Ingeniería del Conocimiento

Problemas de Sistemas de Producción
Universidad Rey Juan Carlos
4º Ingeniería Informática

Diego Chaparro González
dchaparro@acm.org

11 de septiembre de 2001

Índice General

Capítulo 1

Ejercicio 1: Jarras de agua

1.1 Apartado a

En este primer apartado se pide que se utilize la siguiente plantilla para hacer un programa en CLIPS:

```
(deftemplate jarra
(slot capacidad
(type INTEGER))

(slot litros
(type INTEGER)
(default 0)))
```

Pues este primer programa es muy sencillo, lo único que hace es pedir los litros que tienen 3 jarras, y el programa averigua cual es la jarra que tiene mayor cantidad, y cual tiene menor cantidad.

Un ejemplo de ejecución es el siguiente:

```
XCLIPS for:         CLIPS (V6.10 07/01/98)
CLIPS> (load "/home/dchaparro/Universidad/2Cuatrimestre/IngConoc/
Practicas/Ej1/jar.clp")
Defining deftemplate: jarra
Defining defrule: Estado-Inicial +j
Defining defrule: Pedir_Litros +j+j+j
Defining defrule: Jarra_Mas_Llena =j=j+j
Defining defrule: Jarra_Menos_Llena =j=j+j
TRUE
CLIPS> (reset)
CLIPS> (run)

Tenemos 3 jarras de capacidad 10. Cuales son los litros de cada una(distintos)??
Litros de la jarra1??: 3

Litros de la jarra2??: 7
```

```
Litros de la jarra3??: 9

La jarra con 3 litros es la menos llena
La jarra con 9 litros es la más llena
```

Simplemente hay una regla que pide las cantidades nada más empezar, y luego hay otras dos reglas que comparan el contenido de las jarras y comprueban el valor mayor y menor de esas cantidades.

1.2 Apartado b

Realice un programa en CLIPS para solucionar el problema de las jarras realizando una búsqueda en amplitud. Para realizar este programa es necesario controlar en cada momento la profundidad del árbol, y los estados que no han sido explorados.

La estructura utilizada para resolver este problema es la siguiente:

```
(deftemplate estado
(slot litros_j3l ; Litros que tiene la jarra de 3l
(type INTEGER)
(default 0)
)
(slot litros_j4l ; litros que tiene la jarra de 4l
(type INTEGER)
(default 0)
)

(slot profundidad ; Profundidad del nodo en el arbol
(type INTEGER)
(default 0)
(range 0 ?VARIABLE)
)

(slot nodo ; Número de nodo en el árbol
(type INTEGER)
(default 0)
)

(slot padre ; Nodo padre
(type FACT-ADDRESS SYMBOL)
(allowed-symbols no-padre)
)

(slot operacion ; Descripcion del movimiento
(type STRING)
)
)
```

El programa va construyendo un árbol con nodos que contienen esta estructura anterior, y lo va expandiendo en amplitud. Parte de un nodo en el que el estado de las jarras es vacío, y a partir de ahí va expandiendo las ramas todo lo que puede con las operaciones que tiene disponibles.

Las posibles operaciones que se pueden realizar sobre las jarras son:

- Llena_Jarra3l
- Llena_Jarra4l

- Vacia_Jarra3l
- Vacia_Jarra4l
- Echar_Parte_J3l_en_J4l
- Echar_Parte_J4l_en_J3l
- Volcar_J3l_en_J4l
- Volcar_J4l_en_J3l

Al expandir un nodo se relizan todas las operaciones posibles sobre ese nodo, y ese número de operaciones será el número de nodos que colgarán por debajo de él.

Cada nodo tiene almacenado la profundidad que tiene, la última operación que se ha hecho sobre él, y sabe también qué nodo es su padre.

El árbol se va expandiendo por niveles de profundidad hasta que se encuentra un nodo que cumple con el requisisto buscado (jarra de 4 litros tiene 2 litros), en ese momento se recorre el árbol hacia arriba desde ese nodo para saber cuales son las operaciones que se han realizado sobre él, y después se muestran por pantalla.

El ejemplo de ejecución es el siguiente:

```
XCLIPS for:         CLIPS (V6.10 07/01/98)
CLIPS> (load "/home/dchaparro/Universidad/2Cuatrimestre/IngConoc/
Practicas/Ej1/jarras.clp")
Defining deftemplate: estado
Defining defrule: Estado-Inicial +j
Defining defrule: Llena_Jarra3l +j+j
Defining defrule: Llena_Jarra4l =j+j
Defining defrule: Vacia_Jarra3l =j+j
Defining defrule: Vacia_Jarra4l =j+j
Defining defrule: Echar_Parte__J3l_en_J4l =j+j
Defining defrule: Echar_Parte__J4l_en_J3l =j+j
Defining defrule: Volcar_J3l_en_J4l =j+j
Defining defrule: Volcar_J4l_en_J3l =j+j
Defining defrule: Incrementa-cuenta-nodos =j+j
Defining defrule: actualiza-profundidad +j+j
Defining defrule: Hecho +j
Defining defrule: Completa =j+j+j
Defining defrule: Camino-Completado +j+j+j
Defining defrule: alcanza-maxima-profundidad =j+j+j
Defining defrule: FIN =j+j
TRUE
CLIPS> (reset)
CLIPS> (run)
El estado inicial es jarra3l = 0 litros y jarra4l = 0 litros

- Solucion en el nodo 528:
```

```
Llena la jarra de 4l
Echar parte de J4l en J3l
Vacia la jarra de 3l
Volcar J4l en J3l
Llena la jarra de 4l
Echar parte de J4l en J3l

- Solucion en el nodo 578:
Llena la jarra de 3l
Volcar J3l en J4l
Llena la jarra de 3l
Echar parte de J3l en J4l
Vacia la jarra de 4l
Volcar J3l en J4l

Sobrepasado el nivel de profundidad ,6.
Un total de 605 nodos en el arbol
```

Como podemos ver en los resultados, expandiendo 6 niveles conseguimos 2 posibles soluciones, una en el nodo 528 y la otra en el nodo 578.

Las soluciones encontradas son:

Solución 1:

OPERACION	Jarra3l	Jarra4l
Estado Inicial	0	0
Llenar la jarra de 4l	0	4
Echar parte de J4l en J3l	3	1
Vacia la jarra de 3l	0	1
Volcar J4l en J3l	1	0
Llena la jarra de 4l	1	4
Echar parte de J4l en J3l	3	**2**

Solución 2:

OPERACION	Jarra3l	Jarra4l
Estado Inicial	0	0
Llena la jarra de 3l	3	0
Volcar J3l en J4l	0	3
Llena la jarra de 3l	3	3
Echar parte de J3l en J4l	2	4
Vacia la jarra de 4l	2	0
Volcar J3l en J4l	0	**2**

1.3 Apartado c

Cree una regla llamada eliminar-estados-repetidos que controle si un nuevo estado creado ya existe, y si es así lo elimine. Compruebe cómo esto mejora el rendimiento del programa.

Para hacer este apartado, he añadido una regla que comprueba si existen dos nodos que tengan el mismo estado (igual número de litros en cada una de las jarras respectivamente). En caso de que se cumpla la condición se elimina el nodo con mayor profundidad. La regla comentada es la siguiente:

```
(defrule eliminar-estados-repetidos
(declare (salience 2000))
(estado
(profundidad ?p1)
(litros_j3l ?cantidad3)
(litros_j4l ?cantidad4)
)
?nodo <- (estado
(profundidad ?p2&:(< ?p1 ?p2))
(litros_j3l ?cantidad3)
(litros_j4l ?cantidad4)
 )
=>
(retract ?nodo)
)
```

Un ejemplo de ejecución del programa con esta mejora es el siguiente:

```
XCLIPS for:         CLIPS (V6.10 07/01/98)
CLIPS> (load "/home/dchaparro/Universidad/2Cuatrimestre/IngConoc/
Practicas/Ej1/jarras2.clp")
Defining deftemplate: estado
Defining defrule: Estado-Inicial +j
Defining defrule: Llena_Jarra3l +j+j
Defining defrule: Llena_Jarra4l =j+j
Defining defrule: Vacia_Jarra3l =j+j
Defining defrule: Vacia_Jarra4l =j+j
Defining defrule: Echar_Parte__J3l_en_J4l =j+j
Defining defrule: Echar_Parte__J4l_en_J3l =j+j
Defining defrule: Volcar_J3l_en_J4l =j+j
Defining defrule: Volcar_J4l_en_J3l =j+j
Defining defrule: Incrementa-cuenta-nodos =j+j
Defining defrule: actualiza-profundidad +j+j
Defining defrule: Hecho +j
Defining defrule: eliminar-estados-repetidos =j+j
Defining defrule: Completa =j+j+j
Defining defrule: Camino-Completado +j+j+j
```

```
Defining defrule: alcanza-maxima-profundidad =j+j+j
Defining defrule: FIN =j+j
TRUE
CLIPS> (reset)
CLIPS> (run)
El estado inicial es jarra3l = 0 litros y jarra4l = 0 litros
- Solucion en el nodo 14:
Llena la jarra de 4l
Echar parte de J4l en J3l
Vacia la jarra de 3l
Volcar J4l en J3l
Llena la jarra de 4l
Echar parte de J4l en J3l
- Solucion en el nodo 15:
Llena la jarra de 3l
Volcar J3l en J4l
Llena la jarra de 3l
Echar parte de J3l en J4l
Vacia la jarra de 4l
Volcar J3l en J4l

Sobrepasado el nivel de profundidad ,6.
Un total de 15 nodos en el arbol
```

Podemos observar que la mejora es muy significativa en el rendimiento del programa. En el caso anterior sin mejora el programa llegaba a expandir 605 nodos en el árbol, y las soluciones encontradas estaban en el nodo 528 y 578.

Y en este caso con la mejora el árbol solo llega a expandir 15 nodos, las soluciones encontradas están en el nodo 14 y 15.

Por tanto conseguimos que sea más eficiente en memoria y tiempo ...

Capítulo 2

Ejercicio 4: Clasificación de peces

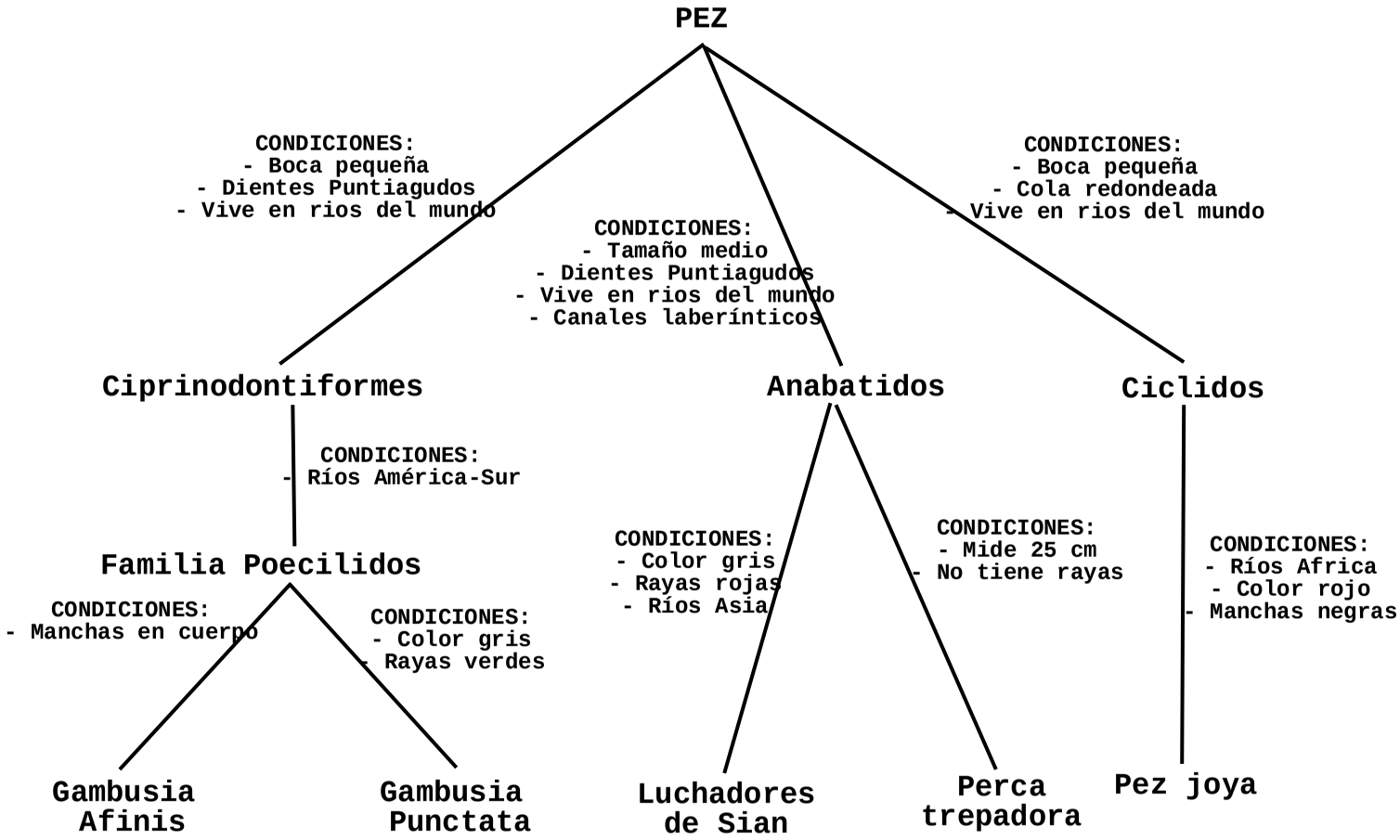
Modelizar el problema de clasificación de peces mediante un sistema de producción simulando encadenamiento hacia atrás

CLIPS no implementa el encadenamiento hacia atrás, pero éste puede ser simulado mediante reglas. A continuación se muestra una forma de simularlo. Puede hacer uso de ella y completar el sistema para que incluya la información del dominio del enunciado: El sistema de producción, en base a ciertas características de un pez, que se preguntarán al usuario, deberá informar del orden, especie, raza e información adicional sobre él, en caso de que exista.

Para este programa se han utilizado las reglas del enunciado para resolver el problema del encadenamiento hacia atrás. Éstas reglas apenas se han modificado, excepto algún que otro detalle.

Lo que se ha hecho es definir todas las reglas necesarias para la clasificación de los peces, así como las preguntas necesarias para ir alcanzando cada una de las metas.

Se ha partido e la estructura siguiente para la clasificación de los peces:


PEZ
CONDICIONES:
- Boca pequeña
- Dientes Puntiagudos
- Vive en rios del mundo
CONDICIONES:
- Tamaño medio
- Dientes Puntiagudos
- Vive en rios del mundo
- Canales laberínticos
CONDICIONES:
- Boca pequeña
- Cola redondeada
- Vive en rios del mundo
Ciprinodontiformes
Anabatidos
Ciclidos
CONDICIONES:
- Ríos América-Sur
Familia Poecilidos
CONDICIONES:
- Manchas en cuerpo
CONDICIONES:
- Color gris
- Rayas verdes
CONDICIONES:
- Color gris
- Rayas rojas
- Ríos Asia
CONDICIONES:
- Mide 25 cm
- No tiene rayas
CONDICIONES:
- Ríos Africa
- Color rojo
- Manchas negras
Gambusia Afinis
Gambusia Punctata
Luchadores de Sian
Perca trepadora
Pez joya

Las reglas definidas se estructuran según se observa en el esquema. Según se van cumpliendo condiciones en el árbol definido se va ajustando la búsqueda del pez.

Al usuario se le van preguntando las características del pez para ir ajustando la búsqueda.

Veamos ejemplos de ejecución del programa:

```
XCLIPS for:          CLIPS (V6.10 07/01/98)
CLIPS> (load "/home/dchaparro/Universidad/2Cuatrimestre/IngConoc/
Practicas/Ej4/clasificacion.clp")
Defining deftemplate: regla
Defining defrule: propagar-meta +j+j
Defining defrule: meta-satisfecha =j+j
Defining defrule: borrar-regla-que-no-equipara +j+j
Defining defrule: modificar-regla-que-equipara =j+j
Defining defrule: regla-satisfecha =j+j
Defining defrule: Hacer-pregunta =j+j
Defining defrule: Dar-Resultados =j+j
Defining deffacts: hechos_iniciales
TRUE
CLIPS> (reset)
CLIPS> (run)
Tiene la cola redondeada (si/no)??: no
Tiene tamaño medio (si/no)??: no
Vive en muchos rios del mundo (si/no)??: no
*************
El pez es: No existe en la BBDD

CLIPS> (reset)
CLIPS> (run)
Tiene la cola redondeada (si/no)??: no
Tiene tamaño medio (si/no)??: no
Vive en muchos rios del mundo (si/no)??: si
Tiene la boca pequeña (si/no)??: si
Tiene los dientes puntiagudos (si/no)??: si
Vive en los ríos de América del sur (si/no)??: no
*************
El pez es: Orden de los Ciprinodontiformes

CLIPS> (reset)
CLIPS> (run)
Tiene la cola redondeada (si/no)??: no
Tiene tamaño medio (si/no)??: no
Vive en muchos rios del mundo (si/no)??: si
Tiene la boca pequeña (si/no)??: si
Tiene los dientes puntiagudos (si/no)??: si
Vive en los ríos de América del sur (si/no)??: si
Es macho (si/no)??: si
```

```
Tiene manchas en el cuerpo (si/no)??: no
Es de color gris (si/no)??: no
*************
El pez es: Orden Ciprinodontiformes {Familia Poecilidos}(Macho con
gnopodio. Hembra 3cm más grande que el macho)

CLIPS> (reset)
CLIPS> (run)
Tiene la cola redondeada (si/no)??: no
Tiene tamaño medio (si/no)??: si
Vive en muchos rios del mundo (si/no)??: si
Tiene los dientes puntiagudos (si/no)??: si
Tiene canales laberínticos (si/no)??: si
Es de color azul (si/no)??: no
Mide 25 centímetros (si/no)??: si
Tiene rayas (si/no)??: no
*************
El pez es: Orden Anabatidos -> Perca trepadora

CLIPS> (reset)
CLIPS> (run)
Tiene la cola redondeada (si/no)??: si
Vive en muchos rios del mundo (si/no)??: si
Tiene la boca pequeña (si/no)??: si
Vive en los ríos de África (si/no)??: si
Es de color rojo (si/no)?: si
Tiene manchas negras (si/no)?: no
*************
El pez es: Orden Ciclidos

CLIPS> (reset)
CLIPS> (run)
Tiene la cola redondeada (si/no)??: no
Tiene tamaño medio (si/no)??: no
Vive en muchos rios del mundo (si/no)??: si
Tiene la boca pequeña (si/no)??: si
Tiene los dientes puntiagudos (si/no)??: si
Vive en los ríos de América del sur (si/no)??: si
Es macho (si/no)??: no
Tiene manchas en el cuerpo (si/no)??: no
Es de color gris (si/no)??: si
Tiene rayas verdes (si/no)??: si
*************
El pez es: Orden Ciprinodontiformes {Familia Poecilidos}->Gambusia
Punctata(Hembra. Macho 3cm más pequeño que la hembra)
```

Capítulo 3

Ejercicio 7: Cuatro en raya

Las cuatro en raya es un juego de mesa en el que dos jugadores realizan alternativamente su jugada colocando fichas en un tablero vertical de 7 columnas por 6 posiciones cada una de ellas. Cada jugada consiste en colocar una ficha en una columna. Esta ficha se colocará en la posición libre más baja de todas las de la columna elegida. El juego termina cuando se da una de las siguientes situaciones:

- **Se han alineado cuatro fichas del mismo color en diagonal, horizontal o vertical sin teger entre ellas ninguna ficha del otro color (cuatro en raya). En este caso gana el jugador que consigue colocar las cuatro en raya.**
- **Todas las posiciones de todas las columnas están ocupadas por fichas sin llegar a obtenerse cuatro en raya. En este caso se produce un empate.**

Se pide construir un sistema de producción que modelice el comportamiento del contrario. Además, cada vez que sea el turno del usuario se le deberá preguntar la jugada que quiere realizar, dando un mensaje de error si la jugada que éste introduce no es válida.

Las plantillas utilizadas para la resolución del problema son las siguientes:

```
(deftemplate turno (slot es (type INTEGER)(range 1 2)))
(deftemplate casilla
(slot columna (type INTEGER))
(slot fila    (type INTEGER))
(slot ficha   (type INTEGER) (range 0 2))
)
```

La primera de ellas es para saber de qué jugador es el turno en cada momento. Si tiene el valor 1 el turno corresponde a la máquina, y si tiene el valor 2 el turno es para el usuario.

La otra plantilla sirve para saber el estado del tablero en cada momento. Al comenzar la partida se insertan hechos con todas las casillas del tablero, y con

el valor de ficha 0, que indica que no hay ninguna ficha introducida. Si el valor de ficha es 1 significa que la ficha es de la máquina y si es 2 la ficha es del usuario.

Básicamente el juego consiste en lo siguiente:

- Si el turno es del usuario:
  - Se le pide la columna en la que quiere introducir la ficha.
  - Si no es posible ese movimiento se le vuelve a pedir la columna.
  - Si es posible realizar ese movimiento se modifica la casilla en la que debe ir la ficha.
  - Se le pasa el turno a la máquina.
- Si el turno es de la máquina:
  - Averigua el movimiento más adecuado en cada momento.
  - Efectúa el movimiento.
  - Le pasa el turno al usuario.

El problema principal del programa propuesto consiste en que la máquina sepa en cada momento cuál es el movimiento más acertado. Para ello se definen reglas con diferente prioridad para cada uno de los movimiento posible de la máquina.

El conjunto de reglas de movimientos de la máquina ordenados por orden de prioridad es el siguiente:

1. La máquina puede hacer 4 en raya (horizontal, vertical o diagonal). Esto ocurre cuando ya tiene 3 en raya, y tiene la opción de hacer las 4 en raya.
2. Evitar que el usuario haga 4 en raya (horizontal, vertical o diagonal). Esto ocurre cuando el usuario tiene 3 en raya, y tiene la opción de hacer las 4 en raya.
3. Evitar que el usuario haga 3 en raya (horizontal, vertical o diagonal). Esto ocurre cuando el usuario tiene 2 en raya, y tiene la opción de hacer 3 en raya.
4. Movimiento aleatorio de la máquina. Si ninguna de las reglas anteriores se cumplen, se elige un número aleatorio entre 1 y 7, y si es posible ese movimiento se efectúa. Si no lo es se vuelve a elegir otro número aleatorio.

Ejemplo de ejecución:

```
XCLIPS for:          CLIPS (V6.10 07/01/98)
CLIPS> (load "/home/dchaparro/Universidad/2Cuatrimestre/IngConoc/
Practicas/Ej7/cuatro_raya.clp")
Defining deftemplate: turno
Defining deftemplate: casilla
Defining deffacts: hechos_iniciales
Defining defrule: Estado-Inicial +j
Defining defrule: turno_usuario +j+j+j
```

```
Defining defrule: comprobar_jugada_invalida1 =j+j+j+j
Defining defrule: comprobar_jugada_invalida2 =j=j+j
Defining defrule: comprobar_jugada_invalida3 =j=j+j
Defining defrule: comprobar_jugada_valida_1 =j=j=j+j
Defining defrule: comprobar_jugada_valida_2 =j=j=j+j
Defining defrule: comprobar_jugada_valida_3 =j=j=j+j
Defining defrule: comprobar_jugada_valida_4 =j=j=j+j
Defining defrule: comprobar_jugada_valida_5 =j=j=j+j
Defining defrule: comprobar_jugada_valida_6 =j=j=j+j
Defining defrule: evitar_4enraya_horizontal_usuario_izq_fila1 +j+j+j+j+j
Defining defrule: evitar_3enraya_horizontal_usuario_izq_fila1 =j=j=j+j
Defining defrule: evitar_4enraya_horizontal_usuario_izq =j+j+j+j+j+j
Defining defrule: evitar_3enraya_horizontal_usuario_izq =j=j=j=j+j
Defining defrule: evitar_4enraya_horizontal_usuario_der_fila1 =j=j=j=j+j
Defining defrule: evitar_3enraya_horizontal_usuario_der_fila1 =j=j=j+j
Defining defrule: evitar_4enraya_horizontal_usuario_der =j=j=j=j=j+j
Defining defrule: evitar_3enraya_horizontal_usuario_der =j=j=j=j+j
Defining defrule: evitar_4enraya_vertical =j=j+j+j+j
Defining defrule: evitar_3enraya_vertical =j=j=j+j
Defining defrule: evitar_4enraya_diagonal_der_sup =j=j+j+j+j+j
Defining defrule: evitar_3enraya_diagonal_der_sup =j=j+j+j+j
Defining defrule: evitar_4enraya_diagonal_der_inf =j=j+j+j+j+j
Defining defrule: evitar_4enraya_diagonal_der_inf_fila1 =j=j=j=j+j
Defining defrule: evitar_3enraya_diagonal_der_inf =j=j=j=j+j
Defining defrule: evitar_3enraya_diagonal_der_inf_fila1 =j=j=j+j
Defining defrule: evitar_4enraya_diagonal_izq_sup =j=j+j+j+j+j
Defining defrule: evitar_3enraya_diagonal_izq_sup =j=j+j+j+j
Defining defrule: evitar_4enraya_diagonal_izq_inf =j=j+j+j+j+j
Defining defrule: evitar_4enraya_diagonal_izq_inf_fila1 =j=j=j=j+j
Defining defrule: evitar_3enraya_diagonal_izq_inf =j=j=j=j+j
Defining defrule: evitar_3enraya_diagonal_izq_inf_fila1 =j=j=j+j
Defining defrule: 4enraya_horizontal_izq_fila1 =j=j+j+j+j
Defining defrule: 4enraya_horizontal_izq =j=j+j+j+j+j
Defining defrule: 4enraya_horizontal_der_fila1 =j+j+j+j+j
Defining defrule: 4enraya_horizontal_der =j=j=j=j=j+j
Defining defrule: 4enraya_vertical =j=j+j+j+j
Defining defrule: diagonal_der_sup =j=j+j+j+j+j
Defining defrule: diagonal_izq_sup =j=j=j=j=j+j
Defining defrule: columna_aleatoria =j+j+j
Defining defrule: movimiento_maquina_1 =j+j+j+j
Defining defrule: movimiento_maquina_2 =j=j=j+j
Defining defrule: movimiento_maquina_3 =j=j=j+j
Defining defrule: movimiento_maquina_4 =j=j=j+j
Defining defrule: movimiento_maquina_5 =j=j=j+j
Defining defrule: movimiento_maquina_6 =j=j=j+j
Defining defrule: casillas_completas +j+j+j+j+j+j
Defining defrule: horizontal_maquina +j+j+j+j
Defining defrule: vertical_maquina =j+j+j+j
Defining defrule: diagonal_der_sup_maquina =j+j+j+j
Defining defrule: diagonal_izq_sup_maquina =j+j+j+j
```

```
Defining defrule: horizontal_usuario +j+j+j+j
Defining defrule: vertical_usuario =j+j+j+j
Defining defrule: diagonal_der_sup_usuario =j+j+j+j
Defining defrule: diagonal_izq_sup_usuario =j+j+j+j
Defining defrule: usuario_gana +j
Defining defrule: mostrar_tablero +j+j+j+j+j+j+j+j+j+j+j+j+j+j+j+j+j+j+j+j+j+j+j+j
+j+j+j+j+j+j+j+j+j+j+j+j+j+j+j+j+j+j+j+j+j+j+j+j
TRUE
CLIPS> (reset)
CLIPS> (run)
             ESTADO DEL TABLERO
             ==================
            1  2  3  4  5  6  7

          -------------------------
          |                       |
          |  0  0  0  0  0  0  0  |
          |                       |
          |  0  0  0  0  0  0  0  |
          |                       |
          |  0  0  0  0  0  0  0  |
          |                       |
          |  0  0  0  0  0  0  0  |
          |                       |
          |  0  0  0  0  0  0  0  |
          |                       |
          |  0  0  0  0  0  0  0  |
          -------------------------
Movimiento (columna): 4

   TU mueves a la columna 4

   EL mueve a la columna 6

             ESTADO DEL TABLERO
             ==================
            1  2  3  4  5  6  7

          -------------------------
          |                       |
          |  0  0  0  0  0  0  0  |
          |                       |
          |  0  0  0  0  0  0  0  |
          |                       |
          |  0  0  0  0  0  0  0  |
          |                       |
          |  0  0  0  0  0  0  0  |
          |                       |
          |  0  0  0  0  0  0  0  |
          |                       |
          |  0  0  0  2  0  1  0  |
          -------------------------
```

```
Movimiento (columna): 3

   TU mueves a la columna 3

   EL mueve a la columna 2

            ESTADO DEL TABLERO
            ==================
           1  2  3  4  5  6  7
         -------------------------
         |                         |
         |  0  0  0  0  0  0  0  |
         |                         |
         |  0  0  0  0  0  0  0  |
         |                         |
         |  0  0  0  0  0  0  0  |
         |                         |
         |  0  0  0  0  0  0  0  |
         |                         |
         |  0  0  0  0  0  0  0  |
         |                         |
         |  0  1  2  2  0  1  0  |
         -------------------------
Movimiento (columna): 2

   TU mueves a la columna 2

   EL mueve a la columna 5

            ESTADO DEL TABLERO
            ==================
           1  2  3  4  5  6  7
         -------------------------
         |                         |
         |  0  0  0  0  0  0  0  |
         |                         |
         |  0  0  0  0  0  0  0  |
         |                         |
         |  0  0  0  0  0  0  0  |
         |                         |
         |  0  0  0  0  0  0  0  |
         |                         |
         |  0  2  0  0  0  0  0  |
         |                         |
         |  0  1  2  2  1  1  0  |
         -------------------------
```

```
Movimiento (columna): 3

   TU mueves a la columna 3

   EL mueve a la columna 4

            ESTADO DEL TABLERO
            ==================
           1  2  3  4  5  6  7
         -------------------------
         |                         |
         |  0  0  0  0  0  0  0  |
         |                         |
         |  0  0  0  0  0  0  0  |
         |                         |
         |  0  0  0  0  0  0  0  |
         |                         |
         |  0  0  0  0  0  0  0  |
         |                         |
         |  0  2  2  1  0  0  0  |
         |                         |
         |  0  1  2  2  1  1  0  |
         -------------------------
Movimiento (columna): 5

   TU mueves a la columna 5

   EL mueve a la columna 2

            ESTADO DEL TABLERO
            ==================
           1  2  3  4  5  6  7
         -------------------------
         |                         |
         |  0  0  0  0  0  0  0  |
         |                         |
         |  0  0  0  0  0  0  0  |
         |                         |
         |  0  0  0  0  0  0  0  |
         |                         |
         |  0  1  0  0  0  0  0  |
         |                         |
         |  0  2  2  1  2  0  0  |
         |                         |
         |  0  1  2  2  1  1  0  |
         -------------------------
```

```
Movimiento (columna): 5

   TU mueves a la columna 5

   EL mueve a la columna 5

            ESTADO DEL TABLERO
            ==================
           1  2  3  4  5  6  7
         -------------------------
         |                         |
         |  0  0  0  0  0  0  0  |
         |                         |
         |  0  0  0  0  0  0  0  |
         |                         |
         |  0  0  0  0  1  0  0  |
         |                         |
         |  0  1  0  0  2  0  0  |
         |                         |
         |  0  2  2  1  2  0  0  |
         |                         |
         |  0  1  2  2  1  1  0  |
         -------------------------
Movimiento (columna): 7

   TU mueves a la columna 7

   EL mueve a la columna 3

            ESTADO DEL TABLERO
            ==================
           1  2  3  4  5  6  7
         -------------------------
         |                         |
         |  0  0  0  0  0  0  0  |
         |                         |
         |  0  0  0  0  0  0  0  |
         |                         |
         |  0  0  0  0  1  0  0  |
         |                         |
         |  0  1  1  0  2  0  0  |
         |                         |
         |  0  2  2  1  2  0  0  |
         |                         |
         |  0  1  2  2  1  1  2  |
         -------------------------
```

```
Movimiento (columna): 6

   TU mueves a la columna 6

   EL mueve a la columna 2

            ESTADO DEL TABLERO
            ==================
           1  2  3  4  5  6  7
         -------------------------
         |                       |
         |  0  0  0  0  0  0  0  |
         |                       |
         |  0  0  0  0  0  0  0  |
         |                       |
         |  0  1  0  0  1  0  0  |
         |                       |
         |  0  1  1  0  2  0  0  |
         |                       |
         |  0  2  2  1  2  2  0  |
         |                       |
         |  0  1  2  2  1  1  2  |
         -------------------------

HAS PERDIDO ...
```