Inteligencia en Redes de Comunicaciones

Ejercicios de CLIPS

Julio Villena Román, Raquel M. Crespo García,
José Jesús García Rueda
{jvillena, rcrespo, rueda}@it.uc3m.es

Universidad
Carlos III de Madrid

Estos ejercicios sirven de introducción al desarrollo de sistemas expertos en CLIPS, usando la versión de CLIPS en Java llamada Jess:

http://www.jessrules.com/

En primer lugar, aquí se plantean unos cuantos programas, que tienes que analizar y probar en Jess. En:

/var/home/asig/ircit/Jess60a5/practicas

tienes los programas CLIPS a los que hace referencia este enunciado. Para ejecutarlos:

> java -classpath /var/home/asig/ircit/Jess60a5/ jess.Main <programa>

Una vez que te hayas familiarizado con el lenguaje, aborda el diseño e implementación de un sistema experto en Jess similar a

http://es.akinator.com/

en el dominio que quieras, para lo que puedes basarte en el ejercicio 10.

Ejercicio 1. Analiza los siguientes ejemplos, que te serán luego útiles.

Hello world (en el fichero 0.clp)

```
;; Hello, world in Jess!
(printout t "Hello, world!" crlf)
```

Funciones sencillas (en los ficheros 1.clp y 2.clp)

```
(deffunction cuadrado (?x)
  (* ?x ?x)
)
(printout t (cuadrado 3) crlf)

(deffunction factorial (?n)
  (if (= ?n 0) then
    1
   else
    (* ?n (factorial (- ?n 1)))
  )
)
(printout t (factorial 4) crlf)

(reset)
(run)
```

```
(deftemplate dato
    (slot x)
    (slot y)
)
(defrule doble
  (dato (x ?x) (y =(* 2 ?x) ))
  =>
  (printout t " ?x = " ?x crlf)
)
(deffacts datos
    (dato (x 2) (y 4))
    (dato (x 3) (y 9))
)
(reset)
(run)
```

Reglas más elaboradas (en el fichero 3.clp)

```
(deffacts refrigerador
  (refrigerador luz on)
  (refrigerador puerta abierta)
  (refrigerador temperatura 12)
)
(defrule regla-frigo
  (refrigerador luz on)
  (refrigerador puerta abierta)
=>
  (printout t "La comida se ha estropeado." crlf)
  (assert (refrigerador comida estropeada))
)
(deftemplate caldera
  (slot estado)
)
```

```
(deffacts caldera
  (error-de-estado confirmado)
  (valvula cerrada)
  (temperatura alta)
)
(defrule fallo-del-sistema
  (error-de-estado confirmado)
  (or
    (and    (temperatura alta)
            (valvula cerrada)
    )
    (and    (temperatura baja)
            (valvula abierta)
    )
  )
  =>
  (printout t "El sistema tiene un problema de fluido." crlf)
)
(reset)
(run)
(facts)
```

Ejercicio 2. Relaciones familiares

Dados los siguientes patrones para hechos que describen relaciones familiares:

- (padre-de (padre <nombre>) (hijo <nombre>))
- (madre-de (madre <nombre>) (hijo <nombre>))
- (hombre <nombre>)
- (mujer <nombre>)
- (mujer-de (esposa <nombre>) (esposo <nombre))
- (marido-de (esposo <nombre>) (esposa <nombre>))

escribir reglas para inferir las siguientes relaciones:

- Hermano.
- Hermana.
- Abuelo.
- Abuela.
- Abuelos. (Abuelo y abuela que son matrimonio)
- Primo.
- Tío.
- Tía.

Aplicar las reglas desarrolladas con el siguiente ejemplo:

> *"Alberto y Belinda son una feliz pareja con dos hijos, Diana y Enrique, casados respectivamente con Carlos y Fiorina. Estas dos últimas parejas han tenido descendencia, dos hermosos chicos que han decidido llamar Gabriel e Hilario respectivamente".*

Ejercicio 3. Plantillas

Describir una plantilla para hechos que contengan información sobre un conjunto. La plantilla debe incluir información sobre el nombre del conjunto, la lista de sus elementos, y si es subconjunto de algún otro conjunto (superconjuntos que posee).

Utilizar dicha plantilla para representar los siguientes conjuntos.

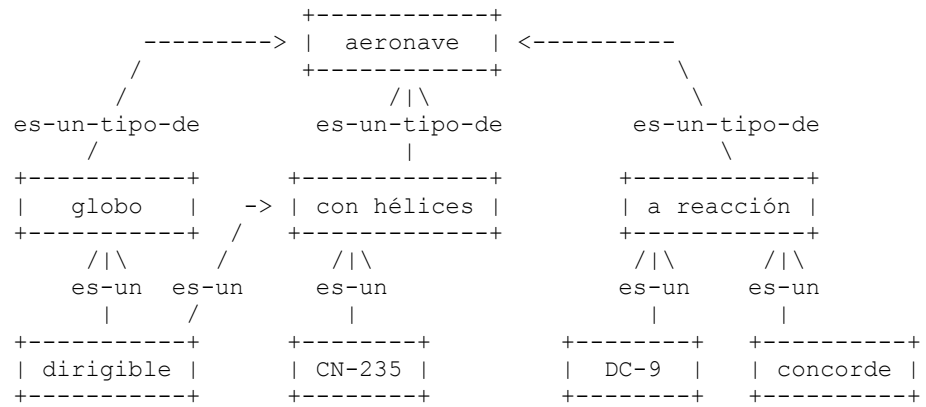
- A={1,2,3}
- B={1,2,3,4,5}
- C={rojo,verde,amarillo,azul}

- D={CPU,memoria,I/O,tarjeta de video}

¿Cómo se incluye el campo "superconjuntos" en la plantilla?
Cargar el programa en CLIPS y examinar el valor que el sistema le da al campo "superconjuntos". Crear un hecho acorde con la plantilla creada para representar conjuntos que no incluya ningún campo y examinar la forma que CLIPS le da.

Ejercicio 4. Red semántica

Convertir la red semántica de la siguiente figura en un conjunto de hechos CLIPS en una declaración "deftemplate". Utilizar varias plantillas para describir los hechos. Por ejemplo, el enlace "es-un" debe ser un nombre de relación y debe tener su propia plantilla. (No lo hagas entero)



Ejercicio 5. Objetivos

Este problema trata de alcanzar un objetivo en el mundo de los bloques apilados. De la situación inicial, en la que los bloques están apilados unos sobre otros en montones arbitrarios, se quiere encontrar la secuencia de acciones a realizar para un bloque determinado quede sobre otro (este código está en el fichero "*stack.clp*").

```
(deftemplate goal
  (slot move)
  (slot on-top-of))

(deffunction print-moved-message (?A ?B)
  "Prove that deffunctions work"
  (printout t ?A " moved on top of " ?B "." crlf))

(defrule push-value
  ?push-value <- (push-value ?value)
  ?stack <- (stack $?rest)
  =>
  (retract ?push-value ?stack)
  (assert (stack ?value ?rest))
  (printout t "Pushing value " ?value crlf))

(defrule pop-value-valid
  ?pop-value <- (pop-value)
  ?stack <- (stack ?value $?rest)
  =>
  (retract ?pop-value ?stack)
  (assert (stack ?rest))
  (printout t "Popped value " ?value " " $?rest crlf))

(defrule pop-value-invalid
```

```
  ?pop-value <- (pop-value)
  (stack)
  =>
  (retract ?pop-value)
  (printout t "Popping from empty stack" crlf))

(defrule move-directly
  ?goal <- (goal (move ?block1) (on-top-of ?block2))
  ?stack1 <- (stack ?block1 $?rest1)
  ?stack2 <- (stack ?block2 $?rest2)
  =>
  (retract ?goal ?stack1 ?stack2)
  (assert (stack ?rest1))
  (assert (stack ?block1 ?block2 ?rest2))
  (print-moved-message ?block1 ?block2))

(defrule move-to-floor
  ?goal <- (goal (move ?block1) (on-top-of floor))
  ?stack1 <- (stack ?block1 $?rest)
  =>
  (retract ?goal ?stack1)
  (assert (stack $?block1))
  (assert (stack $?rest))
  (print-moved-message ?block1 "floor"))

(defrule clear-upper-block
  (goal (move ?block1))
  (stack ?top $?X ?block1 $?Y)
  =>
  (assert (goal (move ?top) (on-top-of floor))))

(defrule clear-lower-block
  (goal (on-top-of ?block1))
  (stack ?top $?X ?block1 $?Y)
  =>
  (assert (goal (move ?top) (on-top-of floor))))

(deffacts initial-data
  (stack A B C)
  (stack D X E F)
  (goal (move C) (on-top-of E)))

(reset)
(facts)
(run)
(facts)
```

Ejercicio 6. Problemas

En el fichero "*wordgame.clp*" está resuelto el problema de las letras, que consiste en resolver puzzles numéricos. Analiza la solución.

```
  GERALD
+ DONALD
--------
= ROBERT
```

Ejercicio 7. Sistema de control de sensores

En una planta industrial hay diez sensores enumerados del 1 al 10. Cada sensor puede dar el mensaje "correcto" o "incorrecto". Construir una plantilla para representar los sensores y escribir una o más reglas que impriman en pantalla un mensaje si 3 o más sensores dan el mensaje "incorrecto".

Crear una declaración "deffacts" en la que se incluyan todos los sensores con el mensaje "correcto" por defecto.

Cargar el programa en CLIPS y probarlo en los siguientes casos:
- Los sensores 3 y 5 dan el mensaje "incorrecto".
- Los sensores 2, 8 y 9 dan el mensaje "incorrecto".
- Los sensores 1, 3, 5 y 10 dan el mensaje "incorrecto".

utilizando el comando "modify" para modificar el contenido de los hechos.

¿Cómo actúa el comando "modify"? ¿Cómo nos aseguramos de que los sensores examinados sean distintos? ¿Qué se debe hacer para evitar que el mensaje aparezca en varias ocasiones?

Ejercicio 8. Sistema experto de diagnóstico de averías

(Este problema lo tienes resuelto en el fichero "*averias.clp*". Intenta pensar un poco en la solución y luego mira el fichero y analiza el código, porque te será útil para el siguiente problema.)

Se trata de realizar un programa en CLIPS que implemente las siguientes comprobaciones rutinarias a realizar a un coche que entra en un taller, para averiguar averías fácilmente detectables y sugerir una posible reparación:

- Lo primero es comprobar si el coche arranca y funciona perfectamente (en cuyo caso no necesita ser reparado), o si arranca pero su funcionamiento no es satisfactorio. También podría ocurrir que no arrancara en absoluto.
- Otro punto a examinar es la rotación del motor: si el coche arranca, sabemos que el motor gira. Si el coche no arranca, hay que comprobarlo.
- También hay que examinar la carga de la batería. Si el motor arranca, la batería esta cargada. Si el motor no gira, hay que comprobarlo. En el caso de que la batería estuviera descargada, la SOLUCIÓN sería cargarla.
- El encendido de las bujías: si el coche arranca y funciona, sabemos que las bujías estaban bien, sin necesidad de comprobarlo. Si el coche no arranca, pero gira el motor, el encendido es defectuoso. Si el motor no gira, el encendido esta completamente estropeado. Si el coche funciona, pero no del todo bien, hay que comprobar el estado del encendido (podría ser defectuoso).

Una vez realizadas las comprobaciones de los cuatro puntos anteriores, podemos pasar a las comprobaciones finales, que nos llevarán a las reparaciones a realizar:

- En el caso de que el coche funciona, pero no satisfactoriamente, cuatro posibles comprobaciones tendrían sentido:
    - La suciedad del filtro de gasolina (SOLUCIÓN: limpiarlo).
    - Las bujías (SOLUCIÓN: cambiarlas).
    - Bloqueo en el motor (SOLUCIÓN: ajustarlo).
    - Bajada de potencia del motor.
- En el caso de que el coche no arrancara, pero el motor girara, una comprobación rutinaria sería ver si tiene gasolina (y la SOLUCIÓN, echarle gasolina).
- Si no arrancara y las bujías estuvieran defectuosas, o bien simplemente la potencia es baja, entonces hay que mirar los contactos de las bujías, que podrían estar de tres formas posibles: normales, quemados u obstruidos. Si

estuvieran quemados, la SOLUCIÓN sería cambiar los contactos. Si estuvieran obstruidos la SOLUCIÓN es limpiarlos.
- Si el motor no arrancara, la batería estuviera cargada, y el encendido estropeado, entonces hay que comprobar si funciona el mecanismo de explosión del coche. Si es así, la SOLUCIÓN es repararlo. Si no, la SOLUCIÓN es cambiar el conducto de la gasolina.
- Si con las anteriores comprobaciones no hemos sido capaces de dar una solución, entonces habrá que llevar el coche a un mecánico que sea más experto.

Además, para luego explicarlo en el informe, hay que conseguir que al finalizar el diagnostico, siempre se sepa el estado de los cuatro puntos siguientes, si es que se chequearon: batería, encendido, giro del motor y funcionamiento general del motor.

INDICACIONES:

- Como puede observarse, tenemos una serie de observaciones sobre el estado del coche, algunas deducidas y otras comprobadas. Representar los resultados de estas observaciones mediante hechos. Las comprobaciones las simularemos preguntándole a un hipotético usuario. Es bastante útil implementar las preguntas al usuario mediante funciones.
- Cuando se consigue una solución, entonces asertar la solución con un hecho de la forma (reparación "……").
- Debemos definir una regla que detecte las reparaciones, y las escribe por pantalla y otra regla que detecta que no se ha conseguido ninguna solución, y escribe el mensaje correspondiente. Otras reglas que al final escriben los puntos comprobados y su estado.
- El grueso del programa consiste en una serie de reglas que representan el conocimiento expresado en el enunciado.

Ejercicio 9. Sistema experto de diagnóstico de enfermedades

Las plantas requieren diferentes tipos de nutrientes para desarrollarse adecuadamente. Tres de los nutrientes de mayor importancia son el nitrógeno, el fósforo y el potasio. Una deficiencia en alguno de estos nutrientes puede producir varios síntomas. Escribir como reglas en CLIPS las siguientes reglas heurísticas que indican qué deficiencia se está produciendo.

- Si la planta crece muy poco entonces puede tener una deficiencia de nitrógeno.
- Si la planta tiene un color amarillo pálido entonces puede tener una deficiencia de nitrógeno.
- Si las hojas tienen un color pardo rojizo entonces la planta puede tener una deficiencia de nitrógeno.
- Si la raíz de la planta tiene poco crecimiento entonces puede tener una deficiencia de fósforo.
- Una planta con tallo fusiforme puede tener una deficiencia de fósforo.
- Una planta con color púrpura puede tener una deficiencia de fósforo.
- Un retraso en la madurez de una planta puede deberse a una deficiencia de fósforo.
- Si los bordes de las hojas aparecen chamuscados, la planta puede tener una deficiencia de potasio.
- Una planta con los tallos debilitados puede tener una deficiencia de potasio.

- Una planta con semillas o frutas marchitas puede tener una deficiencia de potasio.

La entrada del programa debe ser una descripción de los síntomas de la planta. La salida debe indicar de qué nutriente o nutrientes se ha producido una deficiencia imprimiéndolo en pantalla. Hacerlo de forma que no aparezcan en pantalla múltiples salidas indicando una misma deficiencia.

Cargar el programa en CLIPS y utilizarlo con las siguientes entradas:
- La raíz de la planta crece poco
- La planta tiene color púrpura
- La planta tiene los tallos debilitados

Ejercicio 10. Identificación de animales: árbol de decisión con aprendizaje

Has llegado a un sistema con una cierta complejidad. Consiste en un ejemplo de manejo de árboles de decisión con posibilidad de aprendizaje de nuevos elementos, en concreto es un árbol de decisión binario para identificación de animales. Está resuelto en el fichero "*animal.clp*" y hace uso de un fichero de datos externos llamado "*animal.dat*".

Analiza el programa y piensa cómo se podría modificar para que el árbol fuera n-ario.