Práctica 5
Sistemas basados en reglas

Fecha de entrega: 21 de enero del 2018

El primer objetivo de esta práctica consiste en conocer las técnicas de representación de conocimiento basadas en reglas y, más concretamente, en aprender las características básicas del lenguaje CLIPS. El segundo objetivo consiste en analizar distintas posibilidades de aplicación de los sistemas de reglas al diseño de un recomendador de compra y alquiler de pisos.

1. Toma de contacto con CLIPS

Podéis descargaros CLIPS de https://sourceforge.net/projects/clipsrules/files/CLIPS/6.30/.

En el campus tenéis disponible el archivo familia.clp. En este archivo tenemos un conjunto de hechos que establecen los descendientes directos (dd para abreviar) de cada pareja:

(dd Padre Madre Hijo/a Sexohijo/a)

Por ejemplo: (dd Juan María Rosa m) indica que Juan y María son los padres de Rosa y que Rosa es una mujer.

Para comenzar, abrimos CLIPS y cargamos el archivo familia.clp. Para poder ver cómo avanza la ejecución, abrimos las ventanas de Facts, Agenda e Instances en Window y ponemos la vista window en horizontal para poder verlo todo más claro. En Execution->Watch->Watch options activamos: fact, rules y activation.

Para comenzar la ejecución hacemos (reset) y (run). Observamos que ha pasado, que reglas se han activado, cuales se han ejecutado y qué hechos se han añadido a nuestra base de hechos.

2. Sistema de reglas para obtención de parentescos

A partir del archivo familia.clp, debéis construir un sistema de reglas en CLIPS que permita obtener los distintos parentescos entre los miembros de una familia, asumiendo que los nombres de las personas son identificadores únicos. Para ello tenéis que añadir al archivo inicial las reglas necesarias para obtener los hechos correspondientes a los siguientes parentescos: (madre X Y), (hijo X Y), (hija X Y), (hermano X Y), (hermana X Y), (abuelo X Y), (abuela X Y), (primo X Y), (prima X Y) y (ascendiente X Y).

Adoptaremos la siguiente semántica: (P X Y) significa "X es P de Y". Por ejemplo, (abuelo X Y) significa que X es abuelo de Y. Diremos que una persona X es un ascendiente de Y si es su padre/madre, abuelo/a, bisabuelo/a, etc. (la regla debe servir para cualquier nivel de ascendencia, no sólo los existentes en el ejemplo).

Al construir las reglas correspondientes a hijo, hija, hermano y hermana se utilizarán como condiciones los hechos iniciales dd. Para el resto de los parentescos no se utilizarán los hechos dd sino los hechos asertados previamente (padre, madre, hermano, etc.)

Puede utilizarse la condición (test (neq ?x ?y)) para determinar si las variables ?x e ?y tienen valores distintos.

3. Sistema de reglas para recomendación de pisos

El objetivo de la segunda parte de la práctica es el desarrollo de un recomendador de pisos de compra y alquiler. Se pretende simular el comportamiento de un vendedor de pisos, con mucha experiencia, que es capaz de recomendar uno o varios pisos en función de las necesidades, gustos y posibilidades del cliente.

La funcionalidad básica del sistema será la de recomendar uno o varios pisos adecuados a las necesidades, gustos o restricciones del cliente. Cuando no haya ningún piso adecuado, se informará de ello al usuario.

3.1. Modelado del dominio

Primero habrá que delimitar el conocimiento necesario para la implementación del recomendador. Para ello, se analizarán los distintos tipos de conocimiento necesario:

- Información sobre pisos disponibles y sus características.
- Información que es necesario obtener del cliente sobre sus necesidades, gustos y preferencias.
- Reglas de recomendación que pueden utilizarse.

Al menos existirán tres bloques de conocimiento: conocimiento sobre el cliente, conocimiento sobre los pisos y conocimiento de recomendación de pisos. Los dos primeros son de tipo factual y se representarán mediante hechos en el prototipo. El último es de tipo procedimental (qué recomendar en qué casos) y se representará mediante reglas. No se trata de que haya una correspondencia directa entre clientes y pisos, debería haber reglas que establecieran correspondencias intermedias entre las necesidades de los clientes y las características de los pisos.

3.2. Prototipo

Una vez que tengamos el conocimiento habrá que generar un primer prototipo. Se tratará de un prototipo básico que no necesite operaciones de entrada/salida de datos.

Es aconsejable que el prototipo tenga, al menos, 3 partes correspondientes a distintas etapas de la recomendación. En una primera parte se analizarían las características y gustos del usuario para determinar de qué tipo de usuario se trata y qué tipo de pisos necesita. Se podría clasificar los pisos con respecto a distintos criterios en otra parte. Y en una tercera parte se generarían las recomendaciones concretas para el usuario que sean adecuadas a sus necesidades.

En este primer prototipo, los datos correspondientes a las necesidades y gustos del usuario se establecerán directamente en un deffacts, sin necesidad de realizar operaciones de lectura. Se asumirá que el usuario no sabe qué piso elegir. Los datos relativos a los pisos disponibles se introducirán en el sistema también a través de deffacts.

Se recomienda empezar por un prototipo sin módulos y sin multislots. Cuando funcione se podrá incrementar su complejidad.

3.3. Prototipo con E/S

En este segundo prototipo se añadirán operaciones de entrada/salida que permitan obtener los datos necesarios del usuario y generar los resultados adecuadamente formateados.

La entrada/salida debe ser a prueba de fallos, comprobando que los datos introducidos por el usuario son correctos, tanto en tipo como en rango, volviendo a solicitarlos, en caso necesario, hasta que sean correctos.

La aplicación funcionará en modo continuo, repitiendo la recomendación de pisos para distintos usuarios hasta que se decida salir de la aplicación.

3.4. Versión final

Se mejorará el prototipo anterior ampliando el conocimiento disponible y haciendo que sea fácilmente mantenible ante los cambios más habituales (por ejemplo, nuevos pisos, cambio de alguna característica de alguno de ellos, etc.) y ante la aparición de nuevos perfiles de clientes.

El prototipo imprimirá las recomendaciones correspondientes a cada cliente, seleccionando las mejores según un criterio fijo o elegido por el cliente (por ejemplo, por precio, por grado de adecuación a las necesidades del cliente, etc.). Las recomendaciones aparecerán ordenadas, por orden decreciente, de acuerdo a dicho criterio.

Si el número de recomendaciones es alto, se mostrarán sólo las N mejores. Si no hay recomendaciones para un cliente, o son muy escasas, se podrá dar la posibilidad de relajar algunas de las restricciones para aumentar su número.

Solo se utilizarán ifs de forma muy puntual en alguna deffunction. Tampoco se utilizarán las prioridades ni se eliminarán reglas para simular un programa secuencial. Se podrán utilizar prioridades, módulos, hechos centinela o valores por defecto para garantizar que se completa una etapa del programa antes de iniciar la siguiente.

Sólo se utilizará test cuando se requiera hacer comparaciones de menor que, mayor que, negaciones o funciones or. Las comparaciones de igualdad se realizarán mediante encaje de patrones con los hechos disponibles en la memoria de trabajo.

3.5. Enlaces interesantes

- https://www.idealista.com/
- http://www.fotocasa.es/es

4. Memoria

La memoria de la práctica deberá incluir:

- El resultado de la ejecución del sistema de obtención de parentescos, agrupado por tipo de parentesco.
- Descripción del recomendador de pisos, conocimiento utilizado y estructura modular.
- Ejecución del recomendador para dos usuarios diferentes. Cada ejecución incluirá los hechos iniciales que se han utilizado y los resultados que se han obtenido.

5. Entrega de la práctica

La práctica se entregará a través del Campus Virtual. Se habilitará una nueva tarea: **Entrega de la Práctica 5** que permitirá subir un zip que contendrá la memoria de la y los ficheros clp con la solución de los parentescos y del recomendador de pisos.

El fichero subido deberá tener el siguiente nombre: **Practica5GXX.zip**, siendo XX el nombre del grupo. Por ejemplo, *Practica4G03.zip*.

Uno sólo de los miembros del grupo será el encargado de subir la práctica.

La fecha límite para entregar la Práctica 5 será el **21 de enero a las 23:55**. Se recomienda no dejar la entrega para el último momento para evitar problemas de última hora.

No se corregirá ninguna práctica que no cumpla estos requisitos.