rightsquigarrow$ pruning!
- Pero muchas posibles cláusulas aprendidas en cada conflicto $\rightsquigarrow$ explosión en memoria
  Entonces aprender sólo 1. ¿Cuál?
- **1-UIP** [63]: primera cláusula que contiene **un único literal del nivel de decisión del conflicto**

Aprendizaje de cláusulas [63]

$$F = \omega_1 \wedge \omega_2 \wedge \omega_3 \wedge \omega_4 \wedge \omega_5 \wedge \omega_6 \wedge \cdots =$$
$$= (x_1 \vee x_7 \vee \neg x_2) \wedge (x_1 \vee \neg x_3) \wedge (x_2 \vee x_3 \vee x_4)$$
$$\wedge (\neg x_4 \vee \neg x_5) \wedge (x_8 \vee \neg x_4 \vee \neg x_6) \wedge (x_5 \vee x_6) \wedge \ldots$$

$$\sigma = \left\{ x_8 \overset{d}{=} 0@2 \, , \, x_7 \overset{d}{=} 0@3 \, , \, x_1 \overset{d}{=} 0@5 \, , \, x_2 \overset{UP}{=} 0@5 \, , \, x_3 \overset{UP}{=} 0@5 \, , \, x_4 \overset{UP}{=} 1@5 \, , \, x_5 \overset{UP}{=} 0@5 \, , \, x_6 \overset{UP}{=} 0@5 \right\}$$

	Conf	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	$\ldots$
Reason	ω_6	-	ω_1	ω_2	ω_3	ω_4	ω_5	-	-	

Var.	Rea.	Cl. Aprendida	$\#lit_{(dl=5)}$	
-	-	$\omega_6 = (x_5 \vee x_6)$	2	Conflicto
x_5	ω_4	$(x_6 \vee \neg x_4)$	2	
x_6	ω_5	$(\neg x_4 \vee x_8)$	1	
x_4	ω_3	$(x_8 \vee x_2 \vee x_3)$	2	
x_2	ω_1	$(x_8 \vee x_3 \vee x_1 \vee x_7)$	2	
x_3	ω_2	$(x_8 \vee x_1 \vee x_7)$	1	

Aprendizaje de cláusulas [63]

$$F = \omega_1 \wedge \omega_2 \wedge \omega_3 \wedge \omega_4 \wedge \omega_5 \wedge \omega_6 \wedge \cdots =$$
$$= (x_1 \vee x_7 \vee \neg x_2) \wedge (x_1 \vee \neg x_3) \wedge (x_2 \vee x_3 \vee x_4)$$
$$\wedge (\neg x_4 \vee \neg x_5) \wedge (x_8 \vee \neg x_4 \vee \neg x_6) \wedge (x_5 \vee x_6) \wedge \ldots$$

$$\sigma = \left\{ x_8 \overset{d}{=} 0@2 \, , \, x_7 \overset{d}{=} 0@3 \, , \, x_1 \overset{d}{=} 0@5 \, , \, x_2 \overset{UP}{=} 0@5 \, , \, x_3 \overset{UP}{=} 0@5 \, , \, x_4 \overset{UP}{=} 1@5 \, , \, x_5 \overset{UP}{=} 0@5 \, , \, x_6 \overset{UP}{=} 0@5 \right\}$$

	Conf	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	$\ldots$
Reason	ω_6	-	ω_1	ω_2	ω_3	ω_4	ω_5	-	-	

Var.	Rea.	Cl. Aprendida	$\#lit_{(dl=5)}$	
-	-	$\omega_6 = (x_5 \vee x_6)$	2	Conflicto
x_5	ω_4	$(x_6 \vee \neg x_4)$	2	
x_6	ω_5	$(\neg x_4 \vee x_8)$	1	
x_4	ω_3	$(x_8 \vee x_2 \vee x_3)$	2	
x_2	ω_1	$(x_8 \vee x_3 \vee x_1 \vee x_7)$	2	
x_3	ω_2	$(x_8 \vee x_1 \vee x_7)$	1	

Aprendizaje de cláusulas [63]

$$F = \omega_1 \wedge \omega_2 \wedge \omega_3 \wedge \omega_4 \wedge \omega_5 \wedge \omega_6 \wedge \cdots =$$
$$= (x_1 \vee x_7 \vee \neg x_2) \wedge (x_1 \vee \neg x_3) \wedge (x_2 \vee x_3 \vee x_4)$$
$$\wedge (\neg x_4 \vee \neg x_5) \wedge (x_8 \vee \neg x_4 \vee \neg x_6) \wedge (x_5 \vee x_6) \wedge \ldots$$

$$\sigma = \left\{ x_8 \overset{d}{=} 0@2 \, , \, x_7 \overset{d}{=} 0@3 \, , \, x_1 \overset{d}{=} 0@5 \, , \, x_2 \overset{UP}{=} 0@5 \, , \, x_3 \overset{UP}{=} 0@5 \, , \, x_4 \overset{UP}{=} 1@5 \, , \, x_5 \overset{UP}{=} 0@5 \, , \, x_6 \overset{UP}{=} 0@5 \right\}$$

	Conf	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	$\ldots$
Reason	ω_6	-	ω_1	ω_2	ω_3	ω_4	ω_5	-	-	

Var.	Rea.	Cl. Aprendida	$\#lit_{(dl=5)}$	
-	-	$\omega_6 = (x_5 \vee x_6)$	2	Conflicto
x_5	ω_4	$(x_6 \vee \neg x_4)$	2	
x_6	ω_5	$(\neg x_4 \vee x_8)$	1	
x_4	ω_3	$(x_8 \vee x_2 \vee x_3)$	2	
x_2	ω_1	$(x_8 \vee x_3 \vee x_1 \vee x_7)$	2	
x_3	ω_2	$(x_8 \vee x_1 \vee x_7)$	1	

Aprendizaje de cláusulas [63]

$$F = \omega_1 \wedge \omega_2 \wedge \omega_3 \wedge \omega_4 \wedge \omega_5 \wedge \omega_6 \wedge \cdots =$$
$$= (x_1 \vee x_7 \vee \neg x_2) \wedge (x_1 \vee \neg x_3) \wedge (x_2 \vee x_3 \vee x_4)$$
$$\wedge (\neg x_4 \vee \neg x_5) \wedge (x_8 \vee \neg x_4 \vee \neg x_6) \wedge (x_5 \vee x_6) \wedge \ldots$$

$$\sigma = \left\{ x_8 \overset{d}{=} 0@2 \, , \, x_7 \overset{d}{=} 0@3 \, , \, x_1 \overset{d}{=} 0@5 \, , \, x_2 \overset{UP}{=} 0@5 \, , \, x_3 \overset{UP}{=} 0@5 \, , \, x_4 \overset{UP}{=} 1@5 \, , \, x_5 \overset{UP}{=} 0@5 \, , \, x_6 \overset{UP}{=} 0@5 \right\}$$

	Conf	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	$\ldots$
Reason	ω_6	-	ω_1	ω_2	ω_3	ω_4	ω_5	-	-	

Var.	Rea.	Cl. Aprendida	$\#lit_{(dl=5)}$	
-	-	$\omega_6 = (x_5 \vee x_6)$	2	Conflicto
x_5	ω_4	$(x_6 \vee \neg x_4)$	2	
x_6	ω_5	$(\neg x_4 \vee x_8)$	1	
x_4	ω_3	$(x_8 \vee x_2 \vee x_3)$	2	
x_2	ω_1	$(x_8 \vee x_3 \vee x_1 \vee x_7)$	2	
x_3	ω_2	$(x_8 \vee x_1 \vee x_7)$	1	

Aprendizaje de cláusulas [63]

$$F = \omega_1 \wedge \omega_2 \wedge \omega_3 \wedge \omega_4 \wedge \omega_5 \wedge \omega_6 \wedge \cdots =$$
$$= (x_1 \vee x_7 \vee \neg x_2) \wedge (x_1 \vee \neg x_3) \wedge (x_2 \vee x_3 \vee x_4)$$
$$\wedge (\neg x_4 \vee \neg x_5) \wedge (x_8 \vee \neg x_4 \vee \neg x_6) \wedge (x_5 \vee x_6) \wedge \ldots$$

$$\sigma = \left\{ x_8 \overset{d}{=} 0@2 \, , \, x_7 \overset{d}{=} 0@3 \, , \, x_1 \overset{d}{=} 0@5 \, , \, x_2 \overset{UP}{=} 0@5 \, , \, x_3 \overset{UP}{=} 0@5 \, , \, x_4 \overset{UP}{=} 1@5 \, , \, x_5 \overset{UP}{=} 0@5 \, , \, x_6 \overset{UP}{=} 0@5 \right\}$$

	Conf	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	$\ldots$
Reason	ω_6	-	ω_1	ω_2	ω_3	ω_4	ω_5	-	-	

Var.	Rea.	Cl. Aprendida	$\#lit_{(dl=5)}$	
-	-	$\omega_6 = (x_5 \vee x_6)$	2	Conflicto
x_5	ω_4	$(x_6 \vee \neg x_4)$	2	
x_6	ω_5	$(\neg x_4 \vee x_8)$	1	**1-UIP [63]**
x_4	ω_3	$(x_8 \vee x_2 \vee x_3)$	2	
x_2	ω_1	$(x_8 \vee x_3 \vee x_1 \vee x_7)$	2	
x_3	ω_2	$(x_8 \vee x_1 \vee x_7)$	1	

Aprendizaje de cláusulas [63]

$$F = \omega_1 \wedge \omega_2 \wedge \omega_3 \wedge \omega_4 \wedge \omega_5 \wedge \omega_6 \wedge \cdots =$$
$$= (x_1 \vee x_7 \vee \neg x_2) \wedge (x_1 \vee \neg x_3) \wedge (x_2 \vee x_3 \vee x_4)$$
$$\wedge (\neg x_4 \vee \neg x_5) \wedge (x_8 \vee \neg x_4 \vee \neg x_6) \wedge (x_5 \vee x_6) \wedge \ldots$$

$$\sigma = \left\{ x_8 \overset{d}{=} 0@2 \, , \, x_7 \overset{d}{=} 0@3 \, , \, x_1 \overset{d}{=} 0@5 \, , \, x_2 \overset{UP}{=} 0@5 \, , \, x_3 \overset{UP}{=} 0@5 \, , \, x_4 \overset{UP}{=} 1@5 \, , \, x_5 \overset{UP}{=} 0@5 \, , \, x_6 \overset{UP}{=} 0@5 \right\}$$

	Conf	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	$\ldots$
Reason	ω_6	-	ω_1	ω_2	ω_3	ω_4	ω_5	-	-	

Var.	Rea.	Cl. Aprendida	$\#lit_{(dl=5)}$	
-	-	$\omega_6 = (x_5 \vee x_6)$	2	Conflicto
x_5	ω_4	$(x_6 \vee \neg x_4)$	2	
x_6	ω_5	$(\neg x_4 \vee x_8)$	1	**1-UIP [63]**
x_4	ω_3	$(x_8 \vee x_2 \vee x_3)$	2	
x_2	ω_1	$(x_8 \vee x_3 \vee x_1 \vee x_7)$	2	
x_3	ω_2	$(x_8 \vee x_1 \vee x_7)$	1	

Aprendizaje de cláusulas [63]

$$F = \omega_1 \wedge \omega_2 \wedge \omega_3 \wedge \omega_4 \wedge \omega_5 \wedge \omega_6 \wedge \cdots =$$
$$= (x_1 \vee x_7 \vee \neg x_2) \wedge (x_1 \vee \neg x_3) \wedge (x_2 \vee x_3 \vee x_4)$$
$$\wedge (\neg x_4 \vee \neg x_5) \wedge (x_8 \vee \neg x_4 \vee \neg x_6) \wedge (x_5 \vee x_6) \wedge \ldots$$

$$\sigma = \left\{ x_8 \overset{d}{=} 0@2 \, , \, x_7 \overset{d}{=} 0@3 \, , \, x_1 \overset{d}{=} 0@5 \, , \, x_2 \overset{UP}{=} 0@5 \, , \, x_3 \overset{UP}{=} 0@5 \, , \, x_4 \overset{UP}{=} 1@5 \, , \, x_5 \overset{UP}{=} 0@5 \, , \, x_6 \overset{UP}{=} 0@5 \right\}$$

	Conf	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	$\ldots$
Reason	ω_6	-	ω_1	ω_2	ω_3	ω_4	ω_5	-	-	

Var.	Rea.	Cl. Aprendida	$\#lit_{(dl=5)}$	
-	-	$\omega_6 = (x_5 \vee x_6)$	2	Conflicto
x_5	ω_4	$(x_6 \vee \neg x_4)$	2	
x_6	ω_5	$(\neg x_4 \vee x_8)$	1	**1-UIP [63]**
x_4	ω_3	$(x_8 \vee x_2 \vee x_3)$	2	
x_2	ω_1	$(x_8 \vee x_3 \vee x_1 \vee x_7)$	2	
x_3	ω_2	$(x_8 \vee x_1 \vee x_7)$	1	

Aprendizaje de cláusulas [63]

$$F = \omega_1 \wedge \omega_2 \wedge \omega_3 \wedge \omega_4 \wedge \omega_5 \wedge \omega_6 \wedge \cdots =$$
$$= (x_1 \vee x_7 \vee \neg x_2) \wedge (x_1 \vee \neg x_3) \wedge (x_2 \vee x_3 \vee x_4)$$
$$\wedge (\neg x_4 \vee \neg x_5) \wedge (x_8 \vee \neg x_4 \vee \neg x_6) \wedge (x_5 \vee x_6) \wedge \ldots$$

$$\sigma = \left\{ x_8 \overset{d}{=} 0@2 \, , \, x_7 \overset{d}{=} 0@3 \, , \, x_1 \overset{d}{=} 0@5 \, , \, x_2 \overset{UP}{=} 0@5 \, , \, x_3 \overset{UP}{=} 0@5 \, , \, x_4 \overset{UP}{=} 1@5 \, , \, x_5 \overset{UP}{=} 0@5 \, , \, x_6 \overset{UP}{=} 0@5 \right\}$$

	Conf	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	$\ldots$
Reason	ω_6	-	ω_1	ω_2	ω_3	ω_4	ω_5	-	-	

Var.	Rea.	Cl. Aprendida	$\#lit_{(dl=5)}$	
-	-	$\omega_6 = (x_5 \vee x_6)$	2	Conflicto
x_5	ω_4	$(x_6 \vee \neg x_4)$	2	
x_6	ω_5	$(\neg x_4 \vee x_8)$	1	**1-UIP [63]**
x_4	ω_3	$(x_8 \vee x_2 \vee x_3)$	2	
x_2	ω_1	$(x_8 \vee x_3 \vee x_1 \vee x_7)$	2	
x_3	ω_2	$(x_8 \vee x_1 \vee x_7)$	1	

Aprendizaje de cláusulas [63]

$$F = \omega_1 \wedge \omega_2 \wedge \omega_3 \wedge \omega_4 \wedge \omega_5 \wedge \omega_6 \wedge \cdots =$$
$$= (x_1 \vee x_7 \vee \neg x_2) \wedge (x_1 \vee \neg x_3) \wedge (x_2 \vee x_3 \vee x_4)$$
$$\wedge (\neg x_4 \vee \neg x_5) \wedge (x_8 \vee \neg x_4 \vee \neg x_6) \wedge (x_5 \vee x_6) \wedge \ldots$$

$$\sigma = \left\{ x_8 \overset{d}{=} 0@2 \, , \, x_7 \overset{d}{=} 0@3 \, , \, x_1 \overset{d}{=} 0@5 \, , \, x_2 \overset{UP}{=} 0@5 \, , \, x_3 \overset{UP}{=} 0@5 \, , \, x_4 \overset{UP}{=} 1@5 \, , \, x_5 \overset{UP}{=} 0@5 \, , \, x_6 \overset{UP}{=} 0@5 \right\}$$

	Conf	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	$\ldots$
Reason	ω_6	-	ω_1	ω_2	ω_3	ω_4	ω_5	-	-	

Var.	Rea.	Cl. Aprendida	$\#lit_{(dl=5)}$	
-	-	$\omega_6 = (x_5 \vee x_6)$	2	Conflicto
x_5	ω_4	$(x_6 \vee \neg x_4)$	2	
x_6	ω_5	$(\neg x_4 \vee x_8)$	1	**1-UIP [63]**
x_4	ω_3	$(x_8 \vee x_2 \vee x_3)$	2	
x_2	ω_1	$(x_8 \vee x_3 \vee x_1 \vee x_7)$	2	
x_3	ω_2	$(x_8 \vee x_1 \vee x_7)$	1	

Aprendizaje de cláusulas [63]

$$F = \omega_1 \wedge \omega_2 \wedge \omega_3 \wedge \omega_4 \wedge \omega_5 \wedge \omega_6 \wedge \cdots =$$
$$= (x_1 \vee x_7 \vee \neg x_2) \wedge (x_1 \vee \neg x_3) \wedge (x_2 \vee x_3 \vee x_4)$$
$$\wedge (\neg x_4 \vee \neg x_5) \wedge (x_8 \vee \neg x_4 \vee \neg x_6) \wedge (x_5 \vee x_6) \wedge \ldots$$

$$\sigma = \left\{ x_8 \overset{d}{=} 0@2 \, , \, x_7 \overset{d}{=} 0@3 \, , \, x_1 \overset{d}{=} 0@5 \, , \, x_2 \overset{UP}{=} 0@5 \, , \, x_3 \overset{UP}{=} 0@5 \, , \, x_4 \overset{UP}{=} 1@5 \, , \, x_5 \overset{UP}{=} 0@5 \, , \, x_6 \overset{UP}{=} 0@5 \right\}$$

	Conf	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	$\ldots$
Reason	ω_6	-	ω_1	ω_2	ω_3	ω_4	ω_5	-	-	

Var.	Rea.	Cl. Aprendida	$\#lit_{(dl=5)}$	
-	-	$\omega_6 = (x_5 \vee x_6)$	2	Conflicto
x_5	ω_4	$(x_6 \vee \neg x_4)$	2	
x_6	ω_5	$(\neg x_4 \vee x_8)$	1	**1-UIP [63]**
x_4	ω_3	$(x_8 \vee x_2 \vee x_3)$	2	
x_2	ω_1	$(x_8 \vee x_3 \vee x_1 \vee x_7)$	2	
x_3	ω_2	$(x_8 \vee x_1 \vee x_7)$	1	

Aprendizaje de cláusulas [63]

$$F = \omega_1 \wedge \omega_2 \wedge \omega_3 \wedge \omega_4 \wedge \omega_5 \wedge \omega_6 \wedge \cdots =$$
$$= (x_1 \vee x_7 \vee \neg x_2) \wedge (x_1 \vee \neg x_3) \wedge (x_2 \vee x_3 \vee x_4)$$
$$\wedge (\neg x_4 \vee \neg x_5) \wedge (x_8 \vee \neg x_4 \vee \neg x_6) \wedge (x_5 \vee x_6) \wedge \ldots$$

$$\sigma = \left\{ x_8 \overset{d}{=} 0@2 \, , \, x_7 \overset{d}{=} 0@3 \, , \, x_1 \overset{d}{=} 0@5 \, , \, x_2 \overset{UP}{=} 0@5 \, , \, x_3 \overset{UP}{=} 0@5 \, , \, x_4 \overset{UP}{=} 1@5 \, , \, x_5 \overset{UP}{=} 0@5 \, , \, x_6 \overset{UP}{=} 0@5 \right\}$$

	Conf	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	$\ldots$
Reason	ω_6	-	ω_1	ω_2	ω_3	ω_4	ω_5	-	-	

Var.	Rea.	Cl. Aprendida	$\#lit_{(dl=5)}$	
-	-	$\omega_6 = (x_5 \vee x_6)$	2	Conflicto
x_5	ω_4	$(x_6 \vee \neg x_4)$	2	
x_6	ω_5	$(\neg x_4 \vee x_8)$	1	**1-UIP [63]**
x_4	ω_3	$(x_8 \vee x_2 \vee x_3)$	2	
x_2	ω_1	$(x_8 \vee x_3 \vee x_1 \vee x_7)$	2	
x_3	ω_2	$(x_8 \vee x_1 \vee x_7)$	1	

Aprendizaje de cláusulas [63]

$$F = \omega_1 \wedge \omega_2 \wedge \omega_3 \wedge \omega_4 \wedge \omega_5 \wedge \omega_6 \wedge \cdots =$$
$$= (x_1 \vee x_7 \vee \neg x_2) \wedge (x_1 \vee \neg x_3) \wedge (x_2 \vee x_3 \vee x_4)$$
$$\wedge (\neg x_4 \vee \neg x_5) \wedge (x_8 \vee \neg x_4 \vee \neg x_6) \wedge (x_5 \vee x_6) \wedge \ldots$$

$$\sigma = \left\{ x_8 \overset{d}{=} 0@2 \, , \, x_7 \overset{d}{=} 0@3 \, , \, x_1 \overset{d}{=} 0@5 \, , \, x_2 \overset{UP}{=} 0@5 \, , \, x_3 \overset{UP}{=} 0@5 \, , \, x_4 \overset{UP}{=} 1@5 \, , \, x_5 \overset{UP}{=} 0@5 \, , \, x_6 \overset{UP}{=} 0@5 \right\}$$

	Conf	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	$\ldots$
Reason	ω_6	-	ω_1	ω_2	ω_3	ω_4	ω_5	-	-	

Var.	Rea.	Cl. Aprendida	$\#lit_{(dl=5)}$	
-	-	$\omega_6 = (x_5 \vee x_6)$	2	Conflicto
x_5	ω_4	$(x_6 \vee \neg x_4)$	2	
x_6	ω_5	$(\neg x_4 \vee x_8)$	1	**1-UIP [63]**
x_4	ω_3	$(x_8 \vee x_2 \vee x_3)$	2	
x_2	ω_1	$(x_8 \vee x_3 \vee x_1 \vee x_7)$	2	
x_3	ω_2	$(x_8 \vee x_1 \vee x_7)$	1	

Aprendizaje de cláusulas [63]

$$F = \omega_1 \wedge \omega_2 \wedge \omega_3 \wedge \omega_4 \wedge \omega_5 \wedge \omega_6 \wedge \cdots =$$
$$= (x_1 \vee x_7 \vee \neg x_2) \wedge (x_1 \vee \neg x_3) \wedge (x_2 \vee x_3 \vee x_4)$$
$$\wedge (\neg x_4 \vee \neg x_5) \wedge (x_8 \vee \neg x_4 \vee \neg x_6) \wedge (x_5 \vee x_6) \wedge \ldots$$

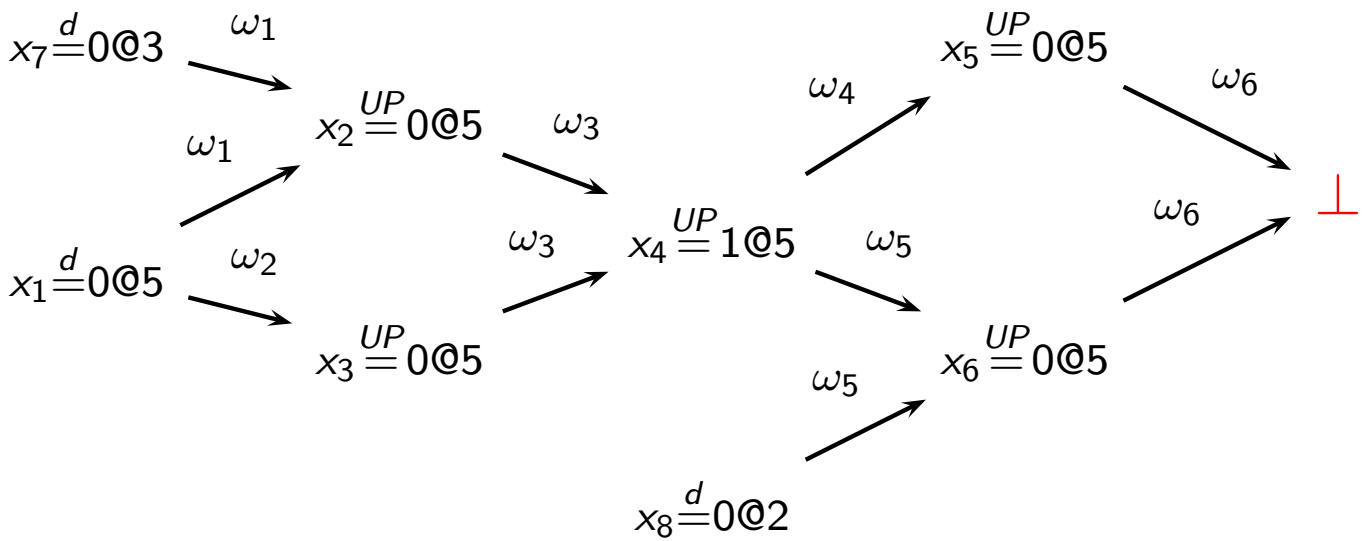
$$\sigma = \left\{ x_8 \overset{d}{=} 0@2 \, , \, x_7 \overset{d}{=} 0@3 \, , \, x_1 \overset{d}{=} 0@5 \, , \, x_2 \overset{UP}{=} 0@5 \, , \, x_3 \overset{UP}{=} 0@5 \, , \, x_4 \overset{UP}{=} 1@5 \, , \, x_5 \overset{UP}{=} 0@5 \, , \, x_6 \overset{UP}{=} 0@5 \right\}$$

	Conf	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	$\ldots$
Reason	ω_6	-	ω_1	ω_2	ω_3	ω_4	ω_5	-	-	

Var.	Rea.	Cl. Aprendida	$\#lit_{(dl=5)}$	
-	-	$\omega_6 = (x_5 \vee x_6)$	2	Conflicto
x_5	ω_4	$(x_6 \vee \neg x_4)$	2	
x_6	ω_5	$(\neg x_4 \vee x_8)$	1	**1-UIP [63]**
x_4	ω_3	$(x_8 \vee x_2 \vee x_3)$	2	
x_2	ω_1	$(x_8 \vee x_3 \vee x_1 \vee x_7)$	2	
x_3	ω_2	$(x_8 \vee x_1 \vee x_7)$	1	

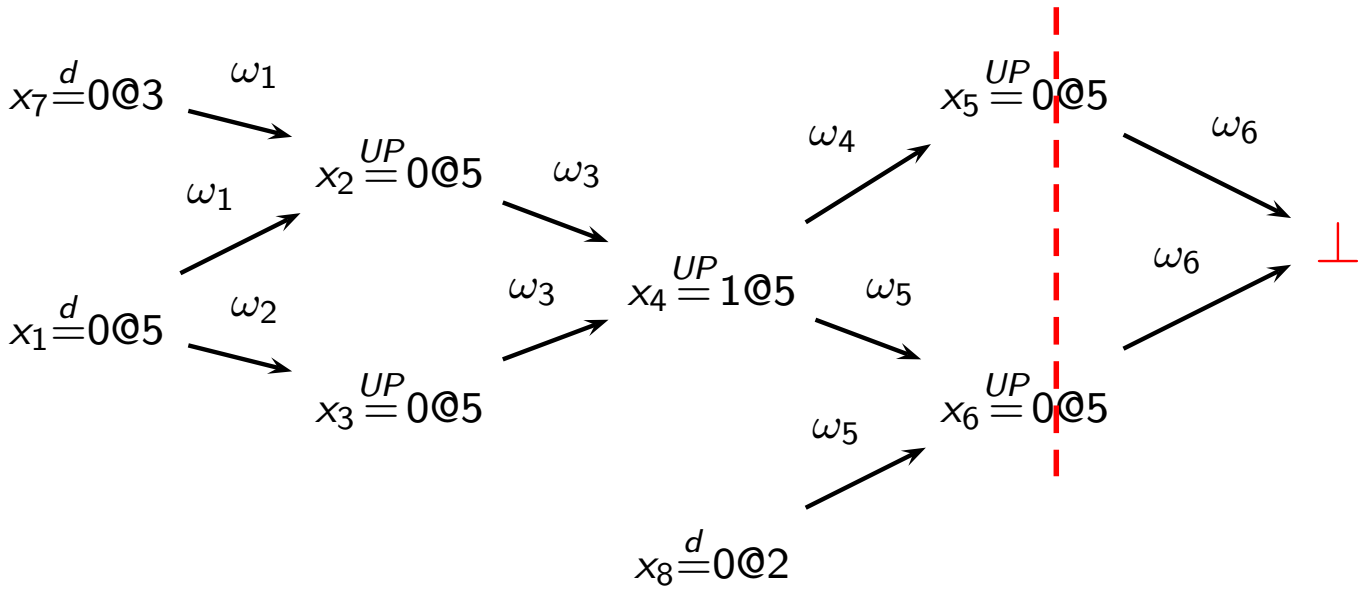
Cláusula aprendida

- Los **UIP** es un **dominador** del grafo de implicaciones
- Se elige el 1UIP ya que habitualmente es la **cláusula más corta**
  - Normalmente se minimiza (*clause minimization*) [63]



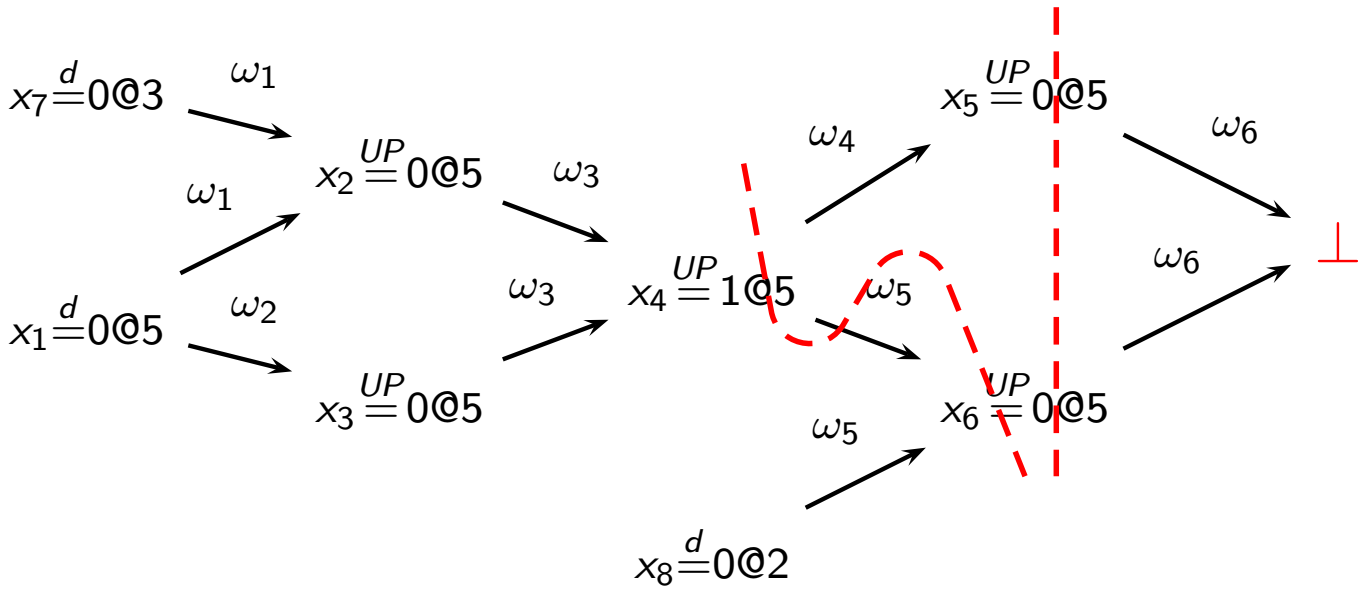
Cláusula aprendida

- Los **UIP** es un **dominador** del grafo de implicaciones
- Se elige el 1UIP ya que habitualmente es la **cláusula más corta**
  - Normalmente se minimiza (*clause minimization*) [63]



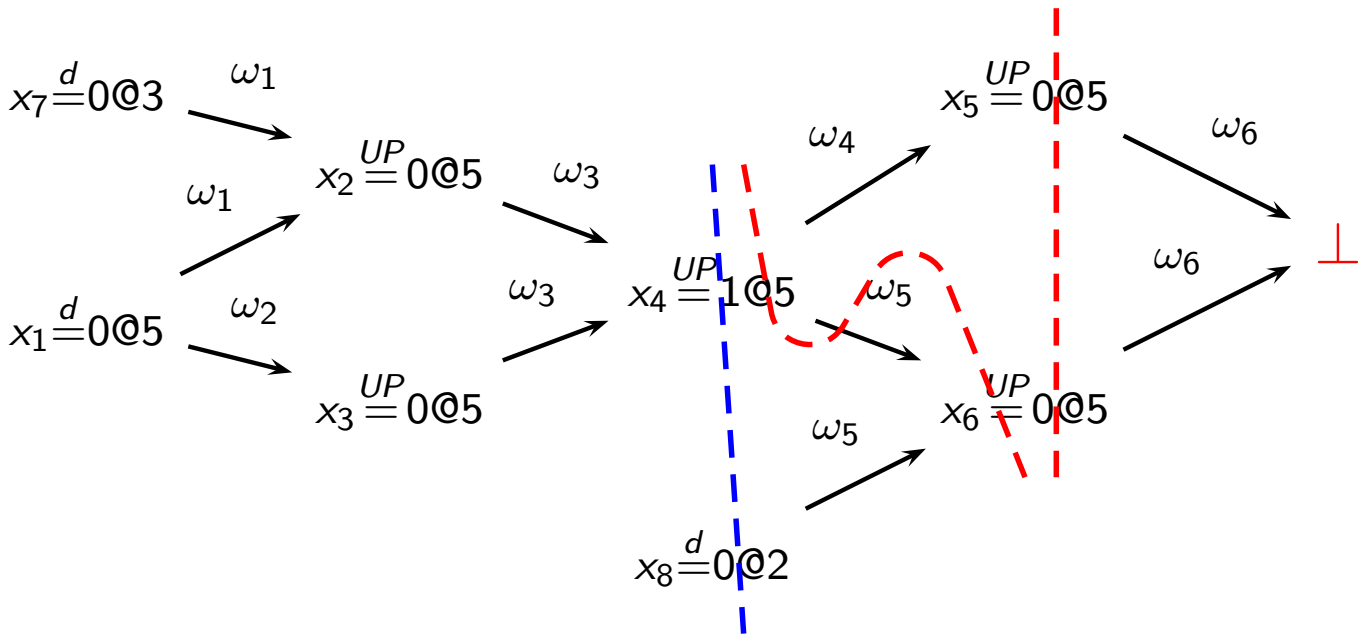
Cláusula aprendida

- Los **UIP** es un **dominador** del grafo de implicaciones
- Se elige el 1UIP ya que habitualmente es la **cláusula más corta**
  - Normalmente se minimiza (*clause minimization*) [63]



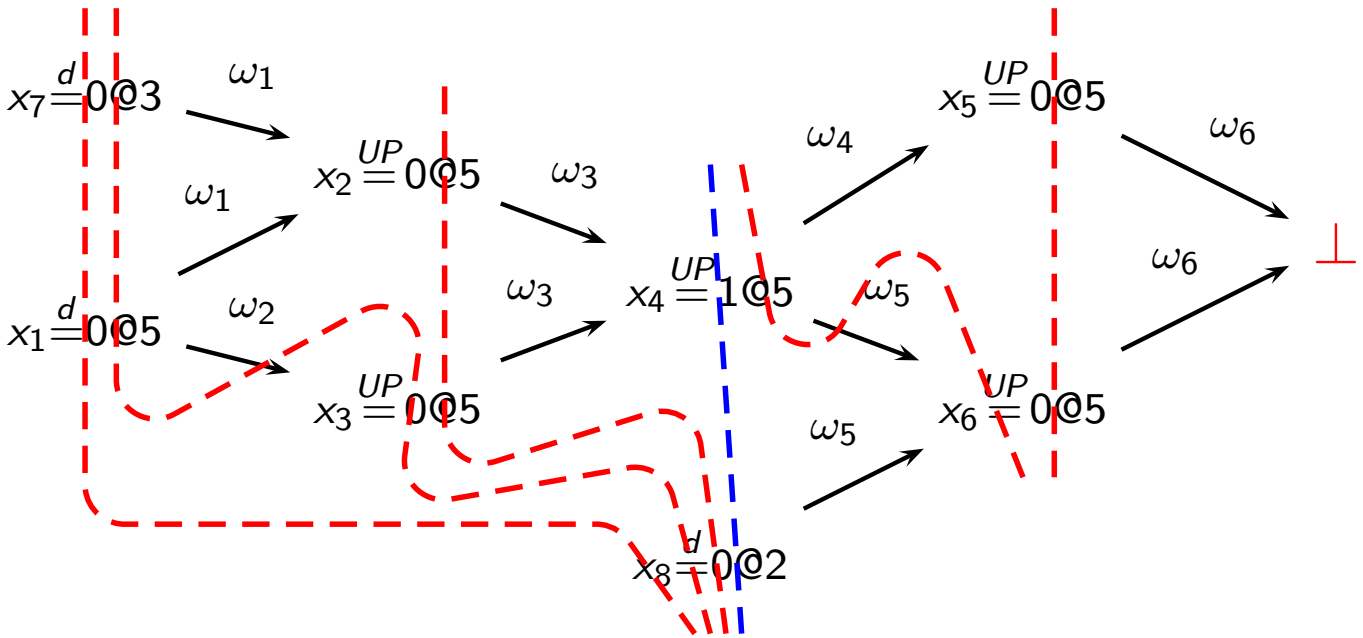
Cláusula aprendida

- Los **UIP** es un **dominador** del grafo de implicaciones
- Se elige el 1UIP ya que habitualmente es la **cláusula más corta**
  - Normalmente se minimiza (*clause minimization*) [63]



Cláusula aprendida

- Los **UIP** es un **dominador** del grafo de implicaciones
- Se elige el 1UIP ya que habitualmente es la **cláusula más corta**
  - Normalmente se minimiza (*clause minimization*) [63]



Backjumping

- Backtracking no cronológico: **backjumping** [63]

$$F = \omega_1 \wedge \omega_2 \wedge \omega_3 \wedge \omega_4 \wedge \omega_5 \wedge \omega_6 \wedge \cdots =$$
$$= (x_1 \vee x_7 \vee \neg x_2) \wedge (x_1 \vee \neg x_3) \wedge (x_2 \vee x_3 \vee x_4)$$
$$\wedge (\neg x_4 \vee \neg x_5) \wedge (x_8 \vee \neg x_4 \vee \neg x_6) \wedge (x_5 \vee x_6) \wedge \ldots$$

$$\sigma = \left\{ x_8 \overset{d}{=} 0@2 \, , \, x_7 \overset{d}{=} 0@3 \, , \, x_1 \overset{d}{=} 0@5 \, , \, x_2 \overset{UP}{=} 0@5 \, , \, x_3 \overset{UP}{=} 0@5 \, , \, x_4 \overset{UP}{=} 1@5 \, , \, x_5 \overset{UP}{=} 0@5 \, , \, x_6 \overset{UP}{=} 0@5 \right\}$$

Cláusula aprendida: $\omega' = (\neg x_4 \vee x_8)$

- En ω': sólo un lit. l_d del nivel dec. del conflicto
- **Backtracking**: **2nd nivel decisión más grande en ω'**

$$\sigma = \left\{ x_8 \overset{d}{=} 0@2 \right\}$$

- **Consecuencia**: l_d se propaga (*asserting literal*)

$$\sigma = \left\{ x_8 \overset{d}{=} 0@2 \, , \, x_4 \overset{UP}{=} 0@2 \right\}$$

Two-watched literals

- Aprender cláusulas (**CL**) evita repetir conflictos, pero...
- ... el coste de UP (lineal) aumenta
- ... y se vuelve inasumible en la práctica
- **Solución**: Estructura de datos "ligera": **Two-Watched Literals** [52]
  - Para cada cláusula dos punteros a sus literales
  - Si alguno satisface la cláusula $\rightsquigarrow$ `OK`
  - Si ambos literales están inasignados $\rightsquigarrow$ `OK`
  - Si sólo uno está inasignado $\rightsquigarrow$ `UP`
  - Si ninguno está inasignado $\rightsquigarrow$ `conflicto!`

Heurística y Borrado de cláusulas

- Aprender cláusulas (**CL**) evita repetir conflictos, pero...
- ... el consumo en memoria es inasumible
- **Solución**: **Borrado de cláusulas**
  - Clave: medir la **utilidad** de cada cláusula aprendida
  - Literal Block Distance: **LBD** [13]
- Pero entonces repetiremos conflictos anteriores...
- ... excepto si se explora ese espacio antes de borrarlas
- **Solución**: **Heurísticas de actividad: VSIDS** [52]
  - Cada variable en un conflicto aumenta su actividad
  - La heurística elige la variable más activa
- CL [63] + VSIDS [52] + LBD [13] **se complementan**
- ..... pero a veces **no es suficiente**! Hacen falta **restarts** [37]

Restarts

- El orden de las decisión afecta a la longitud de la rama que se está explorando
- Existen ramas exponencialmente largas
- Algunos problemas se pueden resolver sin necesidad de explorar estas ramas
- Se puede "comenzar" la búsqueda por ramas cortas

- **Solución**: reiniciar la búsqueda = **Restarts** [37]
  - Restarts periódicos (Luby) vs Restarts adaptativos (**LBD-based**)
  - Condición de completitud: la frecuencia entre restarts debe ser creciente

Preprocessing y Inprocessing

- Existen **dependencias** entre variables y entre cláusulas
  - Ej: variable que representa salida de puerta lógica cuya entrada son otras variables
  - Ej: cláusulas subsumidas por otras
- Si se **detectan** estas dependencias, se reduce el tamaño del problema
- Por tanto, se podrá resolver más rápido!

- **Preprocessing** [29]: detección previa a la búsqueda
- **Inprocessing** [43]: detección durante la búsqueda

CDCL: Un sistema complejo

- **CDCL** resulta relativamente eficiente en la práctica
- Se puede ver en los resultados de las SAT Competitions anuales
- A día de hoy, **no entendemos porqué**
  - MiniSat: 10 parámetros
  - Glucose: 20 parámetros
  - Lingeling: >300 parámetros

- Para cada componente del *algoritmo*:
  - Es fácil conocer los efectos sobre un estado
  - Es **impredecible saber cuándo se activan** (muchas **dependencias** entre componentes)

Métodos incompletos (1)

- Los métodos incompletos no demuestran UNSAT
- Tampoco garantizan encontrar SAT

- Pero en muchos casos **sí encuentran** un *satisfying assignment* (modelo)
- Normalmente lo hacen **muy rápido**
- A menudo, merece la pena hacer una ejecución rápida (con un *timeout* corto) antes de lanzar métodos más costosos

Métodos incompletos (2)

- **SLS**: Stochastic Local Search [58, 57]
  - Basados en **random walk**
  - Búsqueda *aleatoria* en el espaccio de soluciones, normalmente **guiada**
- **SP**: Survey Propagation [18]
  - Método de física estadística
  - *Spin glasses*
- **NeuroSAT** [60, 59, 61]
  - Método basado en Redes Neuronales
  - Usa una arquitectura con:
    - Long Short-Term Memory (LSTM)
    - Multilayer Perceptrons

GSAT [58]

1. Asignación aleatoria de variables
2. Si satisface la fórmula, terminar
3. Si no, computar la variable que produce mayor mejora y cambiar su polaridad ("*flip*")
   - Mejora = mayor incremento de cláusulas satisfechas
4. Repetir (2) hasta `MAX-FLIPS`
5. Repetir (1) hasta `MAX-TRIES`

GSAT [58]

1. Asignación aleatoria de variables
2. Si satisface la fórmula, terminar
3. Si no, computar la variable que produce mayor mejora y cambiar su polaridad ("*flip*")
   - Mejora = mayor incremento de cláusulas satisfechas
4. Repetir (2) hasta `MAX-FLIPS`
5. Repetir (1) hasta `MAX-TRIES`

$$F=(x_1\vee x_2\vee x_3)\wedge(\neg x_1\vee x_2)\wedge(\neg x_1\vee\neg x_2\vee x_3)\wedge(\neg x_3)$$

	x_1	x_2	x_3	ω_1	ω_2	ω_3	ω_4	F
Asignación inicial:	$\bot$	$\bot$	$\bot$					

GSAT [58]

1. Asignación aleatoria de variables
2. Si satisface la fórmula, terminar
3. Si no, computar la variable que produce mayor mejora y cambiar su polaridad ("*flip*")
   - Mejora = mayor incremento de cláusulas satisfechas
4. Repetir (2) hasta `MAX-FLIPS`
5. Repetir (1) hasta `MAX-TRIES`

$$F = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_3)$$

	x_1	x_2	x_3	ω_1	ω_2	ω_3	ω_4	F
Asignación inicial:	$\bot$	$\bot$	$\bot$					
SAT?				No	Sí	Sí	Sí	No

GSAT [58]

1. Asignación aleatoria de variables
2. Si satisface la fórmula, terminar
3. Si no, computar la variable que produce mayor mejora y cambiar su polaridad ("*flip*")
   - Mejora = mayor incremento de cláusulas satisfechas
4. Repetir (2) hasta `MAX-FLIPS`
5. Repetir (1) hasta `MAX-TRIES`

$$F = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_3)$$

	x_1	x_2	x_3	ω_1	ω_2	ω_3	ω_4	F
Asignación inicial:	$\bot$	$\bot$	$\bot$					
SAT?				No	Sí	Sí	Sí	No
Mejora:	0	1	0					

GSAT [58]

1. Asignación aleatoria de variables
2. Si satisface la fórmula, terminar
3. Si no, computar la variable que produce mayor mejora y cambiar su polaridad ("*flip*")
   - Mejora = mayor incremento de cláusulas satisfechas
4. Repetir (2) hasta `MAX-FLIPS`
5. Repetir (1) hasta `MAX-TRIES`

$$F = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_3)$$

	x_1	x_2	x_3	ω_1	ω_2	ω_3	ω_4	F
Asignación inicial:	$\bot$	$\bot$	$\bot$					
SAT?				No	Sí	Sí	Sí	No
Mejora:	0	1	0					
Flip:	$\bot$	$\top$	$\bot$					

GSAT [58]

1. Asignación aleatoria de variables
2. Si satisface la fórmula, terminar
3. Si no, computar la variable que produce mayor mejora y cambiar su polaridad ("*flip*")
   - Mejora = mayor incremento de cláusulas satisfechas
4. Repetir (2) hasta `MAX-FLIPS`
5. Repetir (1) hasta `MAX-TRIES`

$$F = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_3)$$

	x_1	x_2	x_3	ω_1	ω_2	ω_3	ω_4	F
Asignación inicial:	$\bot$	$\bot$	$\bot$					
SAT?				No	Sí	Sí	Sí	No
Mejora:	0	1	0					
Flip:	$\bot$	$\top$	$\bot$					
SAT?				Sí	Sí	Sí	Sí	Sí

GSAT [58]

1. Asignación aleatoria de variables
2. Si satisface la fórmula, terminar
3. Si no, computar la variable que produce mayor mejora y cambiar su polaridad ("*flip*")
   - Mejora = mayor incremento de cláusulas satisfechas
4. Repetir (2) hasta `MAX-FLIPS`
5. Repetir (1) hasta `MAX-TRIES`

$$F = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_3)$$

	x_1	x_2	x_3	ω_1	ω_2	ω_3	ω_4	F
Asignación inicial:	$\bot$	$\bot$	$\bot$					
SAT?				No	Sí	Sí	Sí	No
Mejora:	0	1	0					
Flip:	$\bot$	$\top$	$\bot$					
SAT?				Sí	Sí	Sí	Sí	Sí
– Fin –								

GSAT [58]

1. Asignación aleatoria de variables
2. Si satisface la fórmula, terminar
3. Si no, computar la variable que produce mayor mejora y cambiar su polaridad ("*flip*")
   - Mejora = mayor incremento de cláusulas satisfechas
4. Repetir (2) hasta `MAX-FLIPS`
5. Repetir (1) hasta `MAX-TRIES`

- `MAX-FLIPS` pequeño para encontrar soluciones rápido
  $\Longrightarrow$ **repetir** GSAT hasta `MAX-TRIES`
- `MAX-FLIPS` grande para explorar más espacio de soluciones
  $\Longrightarrow$ problema con **mínimos locales**

WalkSAT

- WalkSAT [57]: Idea similar a GSAT
- Posibilidad de escapar de mínimos locales con movimientos aleatorios
- Con cierta probabilidad p, seleccionar una variable aleatoria para *flip*:
  - p pequeña similar a **GSAT**
  - p alto similar a **MCMC**

Variantes basadas en SLS

- WalkSAT tiene un buen rendimiento en *random* k-CNF
- El modelo aleatorio clásico **"no tiene" mínimos locales**

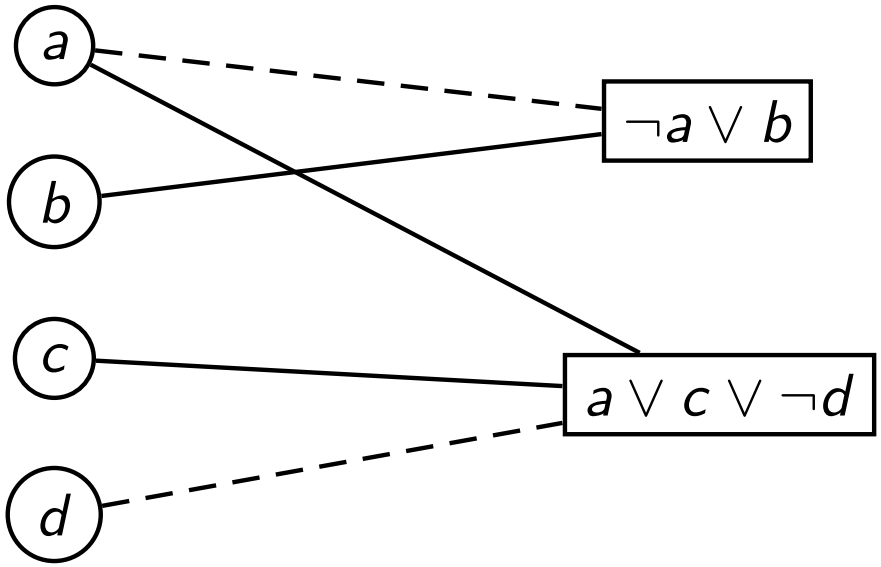
- No suele funcionar bien en instancias industriales
- ¿Relación con la **estructura** de las instancias?

- **Variantes** de WalkSAT para paliar estos problemas [51, 19, 32, 64, 42]
- Aún así, su **rendimiento** suele ser **pobre** comparado con CDCL
- **Preguntas abiertas**: ¿cómo **mejorar** estas técnicas para instancias industriales?

Survey Propagation [18]

- Algoritmo de **paso de mensajes** sobre el *factor graph*

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



- Cada nodo manda *mensaje* a sus nodos vecinos, que depende de los mensajes recibidos. Se repite hasta convergencia
- Mensaje (o *marginal*): probabilidad de que un literal pertenezca a todos los modelos

SP en la práctica

- El cálculo de marginales es **exacto** cuando el *factor graph* es un árbol
- En la práctica esto nunca ocurre: **ciclos**!

- Aún así, **muy efectivo** sobre fórmulas *random* k-CNF
- Resuelve fórmulas **no resueltas** por ningún otro método
  - Fórmulas cercanas al punto de transición de fase
  - Son las fórmulas **más duras** conocidas
- Rendimiento **malísimo** en instancias industriales

- **Preguntas abiertas**: ¿se puede **mejorar** este método para instancias industriales?

MaxSAT

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

- ¿La fórmula F es SAT?
- No, F es UNSAT
- Pero... ¿cuál es el número **máximo** de cláusulas que se pueden satisfacer en F?

- Es un problema de **optimización**!
- Versión de optimización de SAT = **MaxSAT** [47]...
- ... aunque normalmente se resuelve MinUNSAT

MaxSAT

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

- ¿La fórmula F es SAT?
- No, F es UNSAT
- Pero... ¿cuál es el número **máximo** de cláusulas que se pueden satisfacer en F?

- Es un problema de **optimización**!
- Versión de optimización de SAT = **MaxSAT** [47]...
- ... aunque normalmente se resuelve MinUNSAT

Extensiones

- **Weighted MaxSAT**
  - Cada cláusula tiene un **peso** (entero)
  - **Objetivo**: minimizar la suma de los pesos de las cláusulas violadas
- **Partial MaxSAT**
  - Cláusulas *hard*: no se pueden violar
  - Cláusulas *soft*: es posible violarlas
  - **Objetivo**: satisfacer todas las cláusulas *hard* y minimizar el número de cláusulas *soft* violadas
  - Se puede modelar con WMSAT, donde $w(hard) = \infty$
  - En la práctica, $\infty = 1 + sum(w(soft))$
- **Weighted Partial MaxSAT**
  - Combinación de ambos

Ejemplo: *Vertex Coloring*

Un grafo G(V,A) tiene una N-coloración?

$$F = \begin{array}{ll} (x_{i1} \vee x_{i2} \vee \cdots \vee x_{iN}) & \forall i \in V \\ (x_{ij} \rightarrow \neg x_{ij'}) & \forall i \in V, 1 \leq j, j' \leq N \\ (x_{i1} \rightarrow \neg x_{j1}) \wedge \cdots \wedge (x_{iN} \rightarrow \neg x_{jN}) & \forall i \in V, \langle i,j \rangle \in A \end{array}$$

Permite que los nodos tengan color `undef`.
¿Cuál es el **mínimo número** de nodos con color *indefinido*?

$$F' = \begin{array}{ll} \{(x_{i1} \vee x_{i2} \vee \cdots \vee x_{iN} \vee x_{i(N+1)}), \infty\} & \forall i \in V \\ \{(x_{ij} \rightarrow \neg x_{ij'}), \infty\} & \forall i \in V, 1 \leq j, j' \leq N+1 \\ \{(x_{i1} \rightarrow \neg x_{j1}) \wedge \cdots \wedge (x_{iN} \rightarrow \neg x_{jN}), \infty\} & \forall i \in V, \langle i,j \rangle \in A \\ \{\neg x_{i(N+1)}, 1\} & \forall i \in V \end{array}$$

Ejemplo: *Vertex Coloring*

Un grafo G(V,A) tiene una N-coloración?

$$F = \begin{array}{ll} (x_{i1} \vee x_{i2} \vee \cdots \vee x_{iN}) & \forall i \in V \\ (x_{ij} \rightarrow \neg x_{ij'}) & \forall i \in V, 1 \leq j, j' \leq N \\ (x_{i1} \rightarrow \neg x_{j1}) \wedge \cdots \wedge (x_{iN} \rightarrow \neg x_{jN}) & \forall i \in V, \langle i,j \rangle \in A \end{array}$$

Permite que los nodos tengan color `undef`.
¿Cuál es el **mínimo número** de nodos con color *indefinido*?

$$F' = \begin{array}{ll} \{(x_{i1} \vee x_{i2} \vee \cdots \vee x_{iN} \vee x_{i(N+1)}), \infty\} & \forall i \in V \\ \{(x_{ij} \rightarrow \neg x_{ij'}), \infty\} & \forall i \in V, 1 \leq j, j' \leq N+1 \\ \{(x_{i1} \rightarrow \neg x_{j1}) \wedge \cdots \wedge (x_{iN} \rightarrow \neg x_{jN}), \infty\} & \forall i \in V, \langle i,j \rangle \in A \\ \{\neg x_{i(N+1)}, 1\} & \forall i \in V \end{array}$$

Ejemplo: *Vertex Coloring*

Un grafo G(V,A) tiene una N-coloración?

$$
F = \begin{array}{ll}
(x_{i1} \vee x_{i2} \vee \cdots \vee x_{iN}) & \forall i \in V \\
(x_{ij} \rightarrow \neg x_{ij'}) & \forall i \in V, 1 \leq j, j' \leq N \\
(x_{i1} \rightarrow \neg x_{j1}) \wedge \cdots \wedge (x_{iN} \rightarrow \neg x_{jN}) & \forall i \in V, \langle i, j \rangle \in A
\end{array}
$$

Permite que los nodos tengan color `undef`.
¿Cuál es el **mínimo número** de nodos con color *indefinido*?

$$
F' = \begin{array}{ll}
\{(x_{i1} \vee x_{i2} \vee \cdots \vee x_{iN} \vee x_{i(N+1)}), \infty\} & \forall i \in V \\
\{(x_{ij} \rightarrow \neg x_{ij'}), \infty\} & \forall i \in V, 1 \leq j, j' \leq N+1 \\
\{(x_{i1} \rightarrow \neg x_{j1}) \wedge \cdots \wedge (x_{iN} \rightarrow \neg x_{jN}), \infty\} & \forall i \in V, \langle i, j \rangle \in A \\
\{\neg x_{i(N+1)}, 1\} & \forall i \in V
\end{array}
$$

Resolver MaxSAT

- Métodos aproximados (incompletos)
- Branch and Bound

- Métodos **SAT-based** [7, 49, 50]
  - Llamadas iterativas a un SAT solver
  - Ej: llamada inicial con *hard* $\rightsquigarrow$ cota superior de la solución + sucesivas llamadas para mejorar esta cota
  - Aprovechan el buen rendimiento de CDCL (especialmente en instancias industriales)
  - Métodos SAT-UNSAT y métodos UNSAT-SAT

- Elemento **crítico**: **codificación** del problema
- **Estrategia** de resolución: **depende del problema**
- Problemas **relacionados**: **reparación de soluciones** (robustez)

Referencias I

[1] M. Alekhnovich, E. Ben-Sasson, A. A. Razborov, and A. Wigderson.
Space complexity in propositional calculus.
SIAM J. Comput., 31(4):1184–1211, 2002.

[2] C. Ansótegui, M. L. Bonet, J. Giráldez-Cru, and J. Levy.
The fractal dimension of SAT formulas.
In *Automated Reasoning - 7th International Joint Conference, IJCAR 2014, Held as Part of the Vienna Summer of Logic, VSL 2014, Vienna, Austria, July 19-22, 2014. Proceedings*, pages 107–121, 2014.

Referencias II

[3] C. Ansótegui, M. L. Bonet, J. Giráldez-Cru, and J. Levy.
On the classification of industrial SAT families.
In *Artificial Intelligence Research and Development - Proceedings of the 18th International Conference of the Catalan Association for Artificial Intelligence, Valencia, Catalonia, Spain, October 21-23, 2015.*, pages 163–172, 2015.

[4] C. Ansótegui, M. L. Bonet, J. Giráldez-Cru, and J. Levy.
Structure features for SAT instances classification.
J. Applied Logic, 23:27–39, 2017.

Referencias III

[5] C. Ansótegui, M. L. Bonet, and J. Levy.
On the structure of industrial SAT instances.
In *Principles and Practice of Constraint Programming - CP 2009, 15th International Conference, CP 2009, Lisbon, Portugal, September 20-24, 2009, Proceedings*, pages 127–141, 2009.

[6] C. Ansótegui, M. L. Bonet, and J. Levy.
Towards industrial-like random SAT instances.
In *IJCAI 2009, Proceedings of the 21st International Joint Conference on Artificial Intelligence, Pasadena, California, USA, July 11-17, 2009*, pages 387–392, 2009.

[7] C. Ansótegui, M. L. Bonet, and J. Levy.
SAT-based MaxSAT algorithms.
Artif. Intell., 196:77–105, 2013.

Referencias IV

[8] C. Ansótegui, J. Giráldez-Cru, and J. Levy.
The community structure of SAT formulas.
In *Theory and Applications of Satisfiability Testing - SAT 2012 - 15th International Conference, Trento, Italy, June 17-20, 2012. Proceedings*, pages 410–423, 2012.

[9] C. Ansótegui, J. Giráldez-Cru, J. Levy, and L. Simon.
Using community structure to detect relevant learnt clauses.
In *Theory and Applications of Satisfiability Testing - SAT 2015 - 18th International Conference, Austin, TX, USA, September 24-27, 2015, Proceedings*, pages 238–254, 2015.

Referencias V

[10] C. Ansótegui, M. Sellmann, and K. Tierney.
A gender-based genetic algorithm for the automatic configuration of algorithms.
In *Principles and Practice of Constraint Programming - CP 2009, 15th International Conference, CP 2009, Lisbon, Portugal, September 20-24, 2009, Proceedings*, pages 142–157, 2009.

[11] A. Atserias and V. Dalmau.
A combinatorial characterization of resolution width.
J. Comput. Syst. Sci., 74(3):323–334, 2008.

[12] A. Atserias, M. Lauria, and J. Nordström.
Narrow proofs may be maximally long.
In *IEEE 29th Conference on Computational Complexity, CCC 2014, Vancouver, BC, Canada, June 11-13, 2014*, pages 286–297, 2014.

Referencias VI

[13] G. Audemard and L. Simon.
Predicting learnt clauses quality in modern SAT solvers.
In *IJCAI 2009, Proceedings of the 21st International Joint Conference on Artificial Intelligence, Pasadena, California, USA, July 11-17, 2009*, pages 399–404, 2009.

[14] G. Baud-Berthier, J. Giráldez-Cru, and L. Simon.
On the community structure of bounded model checking SAT problems.
In *Theory and Applications of Satisfiability Testing - SAT 2017 - 20th International Conference, Melbourne, VIC, Australia, August 28 - September 1, 2017, Proceedings*, pages 65–82, 2017.

Referencias VII

[15] A. Biere.
Bounded model checking.
In Biere et al. [16], pages 457–481.

[16] A. Biere, M. Heule, H. van Maaren, and T. Walsh, editors.
Handbook of Satisfiability, volume 185 of *Frontiers in Artificial Intelligence and Applications*. IOS Press, 2009.

[17] A. Blake.
Canonical Expressions in Boolean Algebra.
Ph.D. Dissertation. University of Chicago, 1937.

[18] A. Braunstein, M. Mézard, and R. Zecchina.
Survey propagation: An algorithm for satisfiability.
Random Struct. Algorithms, 27(2):201–226, 2005.

Referencias VIII

[19] B. Cha and K. Iwama.
Adding new clauses for faster local search.
In *Proceedings of the Thirteenth National Conference on Artificial Intelligence and Eighth Innovative Applications of Artificial Intelligence Conference, AAAI 96, IAAI 96, Portland, Oregon, USA, August 4-8, 1996, Volume 1.*, pages 332–337, 1996.

[20] V. Chvátal and E. Szemerédi.
Many hard examples for resolution.
J. ACM, 35(4):759–768, 1988.

[21] Clay Mathematics Institute.
Millennium prize problems.
https://web.archive.org/web/20051126084715/http://www.claymath.org:80/millennium/.

Referencias IX

[22] M. Clegg, J. Edmonds, and R. Impagliazzo.
Using the Groebner basis algorithm to find proofs of unsatisfiability.
In *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, USA, May 22-24, 1996*, pages 174–183, 1996.

[23] S. A. Cook.
The complexity of theorem-proving procedures.
In *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, STOC '71, pages 151–158, New York, NY, USA, 1971. ACM.

[24] W. J. Cook, C. R. Coullard, and G. Turán.
On the complexity of cutting-plane proofs.
Discrete Applied Mathematics, 18(1):25–38, 1987.

Referencias X

[25] A. Darwiche and K. Pipatsrisawat.
Complete algorithms.
In Biere et al. [16], pages 99–130.

[26] M. Davis, G. Logemann, and D. Loveland.
A machine program for theorem-proving.
Commun. ACM, 5(7):394–397, July 1962.

[27] M. Davis and H. Putnam.
A computing procedure for quantification theory.
J. ACM, 7(3):201–215, 1960.

Referencias XI

[28] J. Dubois-Lacoste, M. López-Ibáñez, and T. Stützle.
Automatic configuration of state-of-the-art multi-objective optimizers using the TP+PLS framework.
In *13th Annual Genetic and Evolutionary Computation Conference, GECCO 2011, Proceedings, Dublin, Ireland, July 12-16, 2011*, pages 2019–2026, 2011.

[29] N. Eén and A. Biere.
Effective preprocessing in SAT through variable and clause elimination.
In *Theory and Applications of Satisfiability Testing, 8th International Conference, SAT 2005, St. Andrews, UK, June 19-23, 2005, Proceedings*, pages 61–75, 2005.

Referencias XII

[30] J. Elffers, J. Giráldez-Cru, S. Gocht, J. Nordström, and L. Simon.
Seeking practical CDCL insights from theoretical SAT benchmarks.
In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden.*, pages 1300–1308, 2018.

[31] J. Elffers, J. Giráldez-Cru, J. Nordström, and M. Vinyals.
Using combinatorial benchmarks to probe the reasoning power of pseudo-boolean solvers.
In *Theory and Applications of Satisfiability Testing - SAT 2018 - 21st International Conference, SAT 2018, Held as Part of the Federated Logic Conference, FloC*

Referencias XIII

2018, Oxford, UK, July 9-12, 2018, Proceedings, pages 75–93, 2018.

[32] J. Frank.
Learning short-term weights for GSAT.
In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence, IJCAI 97, Nagoya, Japan, August 23-29, 1997, 2 Volumes*, pages 384–391, 1997.

[33] M. R. Garey and D. S. Johnson.
Computers and Intractability: A Guide to the Theory of NP-Completeness.
W. H. Freeman, 1979.

Referencias XIV

[34] J. Giráldez-Cru and J. Levy.
A modularity-based random SAT instances generator.
In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pages 1952–1958, 2015.

[35] J. Giráldez-Cru and J. Levy.
Generating SAT instances with community structure.
Artif. Intell., 238:119–134, 2016.

[36] J. Giráldez-Cru and J. Levy.
Locality in random SAT instances.
In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 638–644, 2017.

Referencias XV

[37] C. P. Gomes, B. Selman, and H. A. Kautz.
Boosting combinatorial search through randomization.
In *Proceedings of the Fifteenth National Conference on Artificial Intelligence and Tenth Innovative Applications of Artificial Intelligence Conference, AAAI 98, IAAI 98, July 26-30, 1998, Madison, Wisconsin, USA.*, pages 431–437, 1998.

[38] A. Haken.
The intractability of resolution.
Theor. Comput. Sci., 39:297–308, 1985.

[39] M. Heule and H. van Maaren.
Look-ahead based SAT solvers.
In Biere et al. [16], pages 155–184.

Referencias XVI

[40] F. Hutter, H. H. Hoos, and K. Leyton-Brown.
Sequential model-based optimization for general algorithm configuration.
In *Learning and Intelligent Optimization - 5th International Conference, LION 5, Rome, Italy, January 17-21, 2011. Selected Papers*, pages 507–523, 2011.

[41] F. Hutter, H. H. Hoos, K. Leyton-Brown, and T. Stützle.
Paramils: An automatic algorithm configuration framework.
J. Artif. Intell. Res., 36:267–306, 2009.

Referencias XVII

[42] A. Ishtaiwi, J. Thornton, Anbulagan, A. Sattar, and D. N. Pham.
Adaptive clause weight redistribution.
In *Principles and Practice of Constraint Programming - CP 2006, 12th International Conference, CP 2006, Nantes, France, September 25-29, 2006, Proceedings*, pages 229–243, 2006.

[43] M. Järvisalo, M. Heule, and A. Biere.
Inprocessing rules.
In *Automated Reasoning - 6th International Joint Conference, IJCAR 2012, Manchester, UK, June 26-29, 2012. Proceedings*, pages 355–370, 2012.

Referencias XVIII

[44] S. Kadioglu, Y. Malitsky, M. Sellmann, and K. Tierney.
ISAC - instance-specific algorithm configuration.
In *ECAI 2010 - 19th European Conference on Artificial Intelligence, Lisbon, Portugal, August 16-20, 2010, Proceedings*, pages 751–756, 2010.

[45] G. Katsirelos and L. Simon.
Eigenvector centrality in industrial SAT instances.
In *Principles and Practice of Constraint Programming - 18th International Conference, CP 2012, Québec City, QC, Canada, October 8-12, 2012. Proceedings*, pages 348–356, 2012.

[46] H. A. Kautz, A. Sabharwal, and B. Selman.
Incomplete algorithms.
In Biere et al. [16], pages 185–203.

Referencias XIX

[47] C. M. Li and F. Manyà.
MaxSAT, hard and soft constraints.
In Biere et al. [16], pages 613–631.

[48] Y. Malitsky, A. Sabharwal, H. Samulowitz, and M. Sellmann.
Non-model-based algorithm portfolios for SAT.
In *Theory and Applications of Satisfiability Testing - SAT 2011 - 14th International Conference, SAT 2011, Ann Arbor, MI, USA, June 19-22, 2011. Proceedings*, pages 369–370, 2011.

Referencias XX

[49] V. M. Manquinho, R. Martins, and I. Lynce.
Improving unsatisfiability-based algorithms for Boolean optimization.
In *Theory and Applications of Satisfiability Testing - SAT 2010, 13th International Conference, SAT 2010, Edinburgh, UK, July 11-14, 2010. Proceedings*, pages 181–193, 2010.

[50] R. Martins, V. M. Manquinho, and I. Lynce.
Open-WBO: A modular MaxSAT solver,.
In *Theory and Applications of Satisfiability Testing - SAT 2014 - 17th International Conference, Held as Part of the Vienna Summer of Logic, VSL 2014, Vienna, Austria, July 14-17, 2014. Proceedings*, pages 438–445, 2014.

Referencias XXI

[51] P. Morris.
The breakout method for escaping from local minima.
In *Proceedings of the 11th National Conference on Artificial Intelligence. Washington, DC, USA, July 11-15, 1993.*, pages 40–45, 1993.

[52] M. W. Moskewicz, C. F. Madigan, Y. Zhao, L. Zhang, and S. Malik.
Chaff: Engineering an efficient SAT solver.
In *Proceedings of the 38th Design Automation Conference, DAC 2001, Las Vegas, NV, USA, June 18-22, 2001*, pages 530–535, 2001.

[53] J. Nordström.
On the interplay between proof complexity and SAT solving.
SIGLOG News, 2(3):19–44, 2015.

Referencias XXII

[54] A. A. Razborov.
Propositional proof complexity.
J. ACM, 50(1):80–82, 2003.

[55] A. A. Razborov.
Resolution lower bounds for the weak functional pigeonhole principle.
Theor. Comput. Sci., 303(1):233–243, 2003.

[56] J. Rintanen.
Planning and SAT.
In Biere et al. [16], pages 483–504.

Referencias XXIII

[57] B. Selman, H. Kautz, and B. Cohen.
Local search strategies for satisfiability testing.
In *Cliques, Coloring and Satisfiability: the Second DIMACS Implementation Challenge, volume 26 of DIMACS Series in DMTCS. Amer. Math. Soc., 1996*, pages 521–532, 1996.

[58] B. Selman, H. J. Levesque, and D. G. Mitchell.
A new method for solving hard satisfiability problems.
In *Proceedings of the 10th National Conference on Artificial Intelligence, San Jose, CA, USA, July 12-16, 1992.*, pages 440–446, 1992.

Referencias XXIV

[59] D. Selsam and N. Bjørner.
Guiding high-performance SAT solvers with unsat-core predictions.
In M. Janota and I. Lynce, editors, *Theory and Applications of Satisfiability Testing - SAT 2019 - 22nd International Conference, SAT 2019, Lisbon, Portugal, July 9-12, 2019, Proceedings*, volume 11628 of *Lecture Notes in Computer Science*, pages 336–353. Springer, 2019.

[60] D. Selsam, M. Lamm, B. Bünz, P. Liang, L. de Moura, and D. L. Dill.
Learning a SAT solver from single-bit supervision.
CoRR, abs/1802.03685, 2018.

Referencias XXV

[61] D. Selsam, M. Lamm, B. Bünz, P. Liang, L. de Moura, and D. L. Dill.
Learning a SAT solver from single-bit supervision.
In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[62] J. P. M. Silva, I. Lynce, and S. Malik.
Conflict-driven clause learning SAT solvers.
In Biere et al. [16], pages 131–153.

[63] J. P. M. Silva and K. A. Sakallah.
GRASP: A search algorithm for propositional satisfiability.
IEEE Trans. Computers, 48(5):506–521, 1999.

Referencias XXVI

[64] J. Thornton, D. N. Pham, S. Bain, and V. F. Jr.
Additive versus multiplicative clause weighting for SAT.
In *Proceedings of the Nineteenth National Conference on Artificial Intelligence, Sixteenth Conference on Innovative Applications of Artificial Intelligence, July 25-29, 2004, San Jose, California, USA*, pages 191–196, 2004.

[65] M. Vinyals, J. Elffers, J. Giráldez-Cru, S. Gocht, and J. Nordström.
In between resolution and cutting planes: A study of proof systems for pseudo-boolean SAT solving.
In *Theory and Applications of Satisfiability Testing - SAT 2018 - 21st International Conference, SAT 2018, Held as Part of the Federated Logic Conference, FloC 2018, Oxford, UK, July 9-12, 2018, Proceedings*, pages 292–310, 2018.

Referencias XXVII

[66] Wikipedia.
P versus NP.
https://en.wikipedia.org/wiki/P_versus_NP_problem.

[67] L. Xu, F. Hutter, H. H. Hoos, and K. Leyton-Brown.
The design and analysis of an algorithm portfolio for SAT.
In *Principles and Practice of Constraint Programming - CP 2007, 13th International Conference, CP 2007, Providence, RI, USA, September 23-27, 2007, Proceedings*, pages 712–727, 2007.

[68] L. Xu, F. Hutter, H. H. Hoos, and K. Leyton-Brown.
Satzilla: Portfolio-based algorithm selection for SAT.
J. Artif. Intell. Res., 32:565–606, 2008.