Clips: Patrones lógicos

José A. Alonso y Francisco J. Martín

Ciencias de la Computación e Inteligencia Artificial

Universidad de Sevilla

Revisión de la definición de reglas

```
<constructor-defrule>
   ::= (defrule <nombre> [<comentario>]
         [<declaración>]
         <condición>*
         =>
         <accion>*)
<declaración>
   ::= (declare (salience <numero>)) |
       (declare (auto-focus <booleano>))
<condición>
   ::= <patrón> |
       <patrón-asignado> |
       <test>
<patrón>
   ::= <patrón-ordenado> |
       <patrón-plantilla>
<patrón-ordenado>
   ::= (<símbolo> <restricción>*)
<restricción>
   ::= <término> |
       ~<término>
       <término>&<restricción> |
       <término>|<restricción>
<término>
   ::= <constante> |
       <variable> |
       :<llamada-a-función> |
       =<llamada-a-función>
```

Revisión de la definición de reglas

```
<variable>
   ::= <variable-simple> |
       <variable-múltiple>
<variable-simple>
   ::= ?[<nombre>]
<variable-múltiple>
   ::= $?[<nombre>]
<llamada-a-función>
   ::= (<nombre-de-función> <expresión>*)
<nombre-de-función>
   ::= Cualquier símbolo correspondiente a una función
       predefinida o definida por el usuario mediante
       deffunction
<expresión>
   ::= <constante> |
       <variable> |
       <llamada-a-función>
<patrón-plantilla>
   ::= (<nombre-de-la-plantilla> <restricción-de-ranura>*)
<restricción-de-ranura>
   ::= <restricción-de-ranura-simple> |
       <restricción-de-ranura-múltiple>
<restricción-de-ranura-simple>
   ::= (<nombre-de-ranura> <restricción>)
<restricción-de-ranura-múltiple>
   ::= (<nombre-de-ranura> <restricción>*)
<patrón-asignado>
   ::= ?<nombre> <- <patrón>
<test>
   ::= (test <llamada-a-función>)
<acción>
   ::= <expresión>
```

Reglas disyuntivas

- Ejemplo: `ej-1.clp`

```
(defrule no-hay-clase-1
  (festivo hoy)
  =>
  (printout t "Hoy no hay clase" crlf))

(defrule no-hay-clase-2
  (sabado hoy)
  =>
  (printout t "Hoy no hay clase" crlf))

(defrule no-hay-clase-3
  (hay-examen hoy)
  =>
  (printout t "Hoy no hay clase" crlf))

(deffacts inicio
  (sabado hoy)
  (hay-examen hoy))
```

Reglas disyuntivas

- **Sesión**

```
CLIPS> (clear)
CLIPS> (unwatch all)
CLIPS> (watch facts)
CLIPS> (watch activations)
CLIPS> (watch rules)
CLIPS> (load "ej-1.clp")
***$
TRUE
CLIPS> (reset)
==> f-0     (initial-fact)
==> f-1     (sabado hoy)
==> Activation 0      no-hay-clase-2: f-1
==> f-2     (hay-examen hoy)
==> Activation 0      no-hay-clase-3: f-2
CLIPS> (run)
FIRE    1 no-hay-clase-3: f-2
Hoy no hay clase
FIRE    2 no-hay-clase-2: f-1
Hoy no hay clase
```

Patrones disyuntivos

- Ejemplo: `ej-2.cl`

```
(defrule no-hay-clase
  (or (festivo hoy)
      (sabado hoy)
      (hay-examen hoy))
  =>
  (printout t "Hoy no hay clase" crlf))

(deffacts inicio
  (sabado hoy)
  (hay-examen hoy))
```

- Comentarios
  - Bloques lógicos
  - Equivalencia de programas

Patrones disyuntivos

- Sesión

```
CLIPS> (clear)
CLIPS> (unwatch all)
CLIPS> (watch facts)
CLIPS> (watch activations)
CLIPS> (watch rules)
CLIPS> (load "ej-2.clp")
CLIPS> *$
TRUE
CLIPS> (reset)
==> f-0     (initial-fact)
==> f-1     (sabado hoy)
==> Activation 0      no-hay-clase: f-1
==> f-2     (hay-examen hoy)
==> Activation 0      no-hay-clase: f-2
CLIPS> (run)
FIRE    1 no-hay-clase: f-2
Hoy no hay clase
FIRE    2 no-hay-clase: f-1
Hoy no hay clase
CLIPS>
```

Conjunción de patrones disyuntivos

- Ejemplo: `ej-3.clp`

```
(defrule no-hay-clase
  (periodo lectivo)
  (or (festivo hoy)
      (sabado hoy)
      (hay-examen hoy))
  =>
  (printout t "Hoy no hay clase" crlf))

(deffacts inicio
  (sabado hoy)
  (hay-examen hoy))
```

Conjunción de patrones disyuntivos

- Sesión

```
CLIPS> (clear)
CLIPS> (unwatch all)
CLIPS> (watch facts)
CLIPS> (watch activations)
CLIPS> (watch rules)
CLIPS> (load "ej-3.clp")
CLIPS> *$
TRUE
CLIPS> (reset)
==> f-0     (initial-fact)
==> f-1     (sabado hoy)
==> f-2     (hay-examen hoy)
CLIPS> (run)
CLIPS> (assert (periodo lectivo))
==> f-3     (periodo lectivo)
==> Activation 0      no-hay-clase: f-3,f-2
==> Activation 0      no-hay-clase: f-3,f-1
<Fact-3>
CLIPS> (run)
FIRE    1 no-hay-clase: f-3,f-1
Hoy no hay clase
FIRE    2 no-hay-clase: f-3,f-2
Hoy no hay clase
CLIPS>
```

Conjunción de patrones disyuntivos

- Programa equivalente sin patrones disyuntivos: ej-4.clp

```
(defrule no-hay-clase-1
  (periodo lectivo)
  (festivo hoy)
  =>
  (printout t "Hoy no hay clase" crlf))

(defrule no-hay-clase-2
  (periodo lectivo)
  (sabado hoy)
  =>
  (printout t "Hoy no hay clase" crlf))

(defrule no-hay-clase-3
  (periodo lectivo)
  (hay-examen hoy)
  =>
  (printout t "Hoy no hay clase" crlf))

(deffacts inicio
  (sabado hoy)
  (hay-examen hoy))
```

Conjunción de patrones disyuntivos

- Sesión

```
CLIPS> (clear)
CLIPS> (unwatch all)
CLIPS> (watch facts)
CLIPS> (watch activations)
CLIPS> (watch rules)
CLIPS> (load "ej-4.clp")
CLIPS> ***$
TRUE
CLIPS> (reset)
==> f-0     (initial-fact)
==> f-1     (sabado hoy)
==> f-2     (hay-examen hoy)
CLIPS> (assert (periodo lectivo))
==> f-3     (periodo lectivo)
==> Activation 0      no-hay-clase-3: f-3,f-2
==> Activation 0      no-hay-clase-2: f-3,f-1
<Fact-3>
CLIPS> (run)
FIRE    1 no-hay-clase-2: f-3,f-1
Hoy no hay clase
FIRE    2 no-hay-clase-3: f-3,f-2
Hoy no hay clase
CLIPS>
```

Limitación de disparos disyuntivos

- Ejemplo: `ej-5.clp`

```
(defrule no-hay-clase
  ?periodo <- (periodo lectivo)
  (or (festivo hoy)
      (sabado hoy)
      (hay-examen hoy))
  =>
  (retract ?periodo)
  (assert (periodo lectivo-sin-clase))
  (printout t "Hoy no hay clase" crlf))

(deffacts inicio
  (sabado hoy)
  (hay-examen hoy)
  (periodo lectivo))
```

Limitación de disparos disyuntivos

- Sesión

```
CLIPS> (clear)
CLIPS> (unwatch all)
CLIPS> (watch facts)
CLIPS> (watch activations)
CLIPS> (watch rules)
CLIPS> (load "ej-5.clp")
CLIPS> *$
TRUE
CLIPS> (reset)
==> f-0     (initial-fact)
==> f-1     (sabado hoy)
==> f-2     (hay-examen hoy)
==> f-3     (periodo lectivo)
==> Activation 0      no-hay-clase: f-3,f-2
==> Activation 0      no-hay-clase: f-3,f-1
CLIPS> (run)
FIRE    1 no-hay-clase: f-3,f-1
<== f-3     (periodo lectivo)
<== Activation 0      no-hay-clase: f-3,f-2
==> f-4     (periodo lectivo-sin-clase)
Hoy no hay clase
CLIPS>
```

Eliminación de causas disyuntivas

- Ejemplo: `ej-6.clp`

```
(defrule no-hay-clase
  ?periodo <- (periodo lectivo)
  (or ?causa <- (festivo hoy)
      ?causa <- (sabado hoy)
      ?causa <- (hay-examen hoy))
  =>
  (retract ?periodo ?causa)
  (assert (periodo lectivo-sin-clase))
  (printout t "Hoy no hay clase" crlf))

(deffacts inicio
  (sabado hoy)
  (hay-examen hoy)
  (periodo lectivo))
```

Eliminación de causas disyuntivas

- Sesión

```
CLIPS> (clear)
CLIPS> (unwatch all)
CLIPS> (watch facts)
CLIPS> (watch activations)
CLIPS> (load "ej-6.clp")
CLIPS> *$
TRUE
CLIPS> (reset)
==> f-0     (initial-fact)
==> f-1     (sabado hoy)
==> f-2     (hay-examen hoy)
==> f-3     (periodo lectivo)
==> Activation 0      no-hay-clase: f-3,f-2
==> Activation 0      no-hay-clase: f-3,f-1
CLIPS> (run)
<== f-3     (periodo lectivo)
<== Activation 0      no-hay-clase: f-3,f-2
<== f-1     (sabado hoy)
==> f-4     (periodo lectivo-sin-clase)
Hoy no hay clase
CLIPS> (facts)
f-0     (initial-fact)
f-2     (hay-examen hoy)
f-4     (periodo lectivo-sin-clase)
For a total of 3 facts.
CLIPS>
```

Patrones conjuntivos

- Conjunciones implícitas

```
(defrule no-hay-clase-por-puente
  (festivo ayer)
  (festivo manana)
  =>
  (printout t "Hoy no hay clase" crlf))
```

- Conjunciones explícitas

```
(defrule no-hay-clase-por-puente
  (and (festivo ayer)
       (festivo manana))
  =>
  (printout t "Hoy no hay clase" crlf))
```

Patrones conjuntivos

- Conjunciones en disyunciones

```
(defrule no-hay-clase
  ?periodo <- (periodo lectivo)
  (or (festivo hoy)
      (sabado hoy)
      (and (festivo ayer)
           (festivo manana)))
  =>
  (retract ?periodo)
  (assert (periodo lectivo-sin-clase))
  (printout t "Hoy no hay clase" crlf))

(deffacts inicio
  (periodo lectivo)
  (festivo ayer)
  (festivo manana))
```

Patrones conjuntivos

- **Sesión**

```
CLIPS> (clear)
CLIPS> (unwatch all)
CLIPS> (watch facts)
CLIPS> (watch activations)
CLIPS> (load "ej-7.clp")
CLIPS> *$
TRUE
CLIPS> (reset)
==> f-0     (initial-fact)
==> f-1     (periodo lectivo)
==> f-2     (festivo ayer)
==> f-3     (festivo manana)
==> Activation 0      no-hay-clase: f-1,f-2,f-3
CLIPS> (run)
<== f-1     (periodo lectivo)
==> f-4     (periodo lectivo-sin-clase)
Hoy no hay clase
CLIPS>
```

Patrones conjuntivos

- Reglas equivalentes

```
(defrule no-hay-clase-1
  ?periodo <- (periodo lectivo)
  (festivo hoy)
  =>
  (retract ?periodo)
  (assert (periodo lectivo-sin-clase))
  (printout t "Hoy no hay clase" crlf))

(defrule no-hay-clase-2
  ?periodo <- (periodo lectivo)
  (sabado hoy)
  =>
  (retract ?periodo)
  (assert (periodo lectivo-sin-clase))
  (printout t "Hoy no hay clase" crlf))

(defrule no-hay-clase-3
  ?periodo <- (periodo lectivo)
  (festivo ayer)
  (festivo manana)
  =>
  (retract ?periodo)
  (assert (periodo lectivo-sin-clase))
  (printout t "Hoy no hay clase" crlf))
```

Patrones negativos

- Ejemplo: Unión

```
(deffacts datos-iniciales
  (conjunto-1 a b)
  (conjunto-2 b c))

(defrule calcula-union
  =>
  (assert (union)))

(defrule union-base
  ?union <- (union $?u)
  ?conjunto-1 <- (conjunto-1 $?e-1)
  ?conjunto-2 <- (conjunto-2)
  =>
  (retract ?conjunto-1 ?conjunto-2 ?union)
  (assert (union ?e-1 ?u))
  (assert (escribe-solucion)))

(defrule escribe-solucion
  (escribe-solucion)
  (union $?u)
  =>
  (printout t "La union es " ?u crlf))
```

Patrones negativos

```
(defrule union-con-primero-compartido
  (union $?u)
  ?conjunto-2 <- (conjunto-2 ?e $?r-2)
  (conjunto-1 $?a-1 ?e $?r-1)
  =>
  (retract ?conjunto-2)
  (assert (conjunto-2 ?r-2)))

(defrule union-con-primero-no-compartido
  ?union <- (union $?u)
  ?conjunto-2 <- (conjunto-2 ?e $?r-2)
  (not (conjunto-1 $?a-1 ?e $?r-1))
  =>
  (retract ?conjunto-2 ?union)
  (assert (conjunto-2 ?r-2))
  (assert (union ?u ?e)))
```

Patrones negativos

- Sesión

```
CLIPS> (watch facts)
CLIPS> (watch rules)
CLIPS> (load "ej-8.clp")
CLIPS> $*****
TRUE
CLIPS> (reset)
==> f-0     (initial-fact)
==> f-1     (conjunto-1 a b)
==> f-2     (conjunto-2 b c)
CLIPS> (run)
FIRE    1 calcula-union: f-0
==> f-3     (union)
FIRE    2 union-con-primero-compartido: f-3,f-2,f-1
<== f-2     (conjunto-2 b c)
==> f-4     (conjunto-2 c)
FIRE    3 union-con-primero-no-compartido: f-3,f-4,
<== f-4     (conjunto-2 c)
<== f-3     (union)
==> f-5     (conjunto-2)
==> f-6     (union c)
FIRE    4 union-base: f-6,f-1,f-5
<== f-1     (conjunto-1 a b)
<== f-5     (conjunto-2)
<== f-6     (union c)
==> f-7     (union a b c)
==> f-8     (escribe-solucion)
FIRE    5 escribe-solucion: f-8,f-7
La union es (a b c)
CLIPS>
```

Patrones lógicos

- Extensión de <condicion>

```
<condicion>
   ::= <patron> |
       <patron-asignado> |
       <test> |
       <patron-disyuntivo> |
       <patron-conjuntivo> |
       <patron-negativo>
<patron-disyuntivo>
   ::= (or <condicion>+)
<patron-conjuntivo>
   ::= (and <condicion>+)
<patron-negativo>
   ::= (not <condicion>)
```