Wiki individual de Sistemas Inteligentes

Wiki individual para seguimiento de clases de teoría de Sistemas Inteligentes

Sesión 5-Tema 3: Forward Checking, propagación de restricciones y algoritmo AC3

Tema 3: Forward Checking, propagación de restricciones y algoritmo AC3 – Resumen

Cómo ya comentamos la semana pasada, forward checking es una versión de backtraking que mira las posibles elecciones que podemos hacer, comprueba la inconsistencia de estas elecciones en las variables según su dominio (Si una variable se queda sin dominio, esa primera elección no tiene solución) y si no hay problema con ninguna, devuelve la solución.

Cuando genera inconsistencia, retrocede una posición de la inicial y vuelve a realizar la comprobación con los posibles valores futuros. Si encuentra inconsistencias y retrocede más de la primera variable, no tiene solución.

La inconsistencia se muestra cuando, por culpa de las restricciones del problema, el dominio de los siguientes valores se ven acotados por esa restricción, en el momento que una variable se queda con el dominio vació, es decir, no tiene ningún valor posible por la restricción, es que el valor inicial está equivocado y hay que probar con uno anterior.

El problema de la generación de inconsistencias en problemas CSP (Problemas de satisfacción de restricciones), se puede transformar en un problema más sencillo sin inconsistencias de arco, para ello definimos la propiedad de consistencia de arista:

"Una arista dirigida c(ep) = < Vi, Vj > es consistente si y sólo si para

todo valor asignable a Vi existe al menos un valor en Vj que

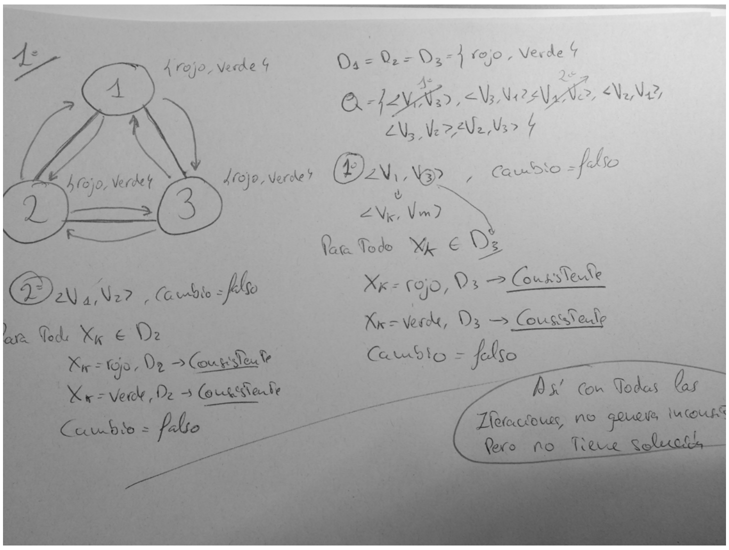
satisface la restricción asociada a la arista."

Para la reducción a un problema más sencillo sin inconsistencias, existe el algoritmo AC3, el cual examina todas las aristas entre dos variables y elimina aquellas que puedan generar inconsistencias. De este modo, al finalizar el algoritmo, sabemos con certeza una de las siguientes afirmaciones:

- No tiene solución
- Hay una única solución
- Tiene muchas soluciones

A continuación, se mostrará la traza de 3 problemas similares a los que aplicamos el algoritmo AC3 para eliminar inconsistencias, mostrando el resultado final. El problema es el mapeado de colores, si tienes 2 o más nodos conectados entre sí, 2 no pueden tener el mismo color si están juntos.

Traza problema 1:



1º

{rojo, verde} {rojo, verde} {rojo, verde}

$D_1 = D_2 = D_3 = \{rojo, verde\}$

$Q = \{\cancel{\langle V_1, V_3\rangle}, \langle V_3, V_1\rangle, \cancel{\langle V_1, V_2\rangle}, \langle V_2, V_1\rangle,$
$\langle V_3, V_2\rangle, \langle V_2, V_3\rangle\}$

1º $\langle V_1, V_3\rangle$, cambio = falso

$\langle V_k, V_m\rangle$

Para Todo $X_k \in D_3$

X_k = rojo, D_3 → Consistente

X_k = verde, D_3 → Consistente

Cambio = falso

2º $\langle V_1, V_2\rangle$, Cambio = falso

Para Todo $X_k \in D_2$

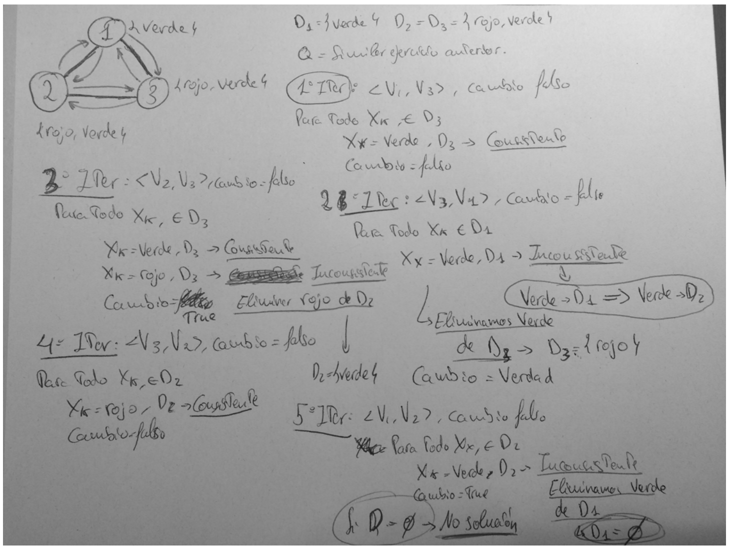
X_k = rojo, D_2 → Consistente

X_k = verde, D_2 → Consistente

Cambio = falso

Así con todas las Iteraciones, no genera inconsis. Pero no tiene solución

Traza problema 2:



{verde} {rojo, verde} {rojo, verde}

$D_1 = \{verde\}$ $D_2 = D_3 = \{rojo, verde\}$

Q = Similar ejercicio anterior.

1º ITer: $\langle V_1, V_3\rangle$, cambio falso

Para Todo $X_k \in D_3$

X_k = verde, D_3 → Consistente

Cambio = falso

2º ITer: $\langle V_3, V_1\rangle$, cambio = falso

Para Todo $X_k \in D_1$

X_x = Verde, D_1 → Inconsistente

Verde → D_1 ⟹ Verde → D_2

↳ Eliminamos Verde de D_3 → $D_3 = \{rojo\}$

Cambio = Verdad

3º ITer: $\langle V_2, V_3\rangle$, cambio = falso

Para Todo $X_k \in D_3$

X_k = Verde, D_3 → Consistente

X_k = rojo, D_3 → ~~Consistente~~ Inconsistente

Cambio = ~~falso~~ True

Eliminar rojo de D_2

$D_2 = \{verde\}$

4º ITer: $\langle V_3, V_2\rangle$, cambio = falso

Para Todo $X_k \in D_2$

X_k = rojo, D_2 → Consistente

Cambio = falso

5º ITer: $\langle V_1, V_2\rangle$, cambio falso

Para Todo $X_k \in D_2$

X_k = Verde, D_2 → Inconsistente

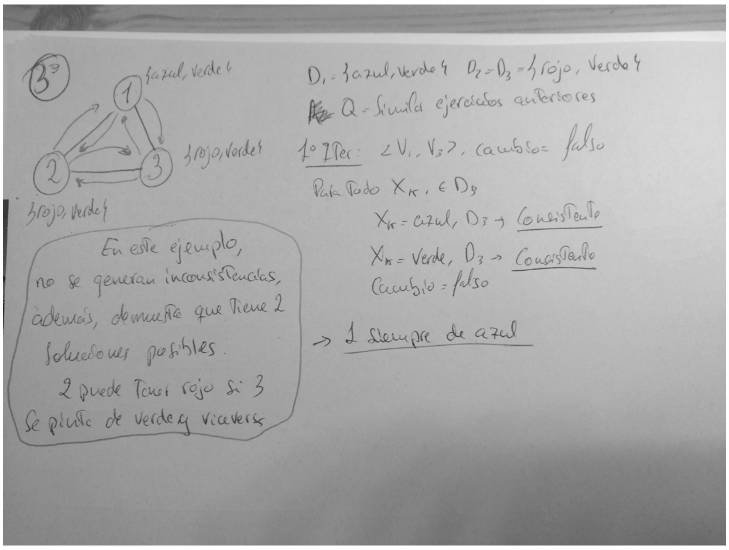
Cambio = True

Eliminamos Verde de D_1

$D_1 = \emptyset$

Si $D = \emptyset$ → No solución

Traza problema 3:



3º

{azul, verde} {rojo, verde} {rojo, verde}

$D_1 = \{azul, verde\}$ $D_2 = D_3 = \{rojo, verde\}$

Q = Similar ejercicios anteriores

1º ITer: $\langle V_1, V_2\rangle$, cambio = falso

Para Todo $X_k \in D_3$

X_k = azul, D_3 → Consistente

X_k = Verde, D_3 → Consistente

Cambio = falso

→ 1 siempre de azul

En este ejemplo, no se generan inconsistencias, además, demuestra que tiene 2 soluciones posibles. 2 puede tomar rojo si 3 se pinta de verde y viceversa

AC-3 algorithm

Índice WIki individual

◄ Resolución de algunos problemas con la Wiki

Ir a...

Wiki pruebas ►