Solución de Problemas con CCP problemas CSP

slides basados en el curso “constraint Programming” de Christian Schulte [2]
Profesor: Camilo Rueda [1]

[1]Universidad Javeriana-Cali,
[2]KTH Royal Institute of Technology, Sweden

PUJ 2008

Problemas de Satisfacción de Restricciones (CSP)

- Variables V
  conjunto finito de variables $V = \{x_0, x_1, ...\}$
- Universo U
  conjunto finito de valores U
- Restricciones C
  - qué variables estan involucradas
  - cuáles son las soluciones

Restricciones

- Una restricción c se define por
  - sus variables
    $var(c) = (x_1, ..., x_n) \in V^n$
  - sus soluciones
    $sol(c) \subseteq U^n = U \times ... \times U$ (n veces)

Asignaciones

- Una asignación a define qué valores pueden tener las variables
  - $a \in V \rightarrow U$
- una asignación a es solución de una restricción c (escrito $a \in c$) ssi
  - $var(c) = (x_1, ..., x_n)$ y
  - $(a(x_1), ..., a(x_n)) \in sol(c)$

Soluciones de un CSP

- Una asignación $a \in V \to U$ es solución a un CSP $P = \langle V, U, C \rangle$ si
  - $a \in c$ para toda $c \in C$
- Las soluciones $sol(P)$ de un CSP P se definen
  - $\{a \in V \to U \mid a$ solución de P$\}$

Ejemplo de un CSP

- $PWD = \langle V, U, C \rangle$ con
  - $V = \{x, y, z\}$
  - $U = \{1, 2, 3\}$
  - $C = \{c_1, c_2, c_3\}$ donde
    - $var(c_1) = (x, y)$
    - $sol(c_1) = \{(1,2), (1,3), (2,1), (2,3), (3,1), (3,2)\}$
    - $var(c_2) = (x, z)$ y $sol(c_2) = sol(c_1)$
    - $var(c_3) = (y, z)$ y $sol(c_3) = sol(c_1)$

Ejemplo: soluciones al CSP

$$sol(PWD) = \left\{ \begin{array}{l} \{x \mapsto 1, y \mapsto 2, z \mapsto 3\} \\ \{x \mapsto 1, y \mapsto 3, z \mapsto 2\} \\ \{x \mapsto 2, y \mapsto 1, z \mapsto 3\} \\ \{x \mapsto 2, y \mapsto 3, z \mapsto 1\} \\ \{x \mapsto 3, y \mapsto 1, z \mapsto 2\} \\ \{x \mapsto 3, y \mapsto 2, z \mapsto 1\} \end{array} \right\}$$

Simplificación

- En lo que sigue nos restringimos a restricciones binarias
  - las unarias: se extienden agregando todos los valores para una variable adicional
  - las n-arias: generalización

Arco consistencia

- Es otra expresión para dominio consistencia
- Considere c con $var(c) = (x, y)$
  - conserve el valor
    $$v \in s(x)$$
    solamente si
    $$(v, w) \in sol(c) \text{ y } w \in s(y)$$
  - análogamente para la variable y

Propagadores de Arco consistencia

- Defina ac-propagadores para las variables x, y
  - para cada restricción c con $var(c) = (x, y)$
- $ac(c, x)(s)(z) =$
  if $z \neq x$ *then* $s(z)$ *else*
  $\{v \in s(x) \mid \exists (v, w) \in sol(c).\ w \in s(y)\}$
  end
- $ac(c, y)(s)(z) =$
  if $z \neq y$ *then* $s(z)$ *else*
  $\{w \in s(y) \mid \exists (v, w) \in sol(c).\ v \in s(x)\}$
  end

Arco consistencia

- Dada la restricción c con $var(c) = (x, y)$ y un store s

s es arco-consistente para c

$\Leftrightarrow$

para todo $v \in s(x)$ existe $w \in s(y)$ tal que
$(v, w) \in sol(c)$

y también

para todo $w \in s(y)$ existe $v \in s(x)$ tal que
$(v, w) \in sol(c)$

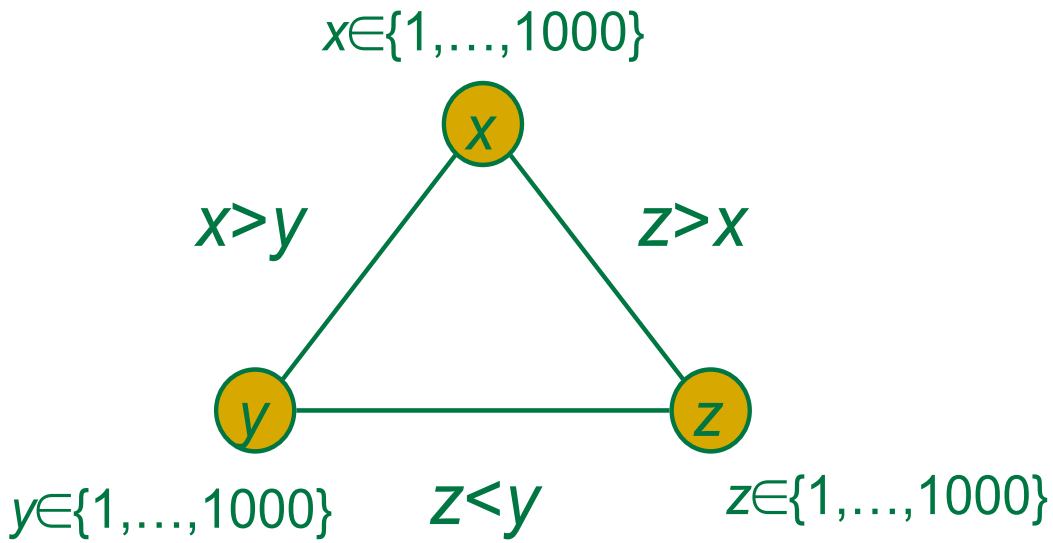
Store Arco consistente

- Un store s es arco-consistente para un CSP $\langle V, U, C \rangle$ si y slo si

para cada $c \in C$, s es arco-consistente para c

Camino Consistencia

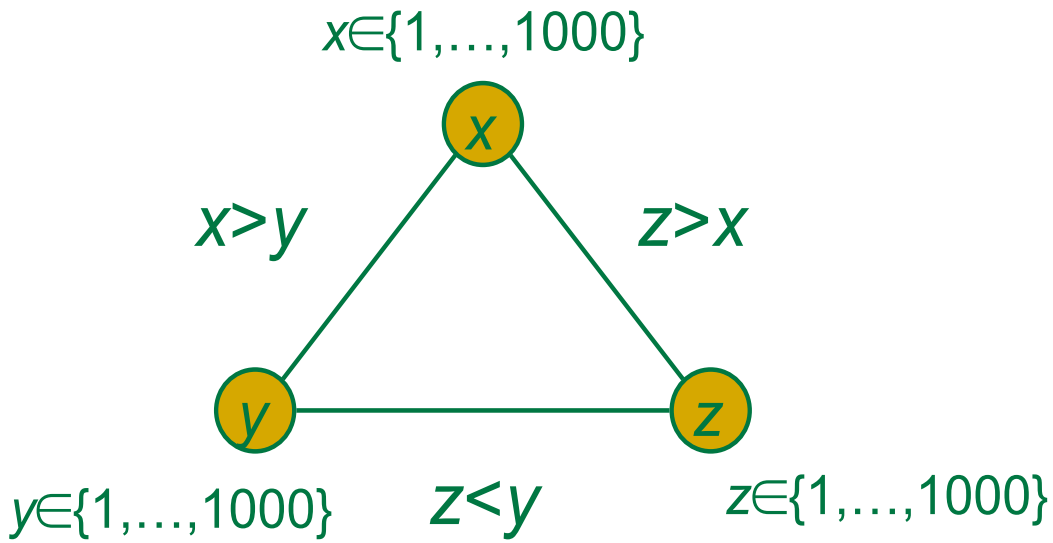
Un caso malo para AC:



requiere pasos de propagación proporcional a tamaño del store!

Camino Consistencia

Un caso malo para AC:



Por qué no propagar al tiempo $z > x$, $x > y$?
camino-consistencia: combinar dos restricciones

Componer restricciones

- Suponga c_1 con $var(c_1) = (x, y)$, $\quad c_2$ con $var(c_2) = (y, z)$
  enonces la combinación $c_1 \odot c_2$ se define como
  $$var(c_1 \odot c_2) = (x, z)$$
  y
  $$sol(c_1 \odot c_2) = \{(v, k) \mid (v, m) \in sol(c_1), (m, k) \in sol(c_2)\}$$

Lograr Camino Consistencia

- Construir composiciones para restricciones arbitrarias
- Realiza AC sobre las restricciones compuestas
- Poderoso pero caro computacionalmente
  - ningún sistema usa esta idea
  - Consultar texto de Apt