Tema 4: Problemas de satisfacción de restricciones

Departamento de Lenguajes y Ciencias de la Computación
Universidad de Málaga

Contenidos

- 4.1 Introducción
- 4.2 Definición del problema y ejemplos
- 4.3 Restricciones y consistencia
- 4.4 Vuelta atrás
- 4.5 Conclusión

4.1 INTRODUCCIÓN

4.1 Introducción

Motivación

- El telescopio espacial Hubble fue lanzado en 1990
- Podía observar objetos nunca antes vistos
- Muchos astrónomos estaban interesados en usarlo

4.1 Introducción

Motivación

- Cada año había que planificar alrededor de 10.000-30.000 observaciones, cada una con varias restricciones operativas y científicas
  - Científica: sólo se puede observar un eclipse cuando está ocurriendo
  - Operativa: no puedes observar un objeto cuando está detrás de la Tierra
- El algoritmo de planificación inicial necesitaba 3 semanas para planificar una semana de observaciones (!)

4.1 Introducción

Representaciones factorizadas

- En los temas anteriores exploramos problemas que pueden resolverse buscando en un espacio de estados
- Cada estado era atómico, es decir, era una caja negra sin estructura interna
- Aquí emplearemos una representación factorizada para cada estado
  - Un conjunto de variables, cada una con su valor
  - Un problema está resuelto cuando cada variable tiene un valor que satisface todas las restricciones sobre dicha variable

4.2 DEFINICIÓN DEL PROBLEMA Y EJEMPLOS

4.2 Definición del problema y ejemplos
Definición (I)

- Un problema de satisfacción de restricciones (*constraint satisfaction problem*, CSP) está formado por tres componentes:
  - Un conjunto de variables, $X=\{X_1,\ldots,X_n\}$
  - Un conjunto de dominios, uno para cada variable: $D=\{D_1,\ldots, D_n\}$
  - Un conjunto de restricciones que especifican combinaciones permitidas de valores, C
- Cada dominio D_i es el conjunto de valores posibles $\{v_1,\ldots, v_k\}$ para la variable v_i

4.2 Definición del problema y ejemplos
Definición (II)

- Cada restricción C_i es un par <*scope*,*rel*> donde *scope* es la tupla de variables que intervienen en la restricción y *rel* es una relación que define los valores que dichas variables pueden tomar
- Por ejemplo, si las variables X_3 y X_5 deben tener valores distintos, podemos escribir esta restricción como $< (X_3,X_5), X_3 \neq X_5 >$

4.2 Definición del problema y ejemplos

Definición (III)

- Cada estado de un CSP se define como una asignación de valores a algunas o a todas las variables, $\{X_i=v_i, X_j=v_j,\ldots\}$
- Una asignación que no viola ninguna restricción se llama asignación consistente o legal
- Si tenemos una asignación en la cual todas las variables están asignadas, la llamamos asignación completa. En otro caso, la llamamos asignación parcial
- Una solución de un CSP es una asignación consistente y completa

4.2 Definición del problema y ejemplos

Ejemplo 1: Coloreado de mapas

- La tarea consiste en colorear cada región de un mapa de tal manera que no haya regiones adyacentes que tengan el mismo color
- Para el mapa de Australia (siguiente transparencia), definimos las variables como $X=\{WA, NT, Q, NSW, V, SA, T\}$
- El dominio de cada variable es $D_i=\{red, green, blue\}$
- Hay nueve restricciones: $C=\{SA \neq WA, SA \neq NT, SA \neq Q, SA \neq NSW, SA \neq V, WA \neq NT, NT \neq Q, Q \neq NSW, NSW \neq V\}$

4.2 Definición del problema y ejemplos

Ejemplo 1: Coloreado de mapas

4.2 Definición del problema y ejemplos

Ejemplo 2: problemas criptoaritméticos

- En un acertijo criptoaritmético, cada letra representa a un dígito distinto (0-9)

$$\begin{array}{cccc} & T & W & O \\ + & T & W & O \\ \hline F & O & U & R \end{array}$$

4.2 Definición del problema y ejemplos

Ejemplo 2: problemas criptoaritméticos

- El requisito de que todas las variables han de tomar diferentes valores se corresponde con la restricción *AllDiff*(*F*,*T*,*U*,*W*,*R*,*O*)
- Introducimos tres variables auxiliares C_{10}, C_{100} y C_{1000}, que representan los dígitos acarreados a las columnas de las decenas, las centenas y los millares, respectivamente
- De esta manera el resto de las restricciones son:
  - $O+O=R+10\cdot C_{10}$
  - $C_{10}+W+W=U+10\cdot C_{100}$
  - $C_{100}+T+T=O+10\cdot C_{1000}$
  - $C_{1000}=F$

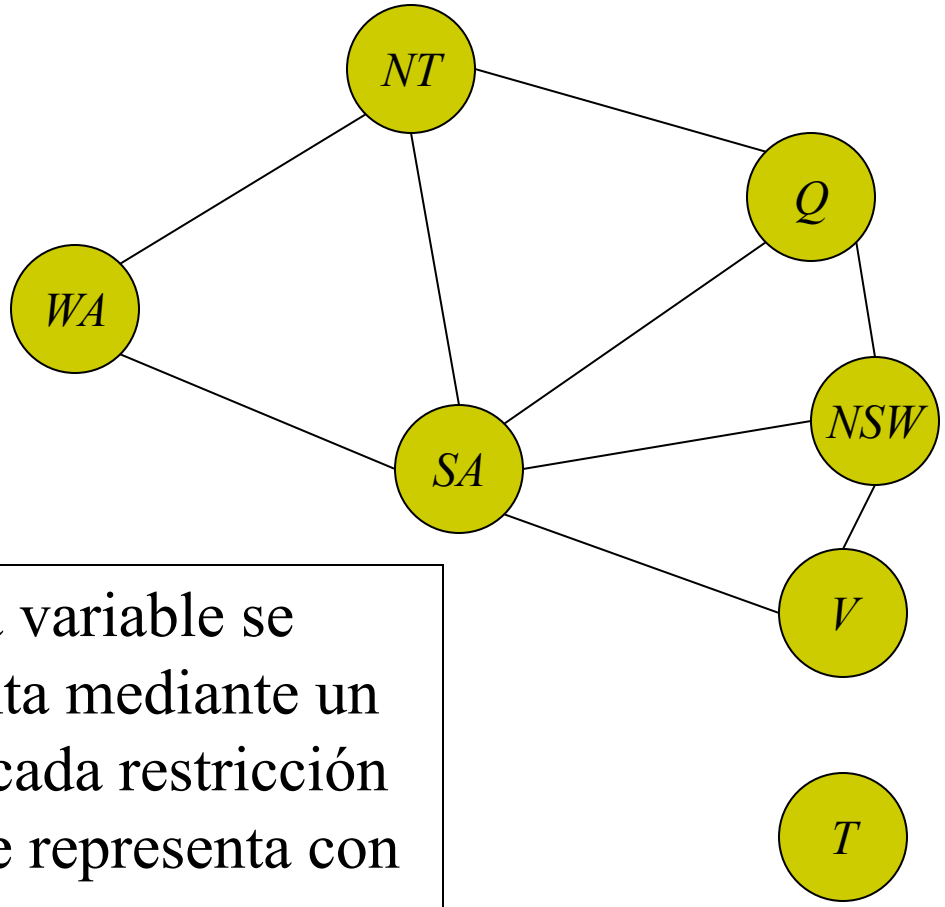
4.3 RESTRICCIONES Y CONSISTENCIA

4.3 Restricciones y consistencia
Tipos de restricciones

- Una restricción unaria restringe el valor de una sola variable
  - Por ejemplo, para imponer que *South Australia* no se coloree de verde escribimos $\langle(SA), SA \neq green\rangle$
- Una restricción binaria relaciona dos variables
  - Por ejemplo, $SA \neq NSW$
- Una restricción en la que participa un número arbitrario de variables se llama restricción global
  - Por ejemplo, $AllDiff(F,T,U,W,R,O)$

4.3 Restricciones y consistencia

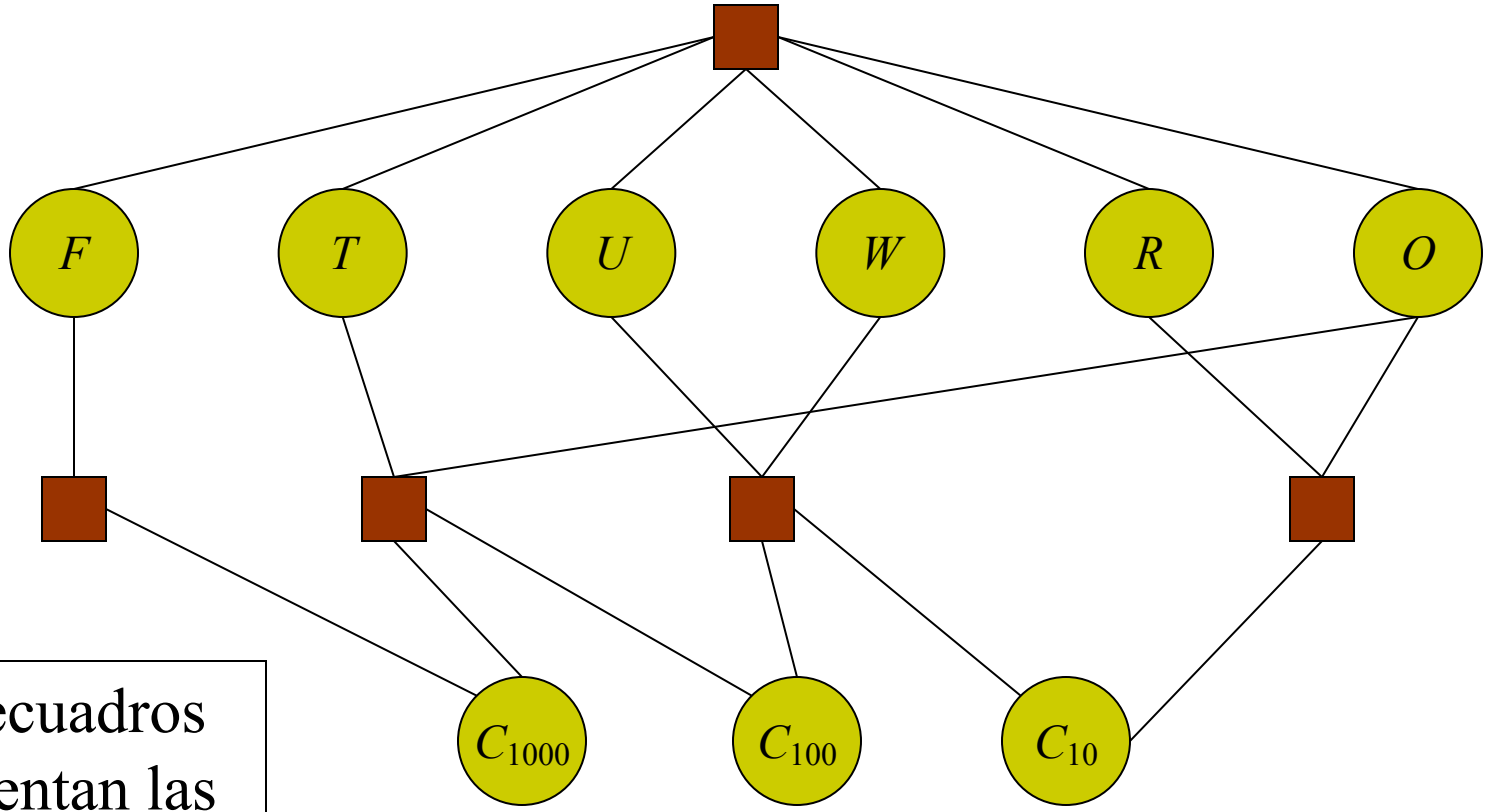
Grafo de restricciones



Cada variable se representa mediante un nodo, y cada restricción binaria se representa con un arco

4.3 Restricciones y consistencia

Hipergrafo de restricciones



Los recuadros representan las restricciones globales

4.3 Restricciones y consistencia
Consistencia de nodos

- Una variable es nodo-consistente si todos los valores del dominio de la variable satisfacen las restricciones unarias sobre dicha variable
  - Por ejemplo, si tenemos la restricción unaria <(*SA*),*SA*≠*green*>, podemos hacer *SA* nodo-consistente quitando *green* de su dominio, lo que deja a *SA* con el dominio reducido {*red*, *blue*}
- Siempre es posible eliminar todas las restricciones unarias de un CSP ejecutando la consistencia de nodos

4.3 Restricciones y consistencia

Consistencia de arcos

- Una variable X_i es arco-consistente si, para cada valor del dominio de X_i y cada restricción binaria (X_i,X_j), podemos encontrar al menos un valor en el dominio de X_j que satisface la restricción
- Una red es arco-consistente si toda variable es arco-consistente con las demás variables

4.3 Restricciones y consistencia
Algoritmo AC-3

- El algoritmo más popular para asegurar la arco-consistencia en una red se llama AC-3
- Mantiene un conjunto de arcos que considerar
- Inicialmente el conjunto contiene todos los arcos del CSP
- A continuación extrae un arco cualquiera (X_i,X_j) del conjunto y hace X_i arco-consistente con respecto a X_j
  - Si esto reduce el dominio D_i, entonces añadimos al conjunto todos los arcos (X_k,X_i), tales que X_k es vecino de X_i
- Si un dominio se reduce a nada, entonces el CSP original no tenía solución. En otro caso obtenemos un CSP en el que es más fácil buscar

4.4 VUELTA ATRÁS

4.4 Vuelta atrás

Algoritmo básico

- Muchos CSPs no se pueden resolver solamente por inferencia sobre las restricciones; llega un momento en el que hay que buscar una solución
- La búsqueda por vuelta atrás es una búsqueda primero en profundidad que en cada momento elige un valor para una sola variable, y vuelve atrás cuando una variable no tiene ningún valor legal que quede por probar
- Debemos considerar las siguientes preguntas:
  - ¿Qué variable debería ser asignada a continuación?
  - ¿En qué orden deberíamos probar sus valores?
  - ¿Qué inferencias deberían realizarse en cada paso?

4.4 Vuelta atrás

Ordenación de las variables

- Habitualmente se elige la variable que tenga el menor número de valores legales. Esto es lo que se llama el heurístico del mínimo número de valores restantes (*minimum remaining values heuristic*, MRV)
- A fin de romper los empates se puede emplear el heurístico del grado (*degree heuristic*, DEG), que elige la variable que interviene en el mayor número de restricciones con otras variables no asignadas
- Esta manera de elegir las variables intenta obtener un fallo tan pronto como sea posible para podar secciones más grandes del árbol de búsqueda rápidamente

4.4 Vuelta atrás

Ordenación de los valores

- En algunos casos el heurístico del valor menos restrictivo (*least constraining value*, LCV) puede resultar útil
- Prefiere el valor que elimina el menor número de opciones para las variables vecinas en el grafo de restricciones
- Este heurístico intenta obtener una solución tan pronto como sea posible eligiendo primero los valores más verosímiles

4.4 Vuelta atrás

Intercalando búsqueda e inferencia

- Cada vez que hacemos una elección de un valor para una variable, intentamos inferir nuevas reducciones de dominio en las variables vecinas
- Una de las estrategias más sencillas es la comprobación hacia delante (*forward checking*)
  - Cada vez que se asigna una variable X, para cada variable no asignada Y que está conectada a X mediante una restricción, borramos del dominio de Y los valores que son inconsistentes con el valor elegido para X
- La comprobación hacia delante es inútil si ya hemos ejecutado la consistencia de arcos como procesamiento previo

4.5 CONCLUSIÓN

4.5 Conclusión

Sumario

- Los problemas de satisfacción de restricciones representan un estado mediante un conjunto de pares variable/valor y representan las condiciones que debe cumplir la solución mediante un conjunto de restricciones sobre las variables
- Las técnicas de inferencia usan las restricciones para inferir qué pares variable/valor son consistentes
- Habitualmente se emplea la búsqueda por vuelta atrás para encontrar una solución

4.5 Conclusión

Epílogo

- Empleando técnicas similares a las estudiadas en este tema, el tiempo de planificación semanal se redujo de tres semanas a 10 minutos (AIMA, p. 221)

Nebulosa planetaria NGC 2818, vista por el telescopio espacial Hubble. Rojo = nitrógeno, verde = hidrógeno, azul = oxígeno.