Práctica 2

Ingeniería del Conocimiento

Curso 2016/2017

Francisco Javier Caracuel Beltrán

76440940A

3º A – Grupo 3

Grado en Ingeniería Informática – Ciencias de la Computación e Inteligencia Artificial

Índice

1. Resumen

El sistema experto desarrollado se ha realizado en *CLIPS* y consta solo de un fichero *.clp* llamado *práctica2.clp*.

Este programa comienza mostrando un menú con tres opciones:

Elección de licencia de software.

Compatibilidad de software a usar con la licencia de mi software.

Asesoramiento sobre ley de protección de datos.

Dependiendo de la opción seleccionada se ejecutan una serie de reglas que hacen que el usuario pueda navegar por el programa con el fin de recibir la información que solicita. El punto final del programa siempre será este menú, que permitirá terminar completamente la ejecución con una cuarta opción para salir de él.

El archivo *.clp* cuenta con una serie de reglas que, pese a que estén todas en el mismo fichero, se encuentran bien estructuradas.

Normalmente, la ejecución se realiza activando o desactivando las reglas correspondientes al flujo que se quiera seguir. Se explica con mayor detalle el funcionamiento de cada opción en siguientes apartados.

2. Descripción del proceso seguido para el desarrollo

a. Obtención del conocimiento manual:

En el apartado primero de la práctica se pide ofrecer la licencia que mejor se ajusta a las características de un software. Para ello se ha analizado la tabla resumen que se encuentra en el fichero *LicenciasSoftware.pdf* y se ha hecho una selección de 14 licencias y 10 características que se creen que pueden ser representativas y válidas para el desarrollo de esta primera parte de la práctica.
La tabla resultante con la información es la siguiente (0 indica que no se cuenta con la característica, 1 si cuenta con la característica y 2 se desconoce).

	C1. Libre	C2. Copyleft	C3. Patentes	C4. Ficheros propietarios	C5. Añadir software comercial	C6. Abierta	C7. Propiedad intelectual	C8. Permisiva	C9. Compatible GNU	C10. Compatible OSI
E1. AFL	1	0	1	2	2	2	2	2	0	1
E2. Apache Software	1	2	1	2	2	1	2	2	0	1
E3. APSL	1	2	2	1	2	2	2	2	0	1
E4. BSD Modificada	1	2	2	2	2	1	2	2	1	1
E5. CDDL	1	0	1	2	2	2	1	2	0	1
E6. CPL	1	2	1	2	2	2	2	2	0	1
E7. EPL	1	2	1	2	2	2	2	2	0	1
E8. GPL	1	1	2	2	2	1	2	2	1	1
E9. MPL	1	1	2	2	2	2	2		0	1
E10. OpenLDAP	1	0	2	2	2	2	2	1	1	0
E11. OSL	1	1	2	2	2	1	2	2	0	1
E12. PHP	1	0	2	2	2	2	2	2	0	1

E13. Python	1	2	2	2	2	2	2	2	1	1
E14. W3C Software	1	2	2	2	2	2	2	2	1	1

Para disminuir el tamaño de esta tabla, se eliminan los atributos en los que todas las licencias son iguales. Los atributos que se eliminan son *Libre* y *Añadir software comercial*. Queda establecido que todas las licencias que se tienen en el conocimiento son libres.

Se entiende que los atributos que no se ha especificado que formen parte de una licencia, no la forman, por lo que se cambia el valor 2 (desconocido) a 0 (No).

También se van a eliminar licencias que son iguales entre ellas para reducir el coste de la rejilla de repertorio. Las licencias que se eliminan son EPL, CPL (igual que AFL) y Python (igual que W3C Software).

La tabla final con la que se trabaja es:

	C1. Copyleft	**C2. Patentes**	**C3. Ficheros propietarios**	**C4. Abierta**	**C5. Propiedad intelectual**	**C6. Permisiva**	**C7. Compatible GNU**	**C8. Compatible OSI**
E1. AFL	0	1	0	0	0	0	0	1
E2. Apache Software	0	1	0	1	0	0	0	1
E3. APSL	0	0	1	0	0	0	0	1
E4. BSD Modificada	0	0	0	1	0	0	1	1
E5. CDDL	0	1	0	0	1	0	0	1
E6. GPL	1	0	0	1	0	0	1	1
E7. MPL	1	0	0	0	0	0	0	1
E8. OpenLDAP	0	0	0	0	0	1	1	0
E9. OSL	1	0	0	1	0	0	0	1
E10. PHP	0	0	0	0	0	0	0	1
E11. W3C Software	0	0	0	0	0	0	1	1

Para ejecutar la técnica de clustering se necesita un criterio que mida la distancia entre pares de elementos. Se usará la suma de la diferencia en valor absoluto entre los atributos de los elementos.

A continuación, se realiza la rejilla de repertorio, creando una matriz triangular del tamaño de las licencias y marcando en azul las licencias que se seleccionan en cada iteración (serán aquellas que se agrupen). Con este sistema, las licencias que se agrupan serán aquellas cuyo valor sea mínimo, lo que implica que la diferencia entre ellas es mínima con respecto a las demás.

Este proceso finaliza cuando solo se tengan dos grupos en la matriz.

	E1	E2	E3	E4	E5	E6	E7	E8	E9	E10	E11
E1		1	2	3	1	4	2	4	3	1	2
E2			3	2	2	3	3	5	2	2	3
E3				3	3	4	2	4	3	1	2
E4					4	1	3	3	2	2	1
E5						5	4	5	4	2	3
E6							2	4	1	3	2
E7								4	1	1	2
E8									5	3	2
E9										2	3
E10											1
E11											

	E3	E4	E6	E7	E8	E9	E11	[E1, E2, E5, E10]
E3		3	4	2	4	3	2	1
E4			1	3	3	2	1	2
E6				2	4	1	2	3
E7					4	1	2	1
E8						5	2	3
E9							3	2
E11								1
[E1, E2, E5, E10]								

	E3	E7	E8	E9	[E1, E2, E5, E10] = α1	[E4, E6, E11]
E3		2	4	3	1	2
E7			4	1	1	2
E8				5	3	1
E9					2	1
[E1, E2, E5, E10] = α1						1

[E4, E6, E11]						

	E3	**E8**	**[E4, E6, E11] = α2**	**[E7, E9, α1]**
E3		4	2	1
E8			1	3
[E4, E6, E11] = α2				1
[E7, E9, α1]				

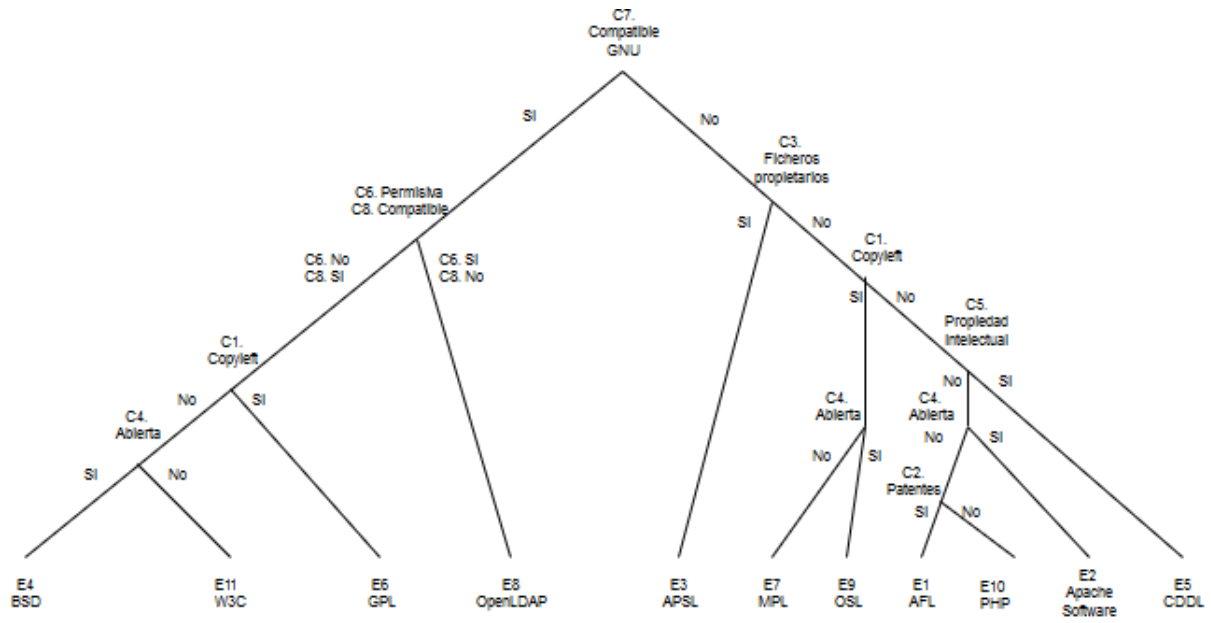
	E3	**[E7, E9, α1] = α3**	**[E8, α2]**
E3		1	2
[E7, E9, α1] = α3			1
[E8, α2]			

	[E8, α2]	**[E3, α3]**
[E8, α2]		1
[E3, α3]		

Se realiza el árbol, comprobando en sentido inverso las licencias que se han agrupado. Cada nuevo agrupamiento implica una nueva rama en el árbol.

Para saber qué pregunta se debe hacer en cada ramificación, se deben agrupar las licencias que forman ambos grupos. Se tendrá un atributo que será igual en todas las licencias de un mismo grupo y el contrario en todas las del otro grupo.

Cuando ya se han obtenido las licencias, el árbol generado es el siguiente:



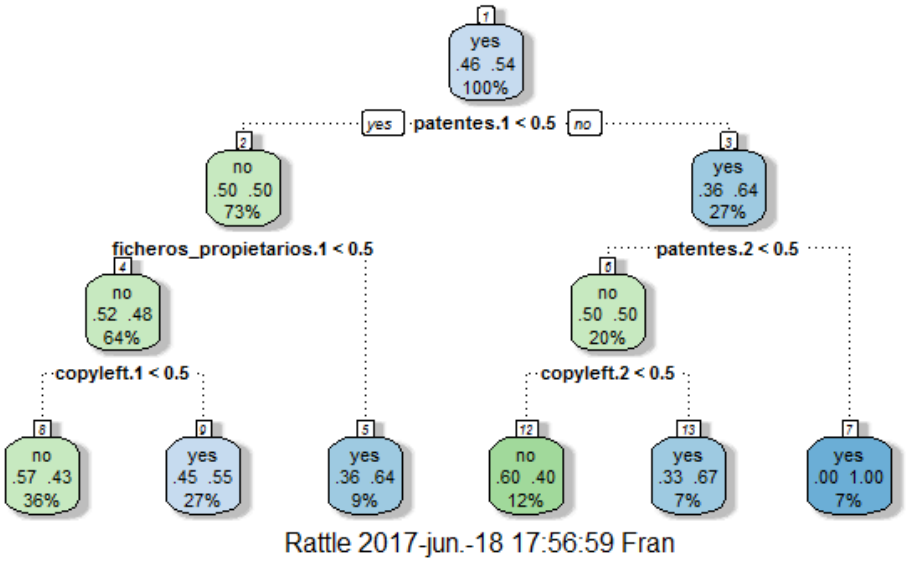
b. Obtención del conocimiento automático:

Se comienza con un archivo .R facilitado con un ejemplo.
Existe un fichero licenses.csv desde el que se leen los datos para generar el árbol. Ese fichero licenses.csv contiene la misma tabla que se ha generado en el apartado anterior.

Lo que se debe hacer es tomar la decisión de qué atributo es el que se considera que nos permite determinar si una licencia es compatible con otra. El atributo que se cree más representativo para realizar este proceso es *Compatible GNU*, por lo que los pasos que se deben realizar para generar el árbol de decisión son:

1. Leer el fichero con los datos.
2. Generar una tabla donde se comparen dos licencias entre sí en cada fila. Se deben comparar todas las licencias con todas.
3. Generar una nueva columna que indique si son compatibles ambas licencias. Ambas licencias serán compatibles siempre y cuando su valor en el atributo *Compatible GNU* sea el mismo.
4. Eliminar ambas columnas *Compatible GNU* de las licencias para que no interfieran en el proceso.
5. Entrenar la muestra, basándose en la nueva columna creada con la compatibilidad.
6. Mostrar el árbol generado.

Una vez realizados los pasos anteriores, el árbol generado es el siguiente:



3. Descripción del sistema desarrollado

a. Variables de entrada:

El sistema no cuenta con ninguna variable de entrada.

b. Variables de salida:

La única salida que ofrece el sistema es la propia que aparece por pantalla o los archivos ARCO que se generan en el apartado 3.

c. Conocimiento global del sistema:

El apartado 3 es el único que cuenta con carga de hechos inicialmente.

El conocimiento que se carga es el correspondiente a las leyes que justifican el tratamiento de la información, nombre de los ficheros ARCO que se generan y datos de la propia "empresa" que almacenará la información.

También se crea una función que devuelve la fecha actual, que será insertada en los ficheros ARCO.

d. Especificación de los módulos que se han desarrollado:

Solo se cuenta con un fichero .clp que contiene todas las reglas necesarias para su funcionamiento. El fichero se puede separar en tres partes, correspondientes a los puntos que se requieren en esta práctica.

Para la primera parte, se utiliza el conocimiento derivado del árbol de decisión generado con la rejilla de repertorio. El objetivo de esta parte es aconsejar al usuario qué licencia debería elegir para su proyecto.

En la segunda parte, se utiliza el conocimiento derivado del árbol de decisión generado automáticamente con el programa en R. El objetivo de esta parte es decidir si dos licencias son compatibles entre sí.

Para la tercera parte, se utiliza el conocimiento como si se fuera un experto y se crean reglas que se derivan del experto. El objetivo de esta parte es informar al usuario de todos los datos correspondientes a la LOPD y generar los ficheros ARCO con los datos rellenados.

e. Estructura de funcionamiento del esquema de razonamiento del sistema:

Para las tres partes que componen esta práctica se mantiene un mismo esquema de razonamiento.

Lo que se pretende hacer es seguir un flujo de información a través de reglas y hechos.

Cuando se comienza, se muestra un menú con varias opciones. Dependiendo de las opciones seleccionadas, se inserta un hecho u otro, activándose de este modo una regla concreta, correspondiente a la opción seleccionada.

Cuando se está en plena navegación de una opción se continúa con el mismo procedimiento. Dependiendo de las respuestas que ofrezca el usuario, se insertarán unos hechos determinados que permiten crear un flujo coherente y llegar al fin de cada opción.

f. La lista de hechos que utiliza el sistema durante su funcionamiento y la forma de representarlos:

Las reglas que utiliza el sistema en general son:

- init: se ejecuta al inicio e introduce el hecho *showMenu*, que muestra el menú con las tres opciones.
- Menu: muestra el menú con las tres opciones. Dependiendo de la opción que se elija introduce el hecho *chosenOption número*, con la que ya las reglas coincidirán dependiendo de cada parte de la práctica.
- option4: finaliza el sistema y muestra un mensaje.

g. Hechos y reglas de cada módulo:

a) Parte 1:

Se utiliza el árbol generado en la rejilla de repertorio. Se encadenan reglas que contienen las preguntas que se deben hacer hasta llegar a un nodo hoja. Será en este punto cuando se aconseje la licencia que se debe utilizar.
Las reglas de esta sección se pueden dividir en dos partes:

- Preguntas correspondientes al árbol: conforme se van haciendo preguntas se insertan hechos que codifican el camino que está siguiendo el usuario. Si se hace una pregunta y la respuesta es positiva, se concatena 1 al hecho, en caso contrario se concatena 0. De este modo, es muy sencillo crear las reglas correspondientes a cada pregunta del árbol.
  Por cada respuesta, se guarda un hecho con la selección para poder justificarlo posteriormente.

- Licencia aconsejada y razones: cuando se ha llegado a un nodo hoja, ya no existen hechos que permitan continuar con las preguntas y una regla con menos *salience* que las reglas de las preguntas informará de la licencia que se ha seleccionado. Como se han ido guardando los motivos por los que iba seleccionando una licencia u otra, existe otra regla con menos *salience* que la que muestra la licencia aconsejada, que indica los motivos.

b) Parte 2:

El procedimiento de la segunda parte es exactamente igual que el de la parte primera, pero en este caso se utiliza el árbol generado automáticamente con el programa en R y con sus respectivas preguntas.

c) Parte 3:

Es la parte más extensa y comienza introduciendo unos hechos que se necesitan para su funcionamiento, como por ejemplo el nombre de los ficheros ARCO o las leyes que son necesarias para justificar las medidas a tomar en cuanto al tratamiento de los datos.

Por motivos de disponibilidad de tiempo, los datos que se solicitan o que se gestionan internamente en la generación de los archivos ARCO son mínimos. Se ha querido demostrar el procedimiento que se debe seguir, utilizando los mínimos datos posibles y sabiendo que para ampliar esta información solo se debe continuar con la estructura existente. Así se ha decidido, también, para la justificación de las medidas que se deben tomar en cuanto a la ley, utilizando solo varios ejemplos y asumiendo que, si se desea incorporar toda la información, solo se deben ampliar.

Esta sección comienza mostrando una lista con los distintos tipos de datos existentes. Se ha creado una variable de tipo lista, que permite acceder fácilmente al tipo de dato que se desea solo con su numeración. Para guardar los datos y poder agruparlos, se introducen unos hechos que siguen una codificación. Cada intervalo de datos corresponde a un grupo distinto y se ha creado un *switch/case* que permite seleccionar cuáles de ellos deben pertenecer a un grupo u otro. Se ha tomado la decisión de utilizar esta instrucción porque se cuenta con nueve grupos diferentes y anidar nueve grupos utilizando *if/else* resulta más que engorroso. Los hechos que se introducen siguen el siguiente patrón: chosen-3X DATO, donde X es el número de subgrupo al que pertenece y DATO es el dato seleccionado.

Cuando ya se han insertado todos los datos que ha seleccionado el usuario, comienza un conjunto de reglas anidadas, que para seguir un orden se van activando conforme se ejecutan. Con estas reglas, se muestran los distintos datos seleccionados por subgrupos.

Cuando ya se han mostrado los datos agrupados, aparece un menú para seleccionar el tipo de organización que utilizará los datos, lugar, etc. También se sigue la estructura de reglas anidadas para mostrar los menús en orden.
El siguiente conjunto de reglas se encarga de mostrar qué datos identifican a una persona sin necesidad de recurrir a más datos, cuáles pueden identificar si se realiza una combinación de varios y cuáles no identifican por sí mismos.
Como cuando se han seleccionado los datos ya se han agrupado, los datos que identifican unívocamente a una persona pertenecen al grupo 2, por lo que todos esos datos se mostrarán con su correspondiente mensaje. Para indicar los datos que posiblemente puedan identificar a una persona, se crean tantas reglas como combinaciones de datos se consideren oportunas que puedan identificar a una persona. Para finalizar, se indica que el resto de datos no pueden identificar a una persona por sí mismos.

Con respecto a las medidas que se deben tomar y su justificación, quizás es la parte más sencilla porque solo se debe mostrar un texto. Para la justificación se han creado hechos con las leyes y cuando una medida requiere de dicha ley, se realiza un *assert* con el nombre de la ley (que debe coincidir con la insertada al inicio). De este modo la siguiente regla a ejecutarse será la que determina la ley.

El último punto de la práctica es la generación de los archivos ARCO. Cuando se selecciona la opción de generar los archivos ARCO, se solicitan al usuario sus datos. Solo se piden una vez, guardando como hechos los datos insertados. Lo ideal en este punto es utilizar una plantilla que se pueda leer desde CLIPS y añadir los datos correspondientes a cada archivo. Se ha encontrado un problema en este punto y se ha decidido insertar el contenido en la regla, leyendo los datos que se deben utilizar de los hechos y guardándolos en un fichero .txt que se generará en el mismo directorio desde el que se ejecuta el fichero práctica2.clp.

4. Manual de uso del sistema

El sistema funciona de una manera muy sencilla. Solo es necesario cargar el fichero práctica2.clp en CLIPS y ejecutar (reset) y (run).

Aparecerá un menú y se debe seleccionar el número correspondiente a la acción que se quiere realizar. Cuando el sistema haga preguntas, informará de las teclas que se deben utilizar para su respuesta.