
Inteligencia Artificial

Problemas de Satisfacción de Restricciones

Jorge Luis Guevara Diaz

www.jorge.sistemasyservidores.com

Que veremos?

- PSR ejemplos
 - Búsqueda Backtraking para PSR
-

Problemas de satisfacción de restricciones

- Problema de búsqueda estandar: □
 - **estado** es una “caja negra” – cualquier estructura de dato que soporte función sucesor, función heurística, y test meta
 - Que es PSR?
 - Conjunto finito de variables X_1, X_2, \dots, X_n
 - Conjunto finito de restricciones C_1, C_2, \dots, C_m
 - Dominio no vacío de posible valores para cada variable $D_{V1}, D_{V2}, \dots, D_{Vn}$
 - Cada restricción C_i limita los valores que cada variable puede tener, ejemplo: $X_1 \neq X_2$
-

Problemas de satisfacción de restricciones

- PSR:
 - Estado se define por variables X_i con valores de un dominio D_i
 - Test meta es un conjunto de restricciones especificando las combinaciones permitidas de valores para subconjuntos de variables
 - Permitiendo algoritmos de propósito general con más poder que los estándar algoritmos de búsqueda
-

Problemas de satisfacción de restricciones

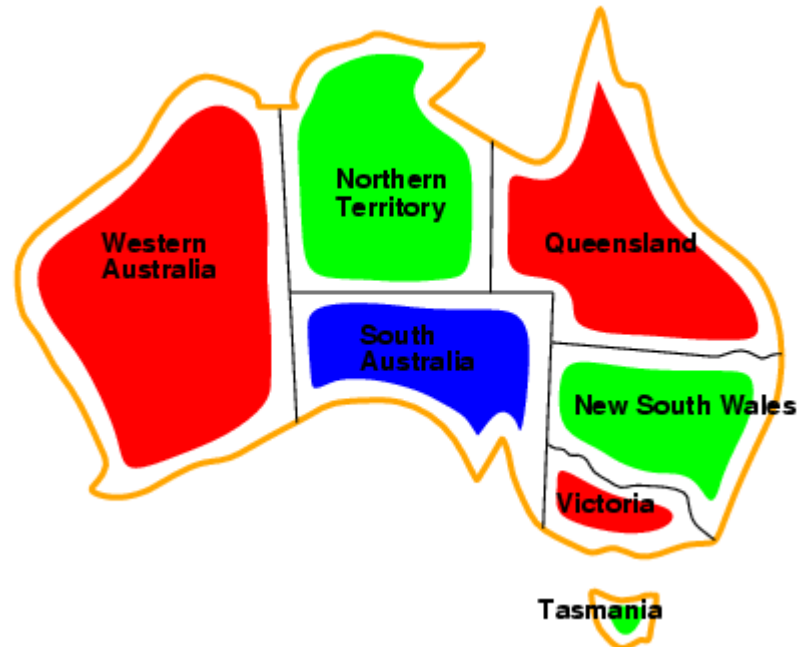
Ejemplo : coloracion de mapas



- Variables WA, NT, Q, NSW, V, SA, T
 - Dominio $D_i = \{\text{rojo, verde, azul}\}$
 - Restricciones: regiones adyacentes deben tener diferentes colores
 - e.g., $WA \neq NT$, o (WA, NT) en $\{(\text{rojo, verde}), (\text{rojo, azul}), (\text{verde, rojo}), (\text{verde, azul}), (\text{azul, rojo}), (\text{azul, verde})\}$
-

Problemas de satisfacción de restricciones

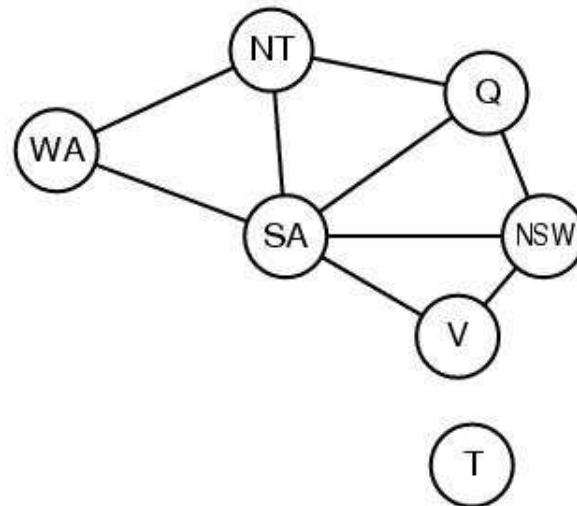
Ejemplo : coloracion de mapas



- Soluciones son **completas** y con asignaciones **consistentes**, e.g., WA = rojo, NT = verde, Q = rojo, NSW = verde, V = rojo, SA = azul, T = verde
-

Problemas de satisfacción de restricciones

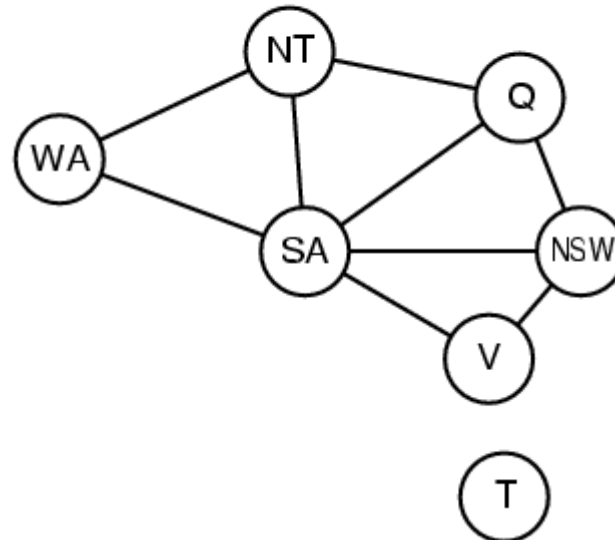
- PSR beneficios
 - Patron de representation estandar
 - Función sucesor y Meta genéricos
 - Heurísticas genéricas .



Problemas de satisfacción de restricciones

Grafo de Restricciones

- **PSR binario:** Cada restricción relaciona dos variables
- **Grafo de Restricciones:** los nodos son las variables, los arcos son las restricciones



Problemas de satisfacción de restricciones

Tipos de PSR

- Variables discretas
 - Dominios finitos; tamaño $d \Rightarrow O(d^n)$ asignaciones completas.
 - Dominios infinitos (enteros, cadenas, etc.)
.
- Variables continuas



Problemas de satisfacción de restricciones

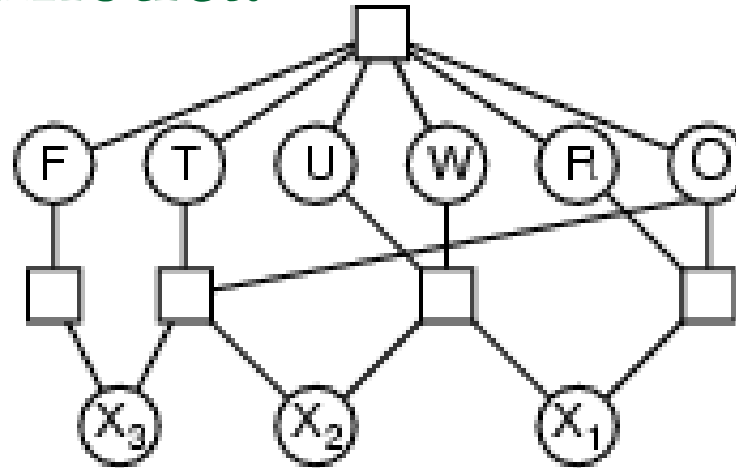
Tipos de restricciones

- Restricciones unarias involucran una sola variable
 - e.g., $SA \neq \text{verde}$
 - Restricciones binarias involucran pares de variables
 - e.g., $SA \neq WA$ □
 - Restricciones de Alto-Orden involucran tres o mas variables
 - criptoaritmética
-

Problemas de satisfacción de restricciones

Ejemplo criptoaritmética

$$\begin{array}{r} \text{ T W O} \\ + \text{ T W O} \\ \hline \text{ F O U R} \end{array}$$



- **Variables:** $F T U W$
 $R O X_1 X_2 X_3$
- **Dominio:** $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$
- **Restriciones:** $\text{TodosDiferentes}(F, T, U, W, R, O)$
 - $O + O = R + 10 \cdot X_1$
 - $X_1 + W + W = U + 10 \cdot X_2$
 - $X_2 + T + T = O + 10 \cdot X_3$
 - $X_3 = F,$

Problemas de satisfacción de restricciones

Problemas reales

- Problemas de asignacion
 - Quien enseña determinada clase?
 - Problemas de Horarios (calendario) □
 - Que clase es ofertada, cuando y donde? □
 - Problemas de transporte
 - Planificación de fabricas, etc
-

Problemas de satisfacción de restricciones

Formulación estándar

Formulación incremental

- *Estado inicial*: sin asignaciones $\{\}$.
 - *Función sucesor*: asignar valores a a variables sin asignar mientras no haya conflicto
 - *Test meta*: si la asignacion actual es completa
 - *Costo del camino*: costo constante para cada paso
-

Problemas de satisfacción de restricciones

Formulación estándar

- Esta formulacion es la **misma** para todos los PSR!!!
 - La solucion es encotrada a **profundidad n** (si existen n variables).
 - Puede ser usado **búsqueda primero en profundidad**.
 - Camino es **irrelevante**, Se pueden utilizar representacion completa de estado.
 - Factor de ramificacion **b** en el nivel superior es **nd** .
 - **$b=(n-l)d$** a profundidad **l** , por lo tanto **$n!d^n$** hojas (pero solamente se tiene **d^n** asignaciones completas).
-

Problemas de satisfacción de restricciones

Conmutatividad

PSR son conmutativos.

– Ejemplo:

$[WA=rojo, NT=verde] = [NT=verde, WA=rojo]$

- Todos los algoritmos de búsqueda PSR consideran la asignación de una sola variable a la vez \Rightarrow entonces existen d^n hojas.



Problemas de satisfacción de restricciones

Problem	Backtracking	BT+MRV	Forward Checking	FC+MRV	Min-Conflicts
USA	(> 1,000K)	(> 1,000K)	2K	60	64
n-Queens	(> 40,000K)	13,500K	(> 40,000K)	817K	4K
Zebra	3,859K	1K	35K	0.5K	2K
Random 1	415K	3K	26K	2K	
Random 2	942K	27K	77K	15K	

Problemas de satisfacción de restricciones

Búsqueda Backtracking

- Búsqueda primero en profundidad para PSR con asignaciones simples de variables es llamada Búsqueda **Backtracking**
 - Búsqueda Backtracking el esl algoritmo básico sin informacion para for CSPs
-

Problemas de satisfacción de restricciones

Búsqueda Backtracking Ejemplo



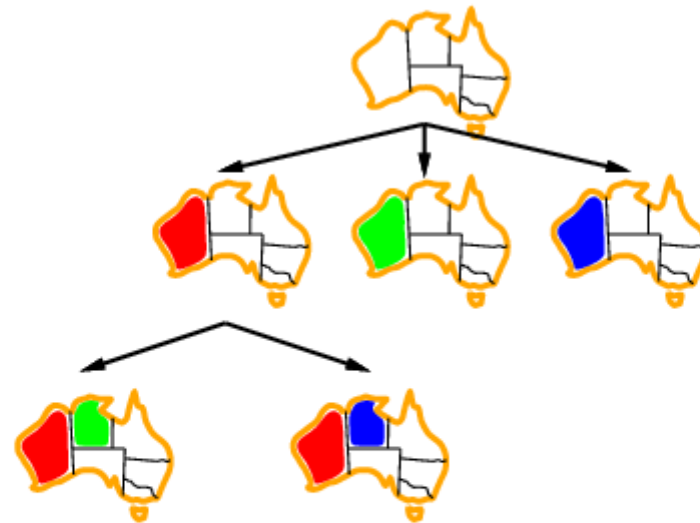
Problemas de satisfacción de restricciones

Búsqueda Backtracking Ejemplo



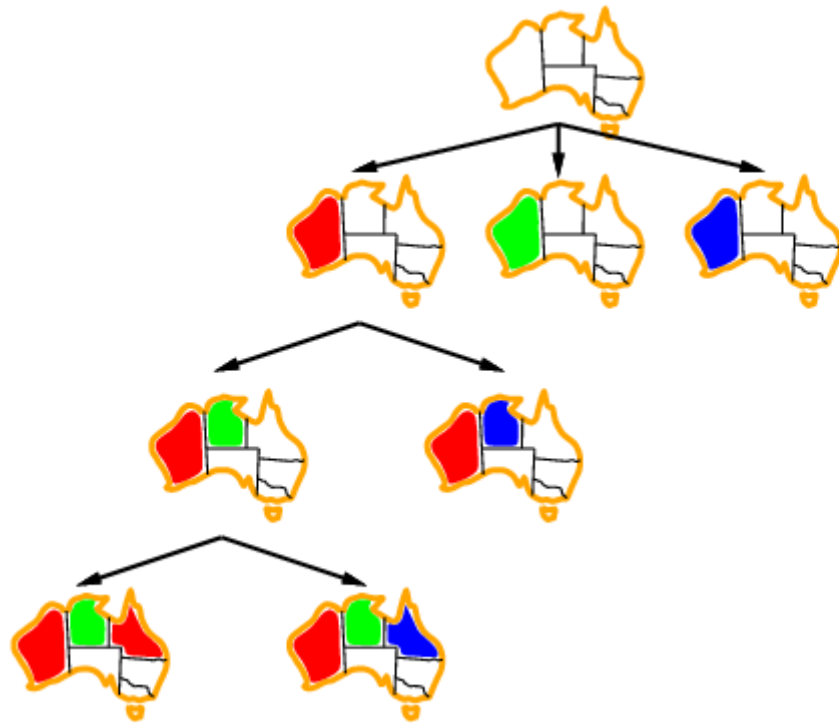
Problemas de satisfacción de restricciones

Búsqueda Backtracking Ejemplo



Problemas de satisfacción de restricciones

Búsqueda Backtracking Ejemplo



Problemas de satisfacción de restricciones

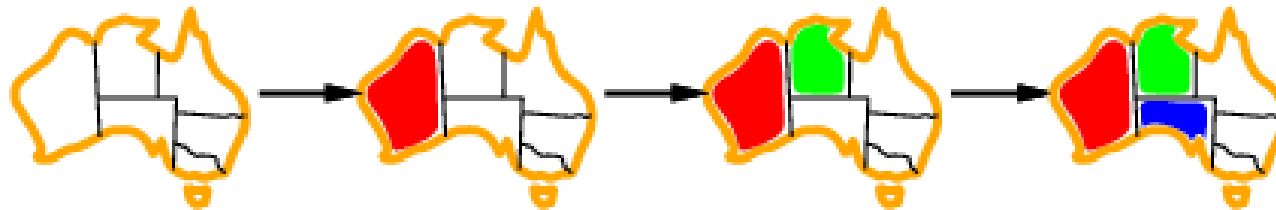
Backtracking Mejorando la eficiencia

- ❑ Que variable debería asignarse después? ❑
 - ❑ En que orden deberían ser probados los valores? ❑
 - ❑ Se pueden detectar fallas tempranamente? ❑
-

Problemas de satisfacción de restricciones

Backtracking: Heurística MVR

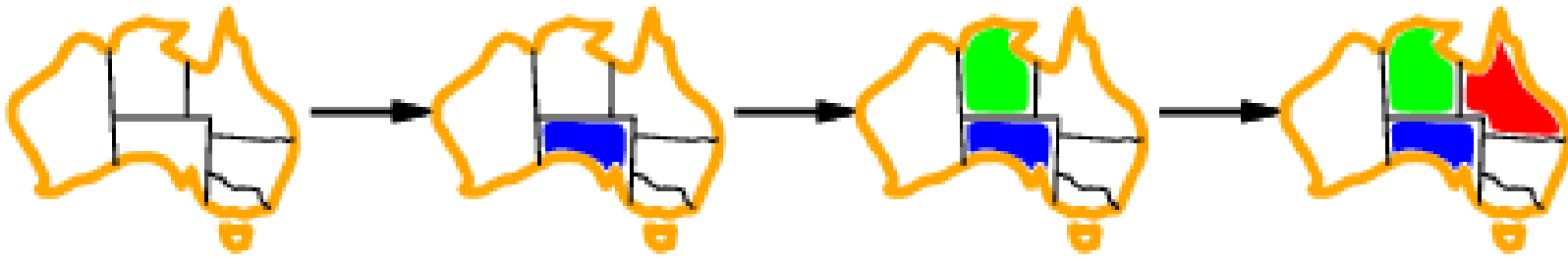
- Escoge la variable más restringida:
Es decir aquella con menos valores legales □



Problemas de satisfacción de restricciones

Backtracking: Heurística de Grado

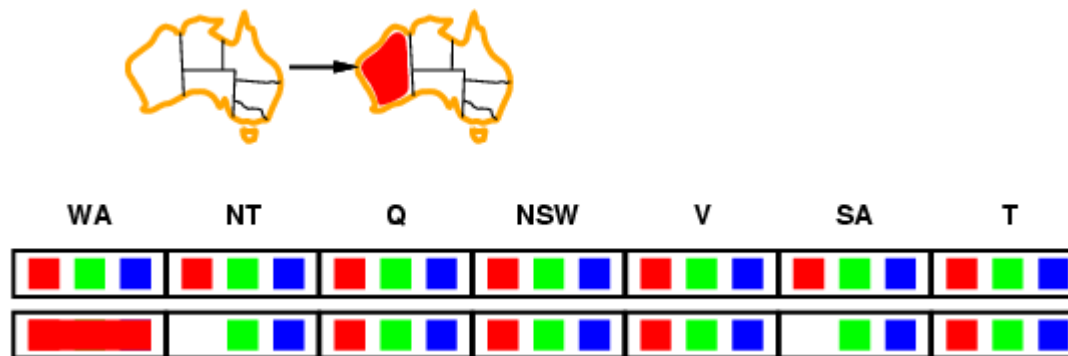
- Escoge las variables con más restricciones



Problemas de satisfacción de restricciones

Propagacion de informacion

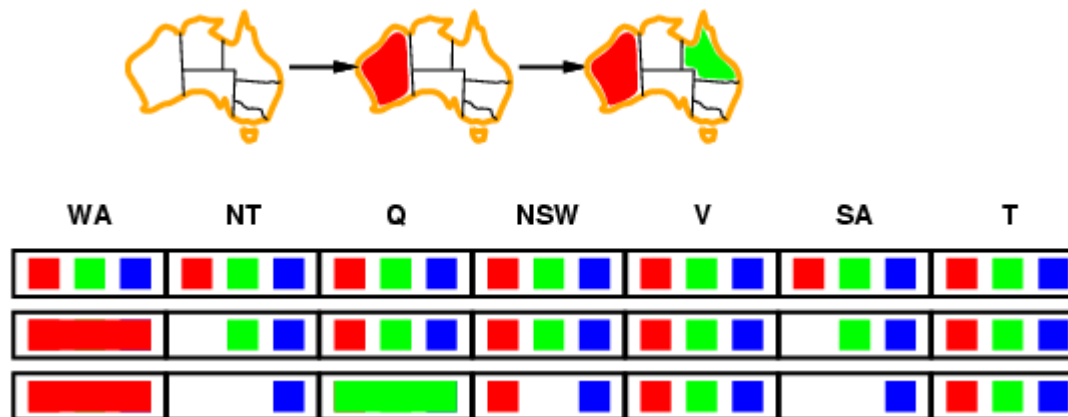
- Forward checking



Problemas de satisfacción de restricciones

Propagacion de informacion

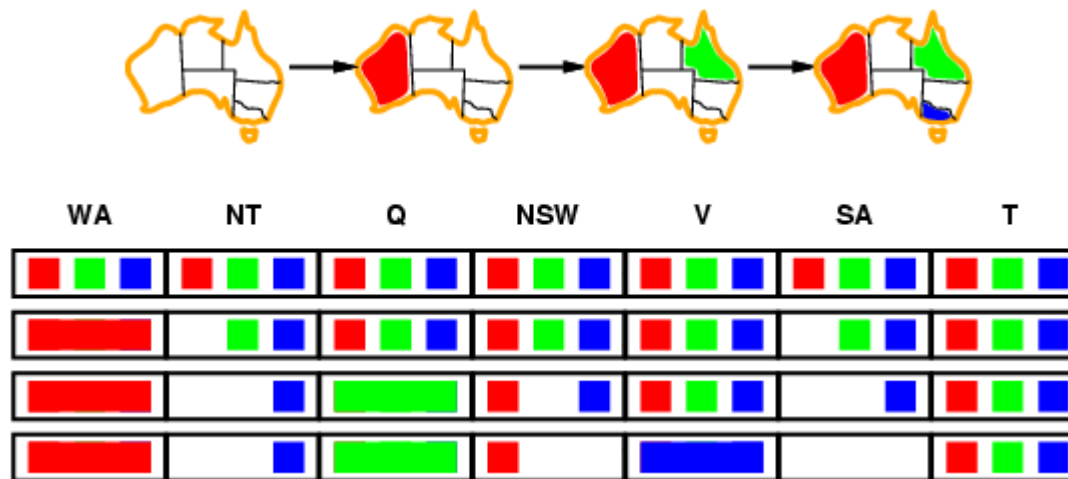
Forward checking



Problemas de satisfacción de restricciones

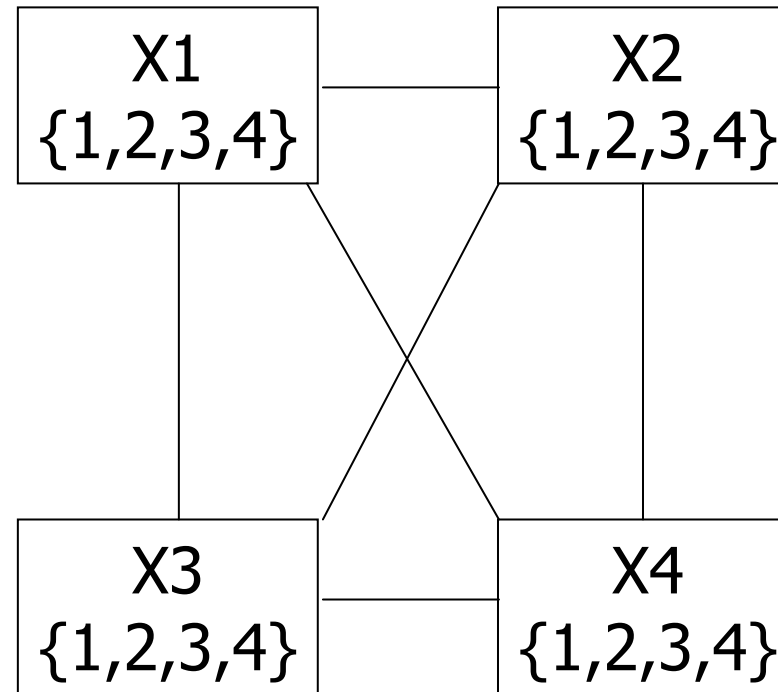
Propagacion de informacion

Forward checking



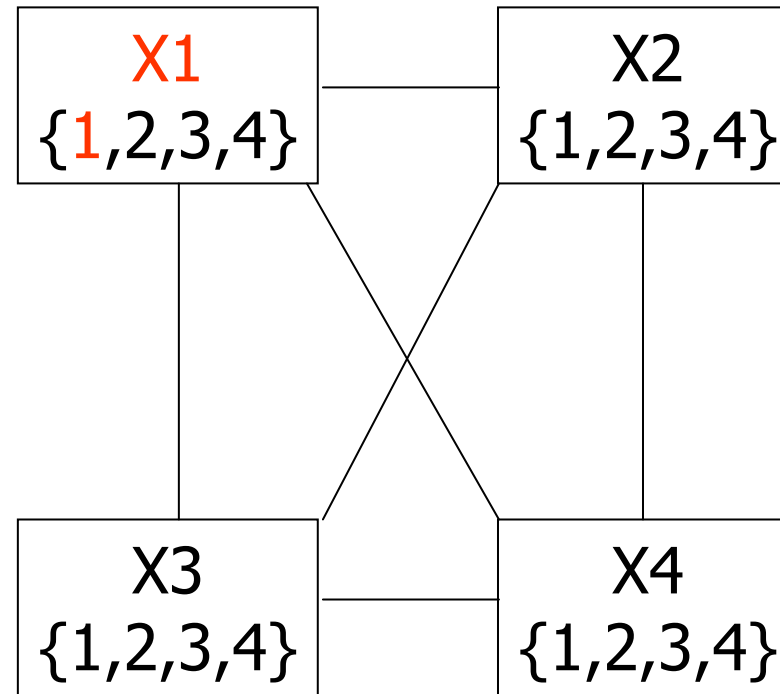
Ejemplo: Problema de las 4-Reynas

	1	2	3	4
1				
2				
3				
4				



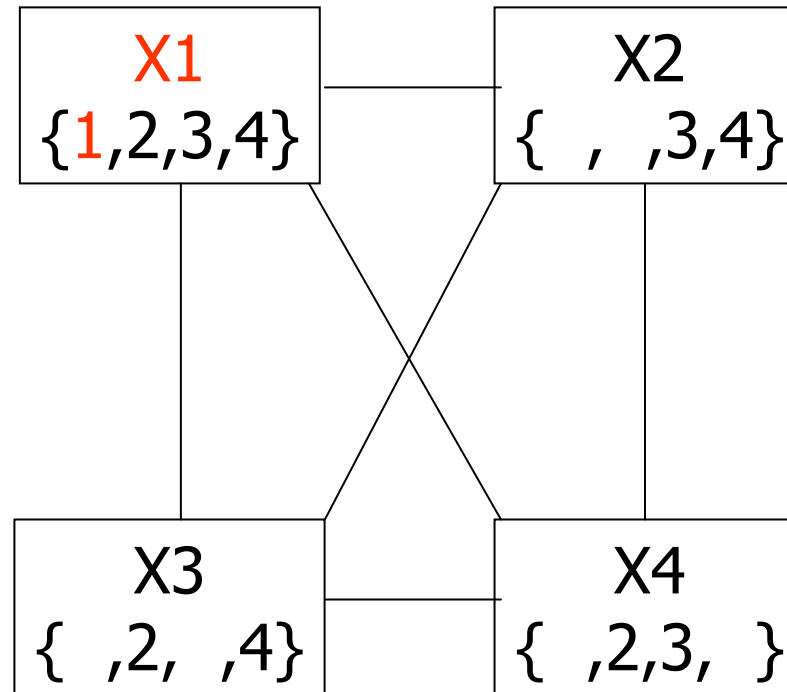
Ejemplo: Problema de las 4-Reynas

	1	2	3	4
1	★	●	●	●
2	■	●	■	□
3	□	■	●	■
4	■	□	■	●



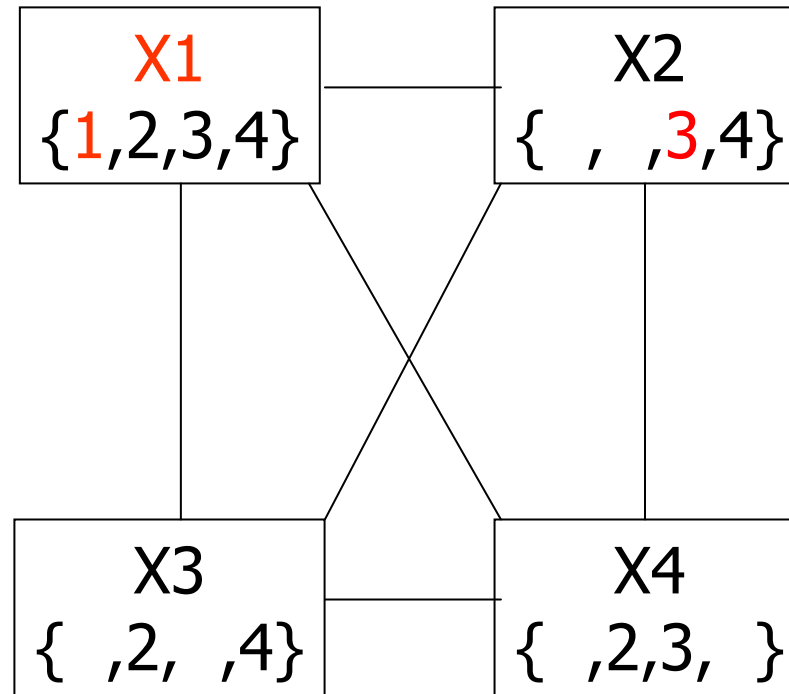
Ejemplo: Problema de las 4-Reynas

	1	2	3	4
1	★	●	●	●
2	■	●	■	□
3	□	■	●	■
4	■	□	■	●



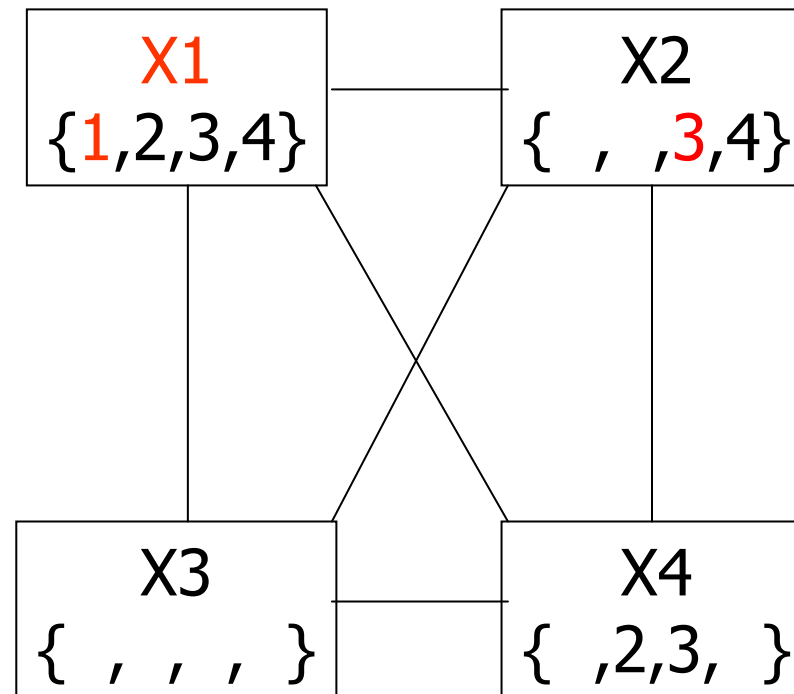
Ejemplo: Problema de las 4-Reynas

	1	2	3	4
1	★	●	●	●
2	■	●	●	□
3	□	★	●	●
4	■	□	●	●



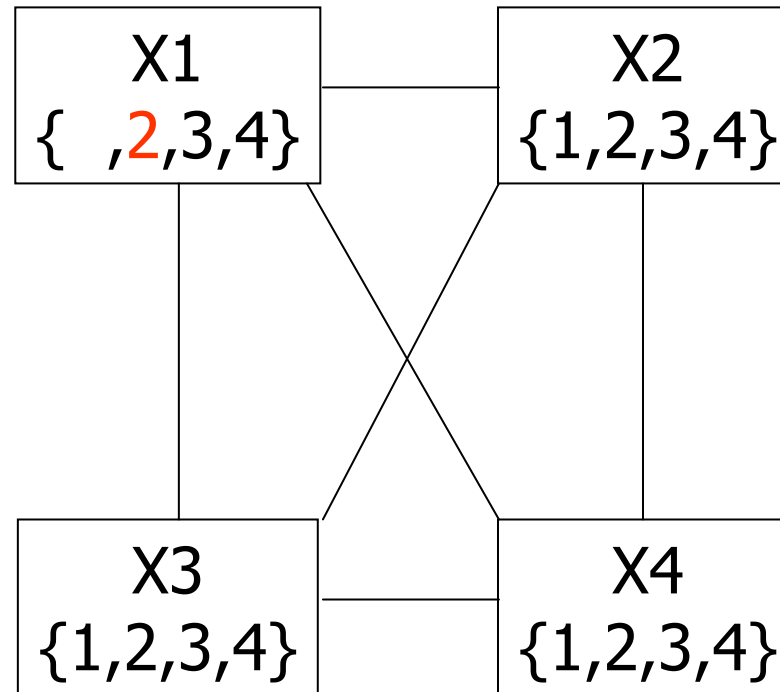
Ejemplo: Problema de las 4-Reynas

	1	2	3	4
1	★	●	●	●
2	■	●	●	□
3	□	★	●	●
4	■	□	●	●



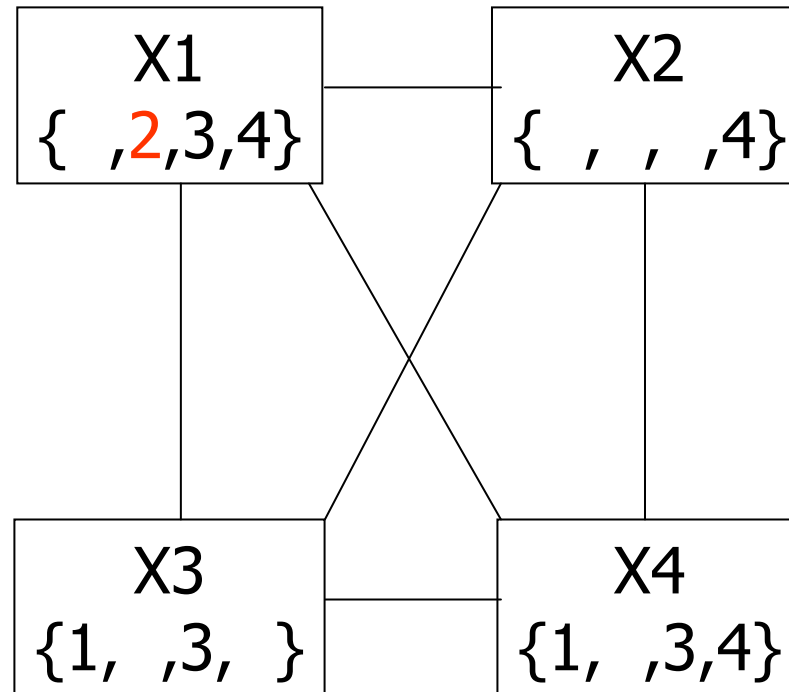
Ejemplo: Problema de las 4-Reynas

	1	2	3	4
1		●		■
2	★	●	●	●
3		●		■
4	■		●	



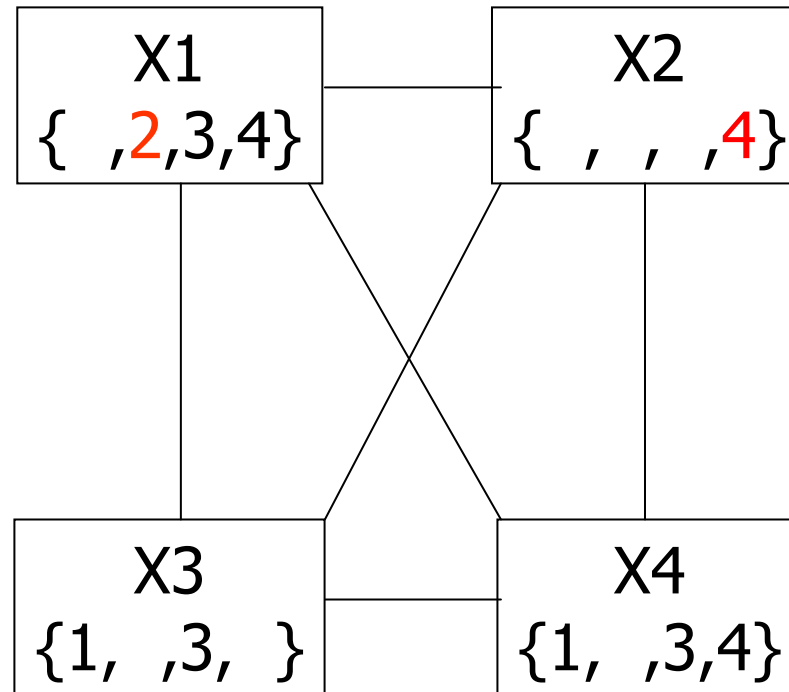
Ejemplo: Problema de las 4-Reynas

	1	2	3	4
1		●		■
2	★	●	●	●
3		●		■
4	■		●	



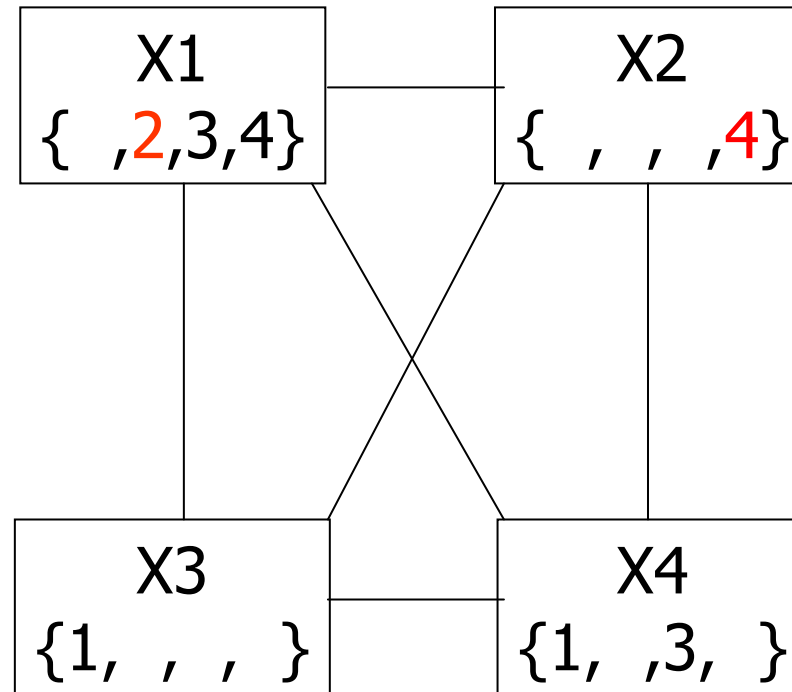
Ejemplo: Problema de las 4-Reynas

	1	2	3	4
1		●		■
2	★	●	●	●
3		●	●	■
4	■	★	●	●



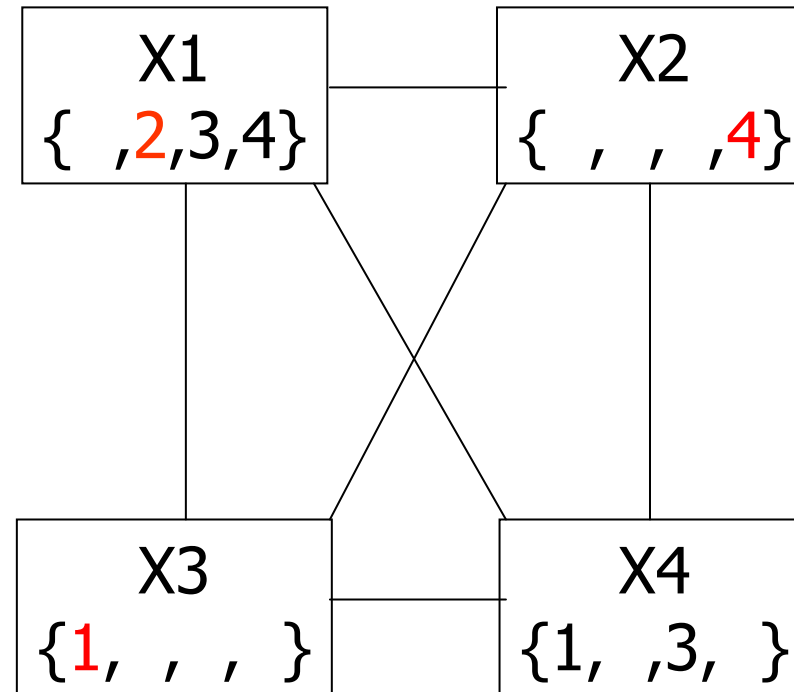
Ejemplo: Problema de las 4-Reynas

	1	2	3	4
1		●		■
2	★	●	●	●
3		●	●	■
4	■	★	●	●



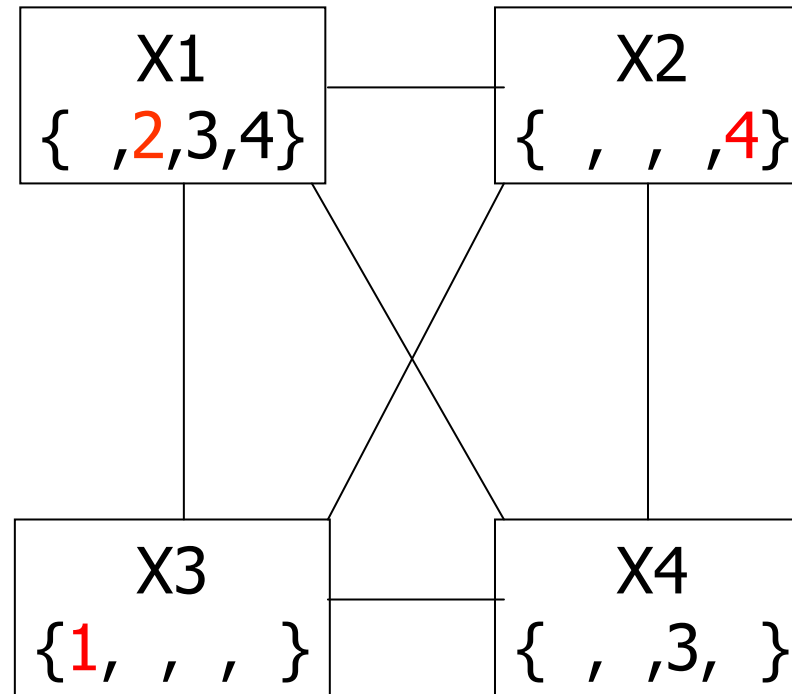
Ejemplo: Problema de las 4-Reynas

	1	2	3	4
1		●	★	●
2	★	●	●	●
3		●	●	
4		★	●	●



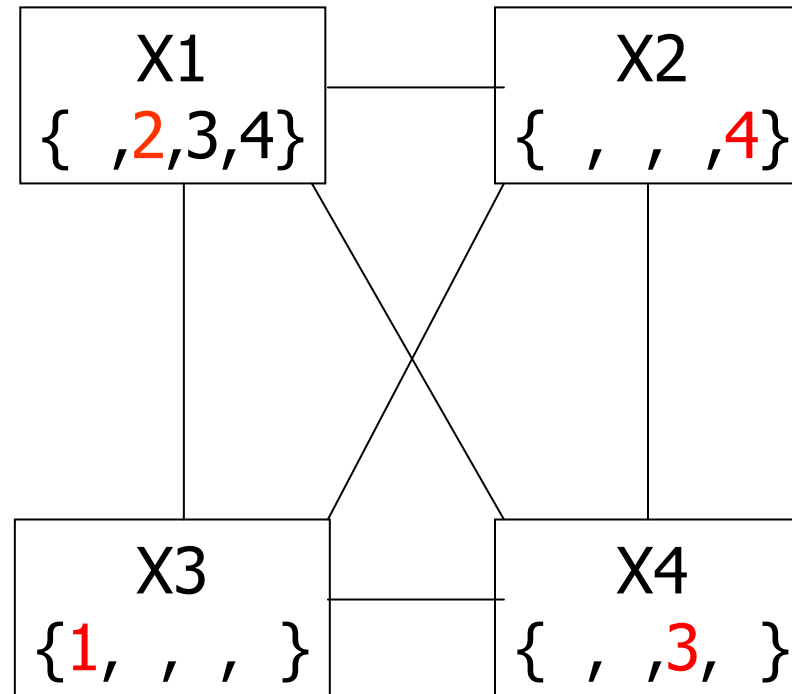
Ejemplo: Problema de las 4-Reynas

	1	2	3	4
1		●	★	●
2	★	●	●	●
3		●	●	
4		★	●	●



Ejemplo: Problema de las 4-Reynas

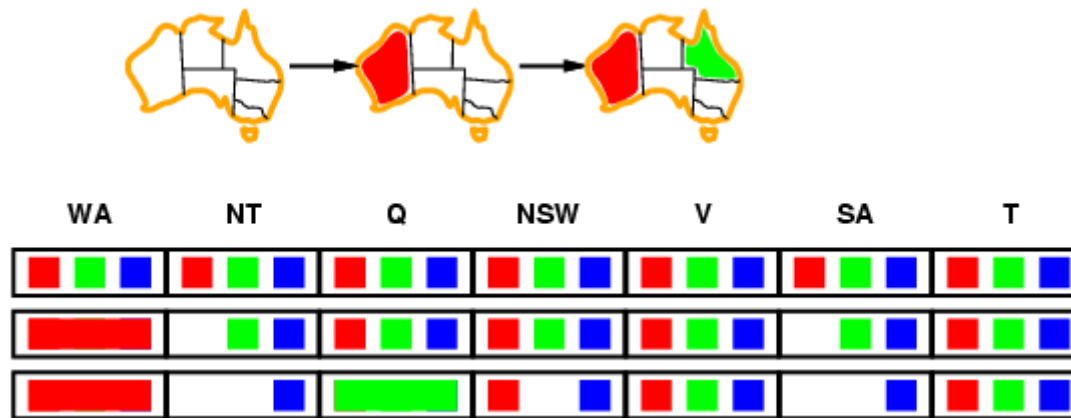
	1	2	3	4
1		●	★	●
2	★	●	●	●
3		●	●	★
4		★	●	●



Problemas de satisfacción de restricciones

Propagación de Restricciones

- Forward checking propaga la información de variables asignadas a variables sin asignar, pero no provee detección temprana para todas las fallas

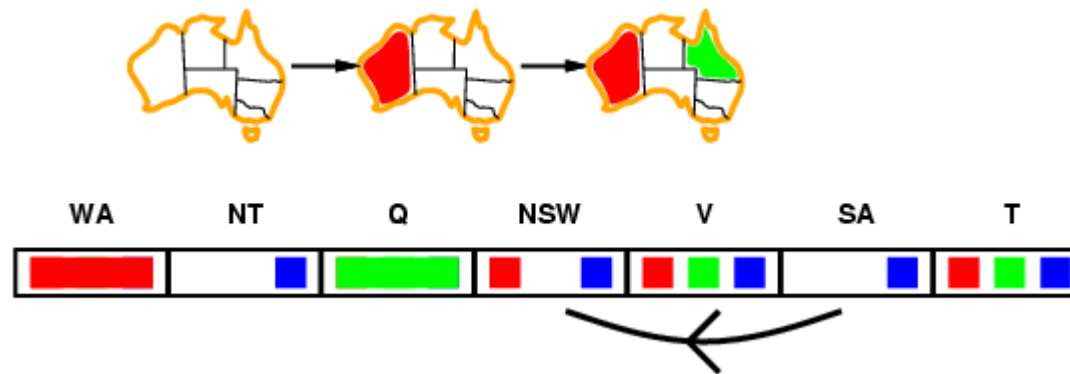


- NT y SA No pueden ser ambas azules
- Propagación de restricciones impone restricciones a nivel local de manera repetida

Problemas de satisfacción de restricciones

Propagación de Restricciones

- Consistencia de Arcos
- Forma simple de propagación hace de cada arco consistente

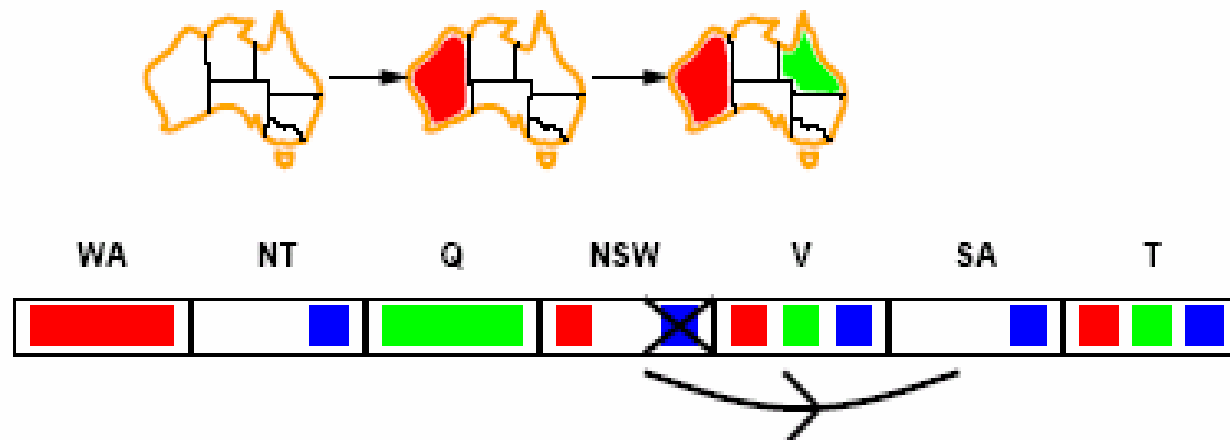


$X \rightarrow Y$ es consistente sii
para *cada* valor x de X existe algún valor permitido y en Y
 $SA \rightarrow NSW$ es consistente sii
 $SA=azul$ y $NSW=rojo$

Problemas de satisfacción de restricciones

Propagación de Restricciones

■ Consistencia de Arcos



$X \rightarrow Y$ es consistente sii

para *cada* valor x de X existe algún valor permitido y en Y

- $NSW \rightarrow SA$ es consistente sii

$NSW=rojo$ y $SA=azul$

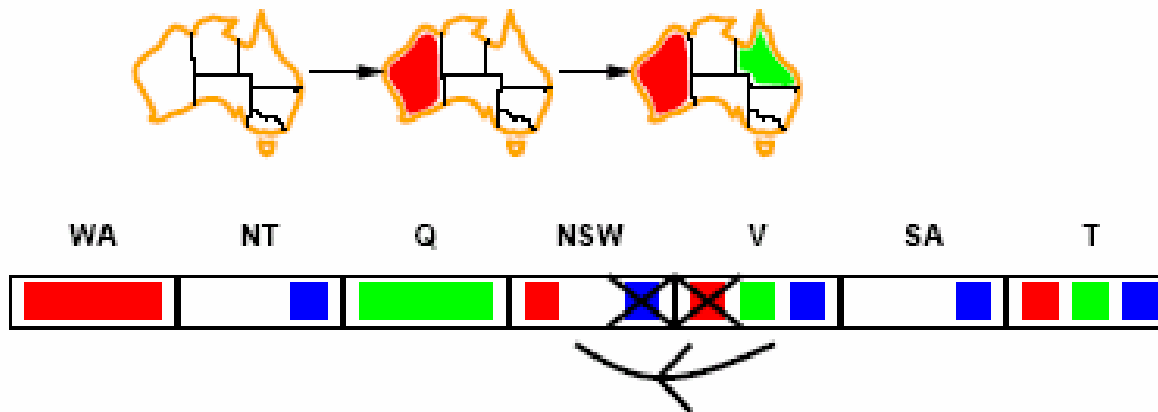
$NSW=azul$ y $SA=???$

EL arco puede ser consistente removiendo *azul* de NSW

Problemas de satisfacción de restricciones

Propagación de Restricciones

■ Consistencia de Arcos



- si X pierde un valor, los vecinos de X se tienen que volver a chequear
- VOLVER A CHEKEAR todos los vecinos !!
 - Remover rojo de V

Problemas de satisfacción de restricciones

Propagación de Restricciones

```
function AC-3(csp) returns the CSP, possibly with reduced domains
inputs: csp, a binary CSP with variables  $\{X_1, X_2, \dots, X_n\}$ 
local variables: queue, a queue of arcs, initially all the arcs in csp

while queue is not empty do
   $(X_i, X_j) \leftarrow \text{REMOVE-FIRST}(\textit{queue})$ 
  if REMOVE-INCONSISTENT-VALUES( $X_i, X_j$ ) then
    for each  $X_k$  in NEIGHBORS[ $X_i$ ] do
      add  $(X_k, X_i)$  to queue
```

```
function REMOVE-INCONSISTENT-VALUES( $X_i, X_j$ ) returns true iff succeeds
  removed  $\leftarrow$  false
  for each  $x$  in DOMAIN[ $X_i$ ] do
    if no value  $y$  in DOMAIN[ $X_j$ ] allows  $(x, y)$  to satisfy the constraint  $X_i \leftrightarrow X_j$ 
      then delete  $x$  from DOMAIN[ $X_i$ ]; removed  $\leftarrow$  true
  return removed
```

Búsqueda Local para PSR

- Hill-climbing, simulated annealing trabajan típicamente con estados “completos” es decir todas las variables están asignadas
 - Aplicando PSR:
 - Permitir estados con restricciones sin satisfacer
 - Los operadores **reassignan** valores a las variables
 - Selección de variables: aleatoriamente seleccionar cualquier variable conflictiva
 - Los valores se seleccionan mediante la heurística **conflictos-minimos** :
 - Escoger el valor que viola menos restricciones
 - i.e., hill-climb con $h(n)$ = número total de restricciones violadas
-

Heurística Conflictos Mínimos

function MIN-CONFLICTS(*csp*, *max-steps*) **returns** a solution or failure

inputs: *csp*, a constraint satisfaction problem

max-steps, the number of steps allowed before giving up

current \leftarrow an initial complete assignment for *csp*

for $i = 1$ to *max-steps* **do**

if *current* is a solution for *csp* **then return** *current*

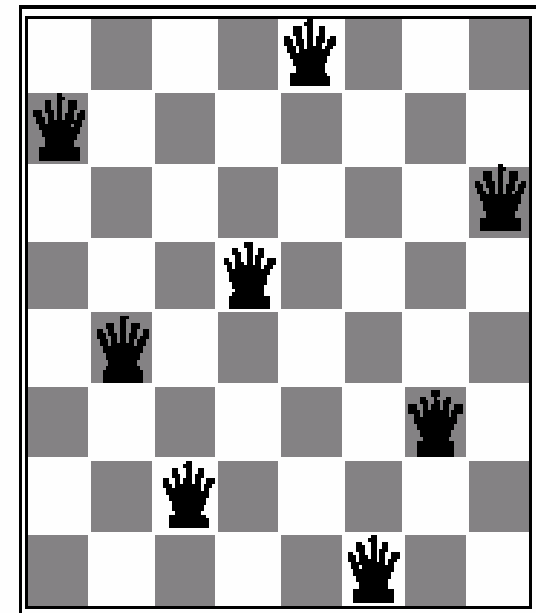
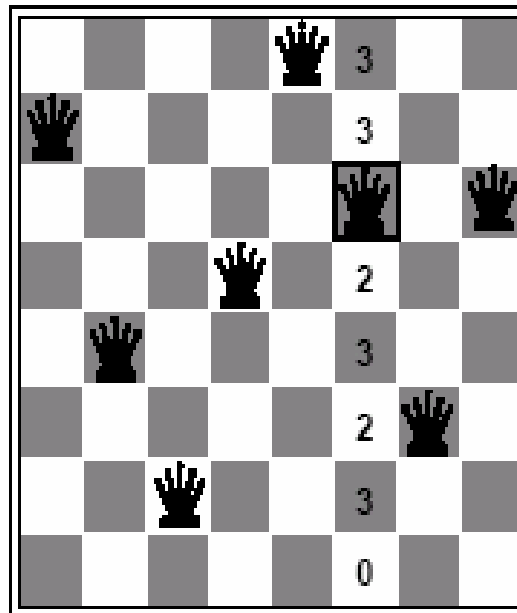
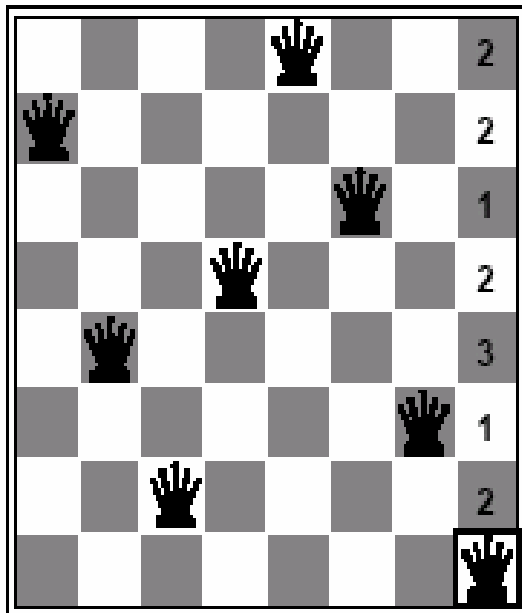
var \leftarrow a randomly chosen, conflicted variable from VARIABLES[*csp*]

value \leftarrow the value v for *var* that minimizes CONFLICTS(*var*, v , *current*, *csp*)

 set *var* = *value* in *current*

return *failure*

Otro Ejemplo: 8-reynas



Búsqueda Local para PSR : Ventajas

- El tiempo de corrida de conflictos-minimos es mas o menos independiente del tamaño del problema
 - Resolver el problema de millon-reynas se hace mas o menos en 50 pasos.
- Búsqueda local puede ser usada en un escenario online.
 - Busqueda Backtracking requiere mas tiempo



Resumen

- PSR sin un especial tipo de problemas
 - Los estados están definidos por valores de un conjunto fijo de variables
 - El test meta está definida por restricciones en los valores de las variables
 - Backtracking = búsqueda primero en profundidad con una variable asignada por nodo
 - Heurísticas de ordenación de variables y de selección de valores ayudan significativamente
 - Forward checking previenen asignaciones que previenen fallas posteriores
 - Propagación de restricciones (ejemplo: consistencia de arcos) hace trabajo adicional para restringir valores y detectar inconsistencias
 - Conflictos-mínimos es usualmente efectivo en la práctica
-

Referencias Bibliográficas

- Capitulo 5 Artificial Intelligence: A Modern Approach, Russell and Norvig
 - <http://www.aaai.org/AITopics/pmwiki/pmwiki.php/AITopics/Search>
-