

en resumen...

- Características:
 - Implication graphs para BCP y análisis de conflictos
 - Learning nuevas cláusulas
 - Non-chronological backtracking

Heurísticas de decisión GRASP

- Procedure `decide()`
 - Qué variable modificar
 - Qué valor asignar

- Heurística por Default en GRASP:

Escoger la variable y asignación que satisface directamente el mayor número de cláusulas

- Existen otras posibilidades de heurísticas

Deducción GRASP

- **Boolean Constraint Propagation** usando **implication graphs**

Ejem. para la cláusula $\omega = (x \vee \neg y)$, si $y=1$, entonces debemos tener $x=1$

- Para una variable x en una cláusula, asignar 0 a las demás literales es llamado **antecedent assignment** $A(x)$

- Ejem. para $\omega = (x \vee y \vee \neg z)$,

$$A(x) = \{(y,0), (z,1)\}, A(y) = \{(x,0), (z,1)\}, A(z) = \{(x,0), (y,0)\}$$

- Las variables son directamente “responsables” para forzar el valor de x
- Antecedentes de una asignación de una variable de decisión es vacío

Implication Graphs (IG)

- Cada nodo es una asignación de variable $x=v(x)$ (decision o implied)
- Predecesores de x son asignaciones antecesoras $A(x)$
 - No predecesoras para asignaciones de decisión!
- Vértices de conflicto κ tienen $A(\kappa)$ = asignaciones a variables en cláusulas no satisfactibles
- Nivel de decisión para una asignación implícita es

$$\delta(x) = \max\{\delta(y) \mid (y, v(y)) \in A(x)\}$$

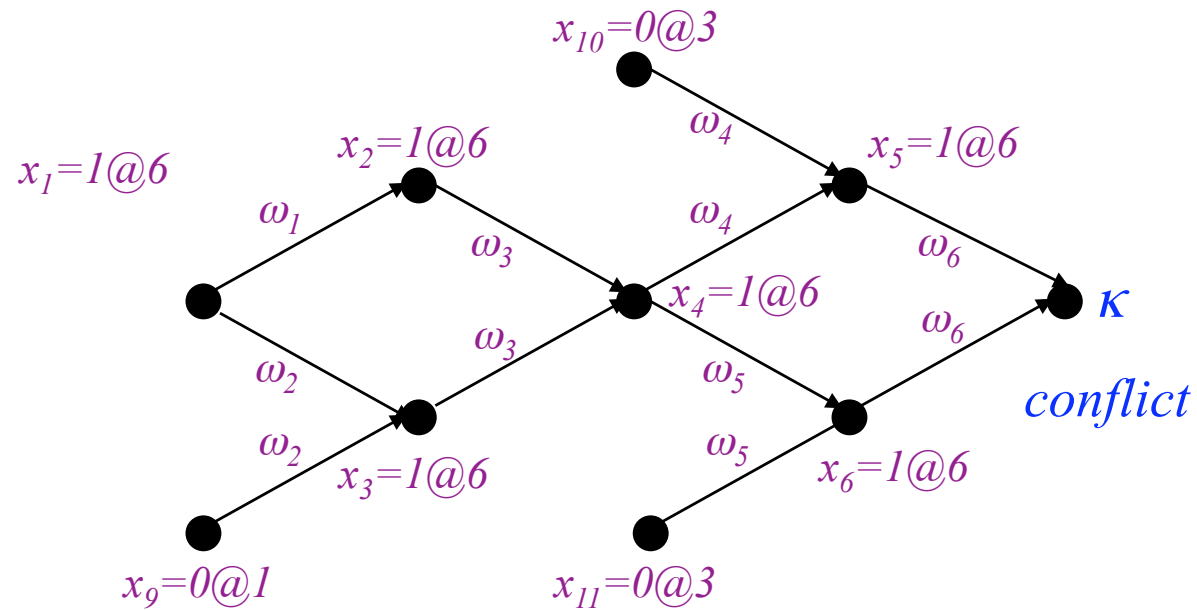
Análisis de conflictos en GRASP

- Después de un conflicto, analizar la gráfica de implicación en el nivel de decisión
- Agregar nuevas cláusulas que podrían prevenir la ocurrencia del mismo conflicto en el futuro $\Rightarrow$ Learning
- Determinar el nivel de decisión a regresar, podría no ser el inmediato $\Rightarrow$ Non-chronological backtracking

Aprendizaje

- Determinar la asignación que causó el conflicto en κ
 - Recorrido hacia atrás de la IG, encontrar la raíz de la IG en el **transitive fanin** de κ
 - La asignación es una condición necesaria para κ
- La Negación de esta asignación es llamada cláusula de conflictos inducidos $\omega_C(\kappa)$
 - Agregando $\omega_C(\kappa)$ a la base de datos de cláusulas prevendrá la ocurrencia de κ (de nuevo).

Aprendizaje



$$\omega_C(K) = (\neg x_1 \vee x_9 \vee x_{10} \vee x_{11})$$

Aprendizaje

- Aprendizaje de nuevas cláusulas incrementa el tamaño de la base de datos de cláusulas
- El incremento puede ser exponencial
- Heuristicamente eliminar cláusulas basándose en parámetros dado por el usuario
 - Si el tamaño de la cláusula aprendida $>$ parámetro, no incluirla

Backtracking

Failure driven assertions (FDA):

- Si $\omega_C(K)$ implica la variable de decisión en curso, una diferente asignación para la variable actual es intentado.
- En nuestro IG, después de borrar la asignación en el nivel 6, $\omega_C(K)$ se convierte en una cláusula $\neg x_1$
- Esto implica inmediatamente $x_1=0$

Ejemplo Implication Graph

Asignación actual: $\{x_9=0@1, x_{10}=0@3, x_{11}=0@3, x_{12}=1@2, x_{13}=1@2\}$

Última asignación: $\{x_1=1@6\}$

$$\omega_1 = (\neg x_1 \vee x_2)$$

$$\omega_2 = (\neg x_1 \vee x_3 \vee x_9)$$

$$\omega_3 = (\neg x_2 \vee \neg x_3 \vee x_4)$$

$$\omega_4 = (\neg x_4 \vee x_5 \vee x_{10})$$

$$\omega_5 = (\neg x_4 \vee x_6 \vee x_{11})$$

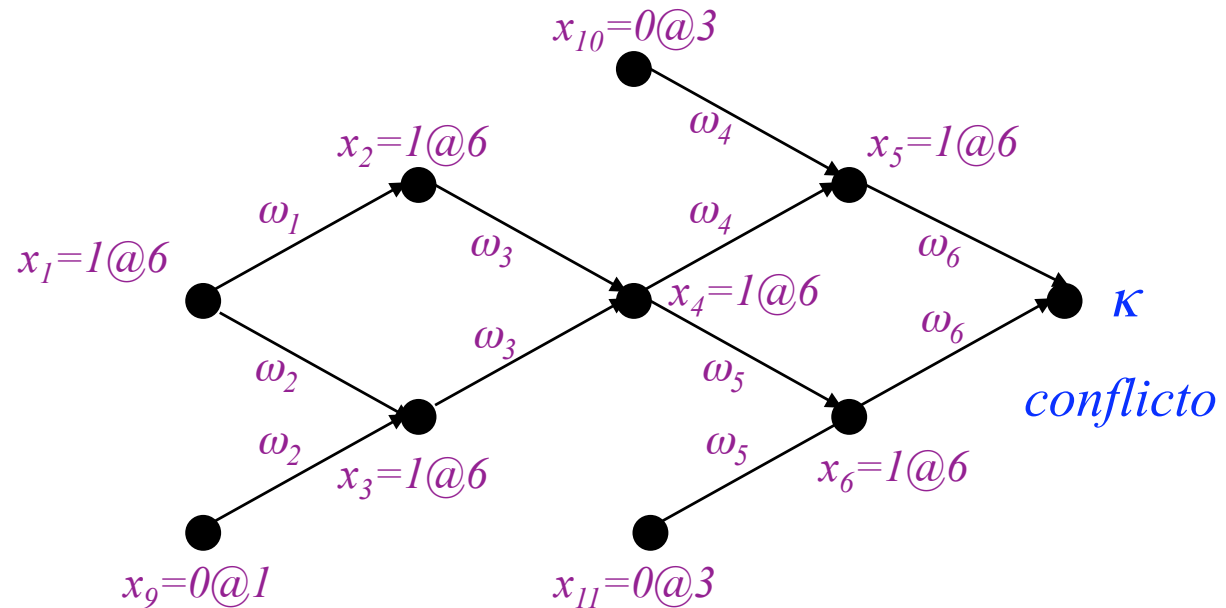
$$\omega_6 = (\neg x_5 \vee \neg x_6)$$

$$\omega_7 = (x_1 \vee x_7 \vee \neg x_{12})$$

$$\omega_8 = (x_1 \vee x_8)$$

$$\omega_9 = (\neg x_7 \vee \neg x_8 \vee \neg x_{13})$$

$$\omega_C(K) = (\neg x_1 \vee x_9 \vee x_{10} \vee x_{11})$$



Ejemplo Implication Graph

Asignación actual: $\{x_9=0@1, x_{10}=0@3, x_{11}=0@3, x_{12}=1@2, x_{13}=1@2\}$

Última asignación: $\{x_1=0@6\}$

$$\omega_1 = (\neg x_1 \vee x_2)$$

$$\omega_2 = (\neg x_1 \vee x_3 \vee x_9)$$

$$\omega_3 = (\neg x_2 \vee \neg x_3 \vee x_4)$$

$$\omega_4 = (\neg x_4 \vee x_5 \vee x_{10})$$

$$\omega_5 = (\neg x_4 \vee x_6 \vee x_{11})$$

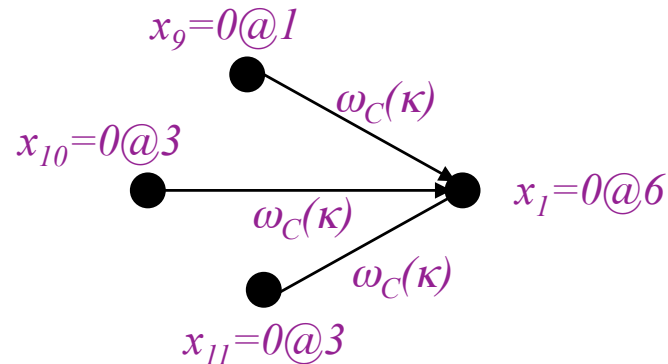
$$\omega_6 = (\neg x_5 \vee x_6)$$

$$\omega_7 = (x_1 \vee x_7 \vee \neg x_{12})$$

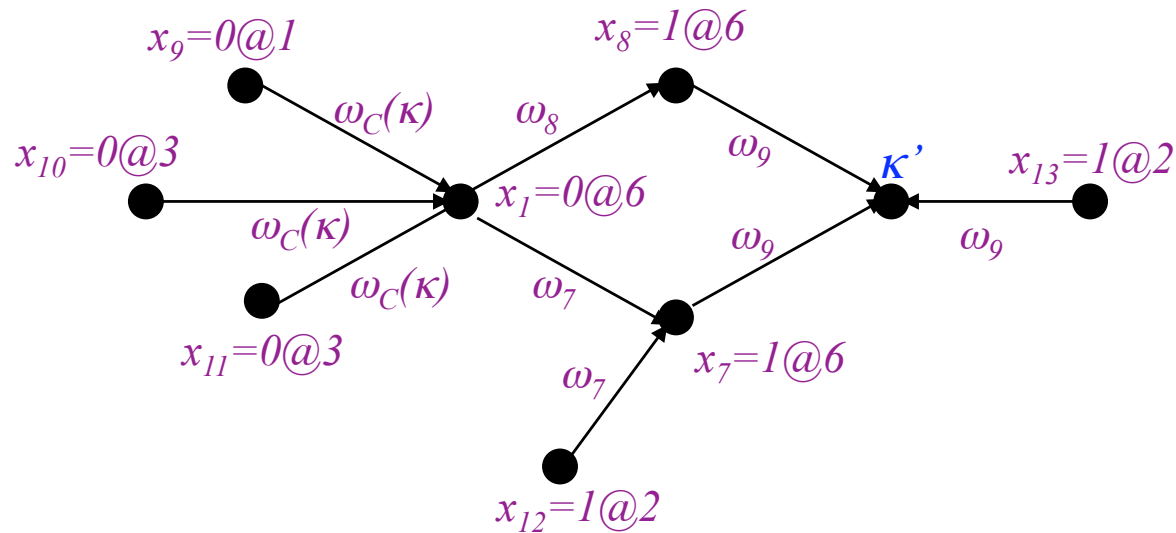
$$\omega_8 = (x_1 \vee x_8)$$

$$\omega_9 = (\neg x_7 \vee \neg x_8 \vee \neg x_{13})$$

$$\omega_C(\kappa) = (\neg x_1 \vee x_9 \vee x_{10} \vee x_{11})$$

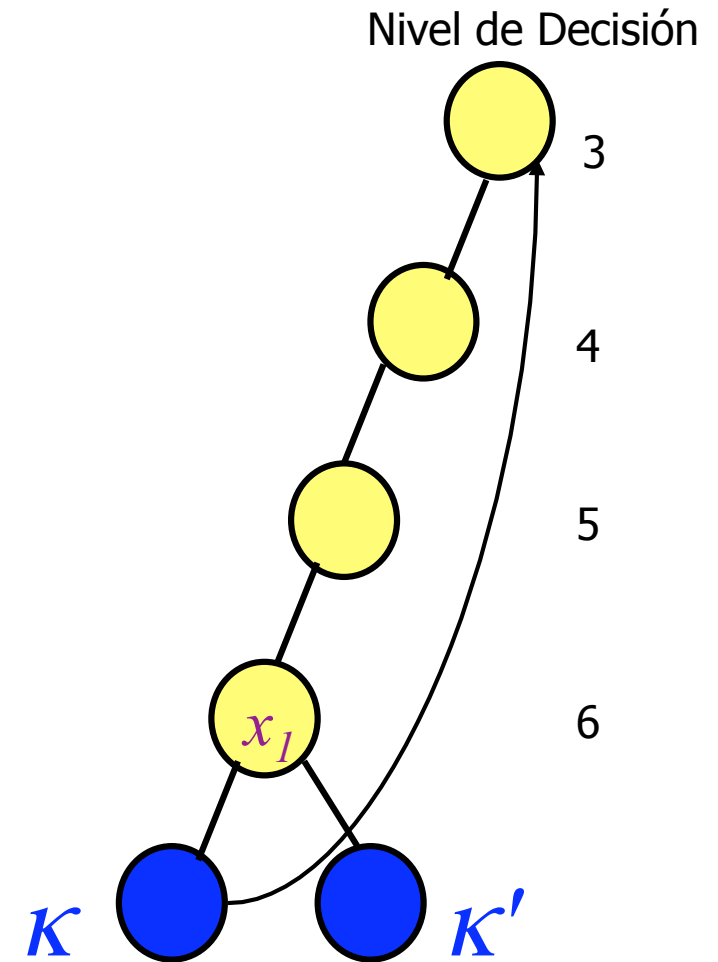


Non-chronological backtracking



$$A_C(\kappa') = \{x_9=0@1, x_{10}=0@3, x_{11}=0@3, x_{12}=1@2, x_{13}=1@2\}$$

$$\omega_C(\kappa) = (x_9 \vee x_{10} \vee x_{11} \vee \neg x_{12} \vee \neg x_{13})$$



Ejemplo

$$\omega_1 = (x_1 \vee x_2 \vee x_4 \vee x_5)$$

$$\omega_2 = (x_1 \vee x_2 \vee x_4 \vee \neg x_5)$$

$$\omega_3 = (\neg x_3 \vee x_6)$$

$$\omega_4 = (x_1 \vee \neg x_4 \vee x_7)$$

$$\omega_5 = (x_2 \vee \neg x_4 \vee \neg x_7)$$

$$x_1 = 0@1$$

$$x_2 = 0@2$$

$$x_3 = 0@3 \xrightarrow{w_3} x_6 = 1@3$$

Implication Graph

- Una gráfica de implicaciones mantiene la pista de por qué las variables son asignadas al valor que tienen.
- Cada variable asignada tiene un nodo.
- Las variables forzan a otra variable a tener cierto valor.

Ejemplo

$$\omega_1 = (x_1 \vee x_2 \vee x_4 \vee x_5) \text{ ✗}$$

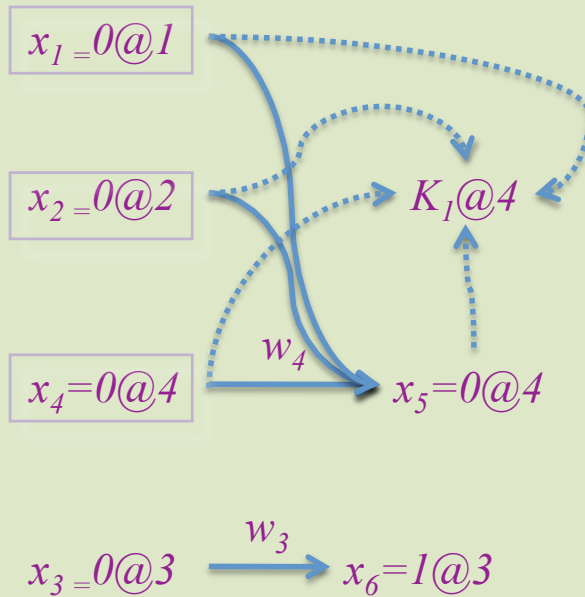
$$\omega_2 = (x_1 \vee x_2 \vee x_4 \vee \neg x_5)$$

$$\omega_3 = (\neg x_3 \vee x_6)$$

$$\omega_4 = (x_1 \vee \neg x_4 \vee x_7)$$

$$\omega_5 = (x_2 \vee \neg x_4 \vee \neg x_7)$$

$$\omega(k_1) = (x_1 \vee x_2 \vee x_4)$$



Cláusulas De Conflicto

- k_1 fue causada por $\neg x_1 \ \& \ \neg x_2 \ \& \ \neg x_4$
- Para evitar k_1 , forzamos a esta que no sea verdadera
- Agregamos una cláusula de conflicto

$$\begin{aligned}w(k_1) &= \neg(\neg x_1 \ \& \ \neg x_2 \ \& \ \neg x_4) \\ &= x_1 + x_2 + x_4\end{aligned}$$

Ejemplo

$$\omega_1 = (x_1 \vee x_2 \vee x_4 \vee x_5)$$

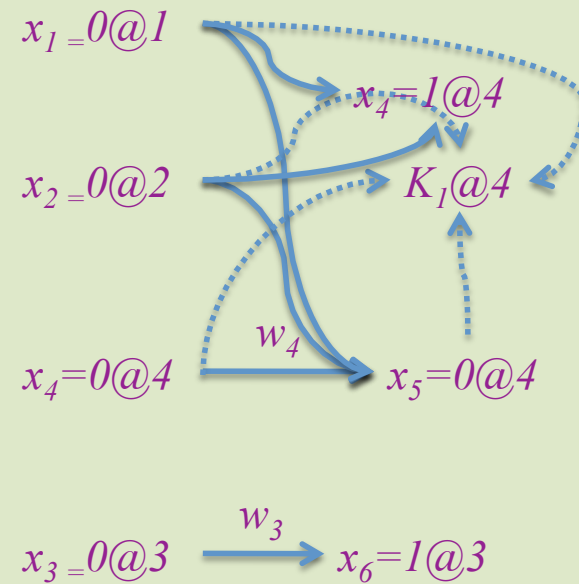
$$\omega_2 = (x_1 \vee x_2 \vee x_4 \vee \neg x_5)$$

$$\omega_3 = (\neg x_3 \vee x_6)$$

$$\omega_4 = (x_1 \vee \neg x_4 \vee x_7)$$

$$\omega_5 = (x_2 \vee \neg x_4 \vee \neg x_7)$$

$$\omega(k_1) = (x_1 \vee x_2 \vee x_4)$$



Ejemplo

$$\omega_1 = (x_1 \vee x_2 \vee x_4 \vee x_5)$$

$$\omega_2 = (x_1 \vee x_2 \vee x_4 \vee \neg x_5)$$

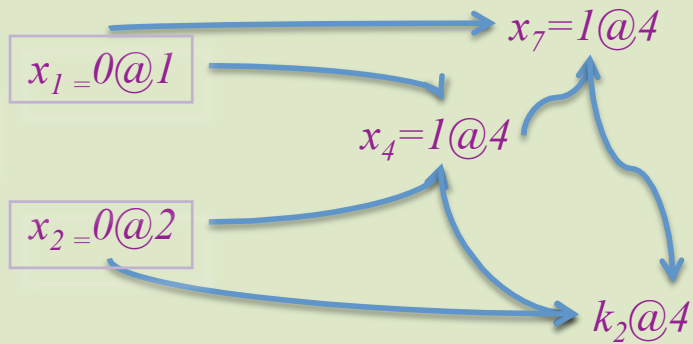
$$\omega_3 = (\neg x_3 \vee x_6)$$

$$\omega_4 = (x_1 \vee \neg x_4 \vee x_7)$$

$$\omega_5 = (x_2 \vee \neg x_4 \vee \neg x_7)$$

$$\omega(k_1) = (x_1 \vee x_2 \vee x_4)$$

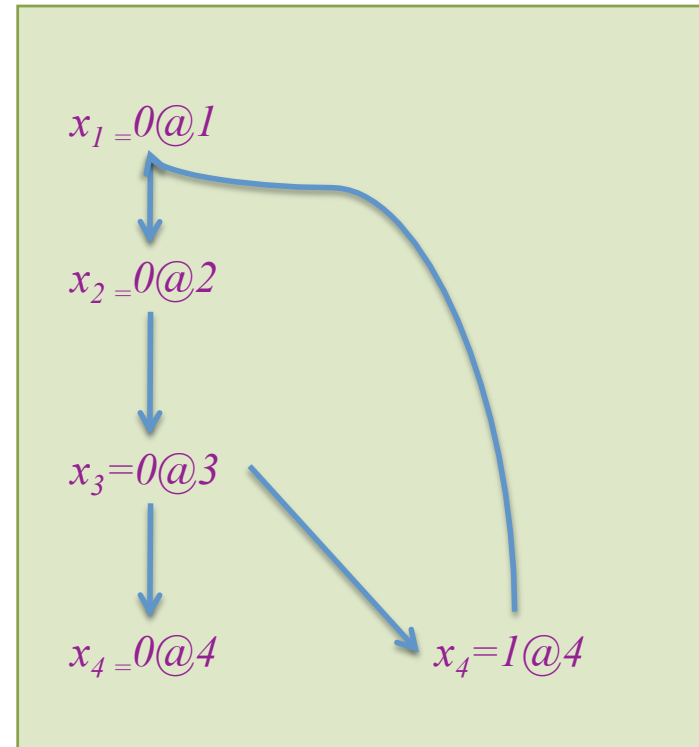
$$\omega(k_2) = (x_1 \vee x_2)$$



$$x_3=0@3 \xrightarrow{w_3} x_6=1@3$$

Non-Chronological Backtracking

- GRASP es non-chronological backtracking: saltará puntos de asignación hasta el punto de decisión del árbol que afecta la asignación.



Non-Chronological Backtracking

- GRASP permite saber que asignaciones no afectan en el conflicto.
- GRASP salta a la última asignación que afecta el conflicto.

$$\omega_1 = (x_1 \vee x_2 \vee x_4 \vee x_5)$$

$$\omega_2 = (x_1 \vee x_2 \vee x_4 \vee \neg x_5)$$

$$\omega_3 = (\neg x_3 \vee x_6)$$

$$\omega_4 = (x_1 \vee \neg x_4 \vee x_7)$$

$$\omega_5 = (x_2 \vee \neg x_4 \vee \neg x_7)$$

$$\omega(k_1) = (x_1 \vee x_2 \vee x_4)$$

$$\omega(k_2) = (x_1 \vee x_2)$$

$$x_1 = 0@1$$

$$x_2 = 1@2$$

$$x_4 = 0@4$$

$$x_3 = 0@3$$

$$x_6 = 1@3$$

$$x_6 = 1@3$$

$$x_6 = 1@3$$

$$x_6 = 1@3$$

$$x_1 = 0@1$$

$$x_2 = 1@2$$

$$x_4 = 0@4$$

$$x_3 = 0@3 \xrightarrow{w_3} x_6 = 1@3$$

Davis-Putnam Algorithm

- Algoritmo recursivo que realiza una búsqueda de árbol haciendo asignaciones a las variables restantes en cada etapa del algoritmo
- Tiene un mejor rendimiento en promedio que una búsqueda exhaustiva

Davis-Putnam Algorithm

- Ejemplo del algoritmo Davis-Putnam:

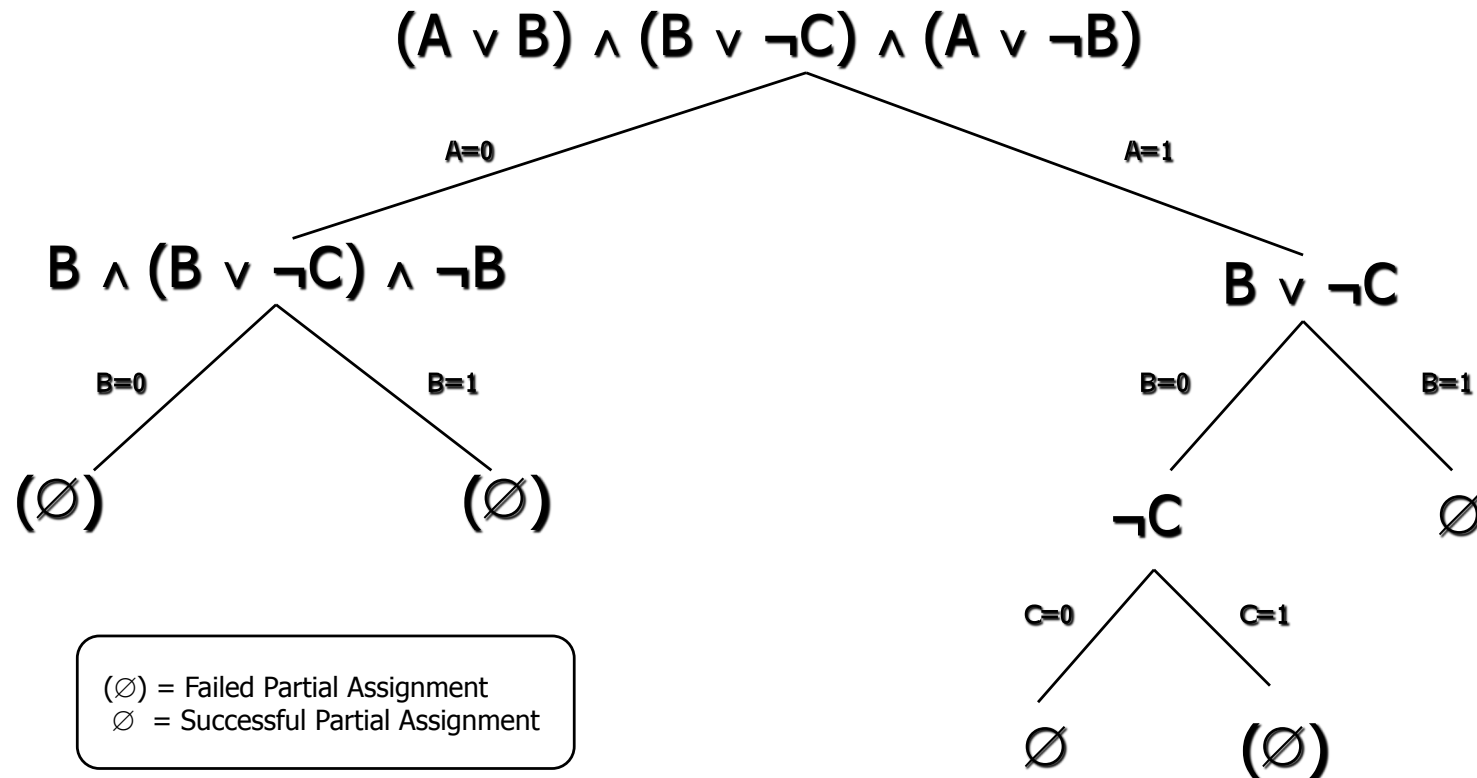
$$(A \vee B) \wedge (B \vee \neg C) \wedge (A \vee \neg B)$$

Davis-Putnam Algorithm:

```
procedure split(E) {  
    if E has an empty clause, then return  
    if E has no clauses, then exit with current partial assignment  
    select next unassigned variable,  $x_i$  in E  
    split(E( $x_i=0$ ))  
    split(E( $x_i=1$ ))  
}
```

Algoritmo Davis-Putnam

Aplicación a: $(A \vee B) \wedge (B \vee \neg C) \wedge (A \vee \neg B)$



Algoritmo Davis-Putnam

- Desafortunadamente, en el peor de los casos el algoritmo Davis-Putnam es exponencial
- Actualmente no hay solución que provea un mejor rendimiento en el peor de los casos que no sea exponencial.

Algoritmo Davis-Putnam

- Resolvedores SAT como Davis-Putnam modifican asignaciones hechas en reserva del orden en el cual son realizadas y siempre realiza la misma asignación.