

Programación declarativa: lógica y restricciones

Programación Lógica con Restricciones *Constraint Logic Programming (CLP)*

Mari Carmen Suárez de Figueroa Baonza

mcsuarez@fi.upm.es



POLITÉCNICA

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

--

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70

ucción (I)

es un lenguaje rico y potente para modelar
emas de optimización

ece una forma declarativa para modelar problemas de
sfacción de restricciones

odelado se basa en variables, dominios y
cciones

dominios: reales, enteros, booleanos, etc.

LP(R)

limitaciones particulares permitidas para cada dominio:

por ejemplo, limitaciones aritméticas (+, *, =, ≤, ≥, <, >)

algoritmos de resolución de restricciones: simplex, gauss, etc.

problema de satisfacción de restricciones se puede presentar como un triple formado por:

un conjunto de **variables** $V = \{X_1, \dots, X_n\}$

para cada variable de V un conjunto de posibles valores D_i , que llamaremos **dominio** de X_i

un **conjunto de restricciones**, normalmente binarias, $C_{ij}(X_i, X_j)$ que determinan los valores que las variables pueden tomar simultáneamente

el **objetivo** es encontrar un valor para cada variable de manera que se satisfagan todas las restricciones del problema

una restricción limita el conjunto de asignaciones para las variables implicadas

ucción (III)

ajas:

Mayor expresividad en el tratamiento de problemas

Diseño más uniforme y mayor efectividad

Puede ahorrar mucha codificación

Aumento de la eficiencia

Gracias a la reducción del espacio de búsqueda

LP: generate-and-test

CLP: limitar-y-generar

ventajas:

Algoritmos de resolución (simplex, gauss, etc.) complejos que pueden afectar al rendimiento

Necesidad de técnicas específicas para el tratamiento de los objetos

Restricciones en Ciao Prolog

Se añaden como extensiones al sistema Prolog principal

que requieren la declaración inicial correspondiente

Se definen operadores especiales para expresar las restricciones

de la forma `<op> .> . , .>= .` etc.

Restricciones sobre el dominio de los **racionales**

```
use_package(clpq).
```

Restricciones sobre el dominio de los **reales**

```
use_package(clpr).
```

Restricciones en SWI Prolog

Extensiones modulares al sistema Prolog principal

requieren la declaración inicial correspondiente

Restricciones sobre el dominio de los **racionales y los reales**

`use_module(library(clpq))` o bien `:- use_module(library(clpr))`

Las restricciones se expresan con los operadores aritméticos usuales, pero encerrados entre llaves: { }

Además, `=/2` expresa una restricción de identidad (no de unificabilidad)

Restricciones sobre dominios finitos (**discretos**)

`use_module(library(clpfd))`. Similar a SICStus Prolog

Las restricciones se expresan con operadores específicos: `#>`, `#=`, etc.

Cuando las restricciones sólo acotan un rango de valores, `label/1` genera (en *backtraking*) todos los valores posibles

K): Programas

Programa en CLP es una colección de reglas de la forma

$a \leftarrow b_1, \dots, b_n$ (donde a es un átomo y los b_i 's son átomos o restricciones)

Hecho es una regla

$a \leftarrow c$ (donde c es una restricción)

Objetivo (o consulta) G es una conjunción de restricciones y átomos

Restricción es una fórmula de primer orden derivada con restricciones primitivas

$p(t_1, \dots, t_n)$, con términos t_1, t_2, \dots, t_n y p símbolo de predicado, es restricción primitiva

g no puede resolver $x-3 = y+5$

\mathcal{R}) es un lenguaje basado en Prolog, que incluye
idades para resolver restricciones sobre números

resiones aritméticas lineales: números, variables y operadores
gación, suma, resta, multiplicación y división)

mplo: $t1 \ R \ t2$, donde $R = \{ >, \geq, =, \leq, <, = \}$

\mathcal{R}) utiliza la misma estrategia de ejecución que

nero en profundidad, de izquierda a derecha

\mathcal{R}) es capaz de resolver directamente (in)-
ciones lineales sobre números reales

Información Lógica vs. CLP(\mathbb{R}) (I)

prolog: (Prolog)

$f(X, Y, Z) :- Z = f(X, Y).$

$f(3, 4, Z).$

$Z = f(3,4)$

$f(X, Y, f(3,4)).$

$X = 3, Y = 4$

$f(X, Y, Z).$

$Z = f(X,Y)$

clp: (Prolog)

$f(X, Y, Z) :- Z \text{ is } X + Y.$

$f(3, 4, Z). \quad \% \text{ modo in-in-out}$

$Z = 7$

$f(X, 4, 7). \quad \% \text{ modo out-in-in}$

Instantiation Error

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
 LLAMA O ENVÍA WHATSAPP: 689 45 44 70
 ...
 ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
 CALL OR WHATSAPP: 689 45 44 70

Información Lógica vs. CLP (\mathcal{R}) (II)

ejemplo: (CLP(\mathcal{R}))

$(X, Y, Z) :- Z = X + Y.$

$(3, 4, Z).$ % modo in-in-out

$Z = 7$

es

$(X, 4, 7).$ % modo out-in-in

$X = 3$

es

$(X, Y, 7).$ % modo out-out-in

$X = 7 - Y$

es

[o.pl](#)

Información Lógica vs. CLP(\mathbb{R}) (III)

ejemplo: Reducción del espacio de búsqueda

g (generar y testear):

relation(X, Y, Z) :- p(X), p(Y), p(Z), test(X, Y, Z).

4). p(15). p(16). p(7). p(3). p(11).

(X, Y, Z) :- Y is X + 1, Z is Y + 1.

... solution(X, Y, Z).

= 14,

= 15,

= 16 ? ;

o

pasos (todas las soluciones: 465 pasos)

EspacioBusqueda.pl

Información Lógica vs. CLP (\mathcal{R}) (IV)

ejemplo: Reducción del espacio de búsqueda

\mathcal{R}) (generar y testear):

relation(X, Y, Z) :- p(X), p(Y), p(Z), test(X, Y, Z).

4). p(15). p(16). p(7). p(3). p(11).

(X, Y, Z) :- Y .=. X + 1, Z .=. Y + 1.

solution(X, Y, Z).

= 16,

= 15,

= 14 ?;

o

pasos (todas las soluciones: 465 pasos)

Información Lógica vs. CLP (\mathcal{R}) (V)

ejemplo: Reducción del espacio de búsqueda

Cambiar 'test(X, Y, Z)' al principio (**restringir y generar**):

Definition(X, Y, Z) :- test(X, Y, Z), p(X), p(Y), p(Z).

4). p(15). p(16). p(7). p(3). p(11).

Log: test(X, Y, Z) :- Y is X + 1, Z is Y + 1.

- solution(X, Y, Z).

Instantiation Error

(\mathcal{R}): test(X, Y, Z) :- Y .=. X + 1, Z .=. Y + 1.

- solution(X, Y, Z).

= 16,

= 15,

= 14 ?;

o

pasos (todas las soluciones: 11 pasos)

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
 LLAMA O ENVÍA WHATSAPP: 689 45 44 70
 ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
 CALL OR WHATSAPP:689 45 44 70

$$mv^2 + 9.81 mh$$

olog, un procedimiento que calcule cualquiera de cuatro variables dadas las otras tres:

```

energia(E,M,V,H):-
    number(M),
    number(V),
    number(H),
    V is sqrt((E - 9.81*M*H)/0.5*M).

energia(E,M,V,H):-
    number(E),
    number(M),
    number(V),
    H is (E - 0.5*M*V*V)/(9.81*M).

```

```

energia(E,M,V,H):-
    number(E),
    number(M),
    number(H),
    V is sqrt((E - 9.81*M*H)/0.5*M).

energia(E,M,V,H):-
    number(E),
    number(V),
    number(H),
    M is E / (0.5*V*V + 9.81*H).

```

CLP(R):

```

:- use_module(library(clpr)).

energia(E,M,V,H):-
    0.5*M*V*V + 9.81*M*H.

```

```

% SWI Prolog
:- use_module(library(clpq)).

energia(E,M,V,H):-
    { E = 0.5*M*V*V + 9.81*M*H }.

```

CLASES PARTICULARES, TUTORIAS TÉCNICAS ONLINE
 LLAMA O ENVIA WHATSAPP: 689 45 44 70
 ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
 CALL OR WHATSAPP:689 45 44 70

Operaciones Lineales (CLP(\mathbb{R}))

Operación de producto de vectores (de números reales):

$$(x_1, \dots, x_n) \cdot (y_1, y_2, \dots, y_n) = x_1 \cdot y_1 + \dots + x_n \cdot y_n$$

Representamos los vectores como listas de números
 prod([1, 2], [3, 4], 0).

$$\text{prod}([X|Xs], [Y|Ys], P) :- \quad P = X * Y + \text{Rest}, \\ \text{prod}(Xs, Ys, \text{Rest}).$$

La consulta se convierte en resolución de restricciones

$$\text{prod}([2, 3], [4, 5], K). \\ K = 23$$

$$\text{prod}([2, 3], [5, X2], 22). \\ X2 = 4$$

$$\text{prod}([2, 7, 3], [Vx, Vy, Vz], 0). \\ Vx = -1.5 * Vz - 3.5 * Vy$$

Cualquier respuesta calculada es, en general, una solución sobre las variables de la consulta

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
 LLAMA O ENVÍA WHATSAPP: 689 45 44 70
 ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
 CALL OR WHATSAPP: 689 45 44 70

Sistemas de Ecuaciones Lineales (CLP(\mathbb{R}))

¿Podemos resolver sistemas de ecuaciones?

$$x - y = 5$$

$$8y = 3$$

```
prod([3, 1], [X, Y], 5), prod([1, 8], [X, Y], 3).
```

```
X = 1.6087, Y = 0.173913
```

Podemos construir un predicado más general imitando la notación vectorial matemática $A \cdot x = b$:

```
prod(_Vars, [], []).
```

```
prod(_Vars, [Co | Coefs], [Ind | Indeps]) :-
```

```
prod(_Vars, Co, Ind),
```

```
prod(_Vars, Coefs, Indeps).
```

```
prod([X, Y], [[3, 1],[1, 8]], [5, 3]).
```

% podemos expresar y resolver sistemas de ecuaciones

```
X = 1.6087, Y = 0.173913
```

www.cartagena99.com/SistemasLineales.pl

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
 LLAMA O ENVÍA WHATSAPP: 689 45 44 70
 ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
 CALL OR WHATSAPP:689 45 44 70

R): Ejemplo

comida consiste en un entrante, un plato principal y postre

tenemos que existe una base de datos con distintos platos de comida y sus valores calóricos

debe producir un **menú con comida *light*** (valor calórico menor de 10Kcal)

www.cartagena99.com

--

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP: 689 45 44 70

Titulo: Pasatiempo

El pasatiempo consta de las siguientes afirmaciones:

Un alemán y un británico viven cada uno en una casa de diferente color y tienen diferentes mascotas

El alemán vive en la casa verde

El británico tiene un perro en la casa blanca

Ante esta situación plantea la siguiente pregunta:

¿Quién tiene un gato?

Ejercicio: Pasatiempo

El pasatiempo consta de las siguientes afirmaciones:

Un alemán, un británico y un sueco viven cada uno en una casa de un color diferente, tienen diferentes mascotas y gustan diferentes comidas.

El alemán vive en la casa verde.

El sueco bebe café.

El británico no le gustan los gatos.

Hay un perro en la casa blanca.

El sueco vive junto a la casa azul.

El británico bebe agua y tiene un pez.

Ante esta información plantea la siguiente pregunta:

¿Quién bebe té?

tempo_ejercicio.pl

Problema: SEND + MORE = MONEY

Este problema quiere decir, en inglés, “Envía más dinero”

Se trata de sustituir, en la suma siguiente, las letras por cifras (de 0 a 9) teniendo en cuenta que a cada letra distinta le corresponde una cifra diferente

Las variables S, E, N, D, M, O, R, Y representan dígitos entre 0 and 9. La tarea consiste en encontrar valores para estas variables de manera que la operación $SEND + MORE = MONEY$ es correcta. Como las variables deben tomar valores únicos, los números deben estar bien formados (lo que implica que $M > 0$ y

$$\begin{array}{r}
 SEND \\
 + MORE \\
 \hline
 MONEY
 \end{array}$$

Programación declarativa: lógica y restricciones

Programación Lógica con Restricciones *Constraint Logic Programming (CLP)*

Mari Carmen Suárez de Figueroa Baonza

mcsuarez@fi.upm.es



POLITÉCNICA

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

--

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENTS
CALL OR WHATSAPP:689 45 44 70