16 de Enero de 2018

Samuel Soutullo Sobral

Universidade de Santiago de compostela

Grao en Enxeñaría Informática

Sistema de recomendación en CLIPS

Memoria del proyecto

Tabla de contenidos

[1 Base teórica del sistema de recomendaciones 2](#_Toc503902011)

[2 Implementación del SBR 3](#_Toc503902012)

[2.1 Plantillas 3](#_Toc503902013)

[2.2 Reglas 4](#_Toc503902014)

[3 Integración con Java 6](#_Toc503902015)

[4 Estructura de ficheros del proyecto 8](#_Toc503902016)

# Base teórica del sistema de recomendaciones

Se ha desarrollado un sistema de recomendaciones para una tienda de productos, tal y como se pedía en el enunciado de la práctica. El sistema trata de recomendar a cada uno de los usuarios productos que les puedan interesar, basándose en su historial de compras. En consecuencia, el sistema no realizará ninguna recomendación hasta que se hayan realizado compras, y será más preciso cuantas más compras se hayan realizado.

Para realizar las recomendaciones, el sistema tiene en cuenta las etiquetas de los productos. Las etiquetas (o *tags*) son palabras clave que describen semánticamente cada uno de los productos de la tienda. Por ejemplo, productos como *escoba* o *lejía*, comparten etiquetas como *hogar* o *limpieza*; mientras que para productos como *tarjeta gráfica* etiquetas válidas podrían ser *informática* o *tecnología*. Finalmente, otro ejemplo que podría considerarse una especie de combinación de los dos anteriores es *robot de limpieza* cuyas etiquetas podrían ser *hogar, limpieza* y *tecnología*.

Para cada usuario, existe un registro que almacena las etiquetas de los productos que ha comprado. De esta forma, si el usuario compra el producto *lejía*, se añadirá dicho registro personal las etiquetas *hogar* y *limpieza*. Si posteriormente, compra una *tarjeta gráfica*, se añadirán también las etiquetas *informática* y *tecnología*, resultando en un registro personal que contiene las cuatro etiquetas mencionadas.

A la hora de realizar las recomendaciones para un usuario en particular, se compara su registro de etiquetas con las etiquetas de cada uno de los productos. Si existe un producto cuyas etiquetas sean un subconjunto del registro de etiquetas del usuario, dicho producto aparecerá en las recomendaciones del usuario.

Siguiendo el ejemplo mencionado en párrafos anteriores, supóngase que el usuario ha comprado los productos *lejía* y *tarjeta gráfica*. Su registro de etiquetas en este momento viene dado por el conjunto . Recuérdese también que las etiquetas del producto *robot de limpieza* vienen dadas por el conjunto . Como se cumple que , el producto *robot* *de limpieza* aparecerá en las recomendaciones del usuario.

Dicho esto, cabe aclarar que existen condiciones adicionales para que un producto aparezca en las recomendaciones de un usuario. En particular, aunque se cumpla el criterio de etiquetas descrito en los párrafos anteriores, un producto no aparecerá en las recomendaciones de los usuarios, si se cumple alguna de las siguientes condiciones:

1. **El usuario ya ha comprado el producto**: es muy raro que un usuario compre un mismo producto dos veces, teniendo en cuenta que se trata de una tienda de carácter genérico.
2. **No hay stock del producto**: no tiene sentido recomendar un producto a un usuario en un momento en el que éste no lo puede comprar. En todo caso, esto sería contraproducente de cara al negocio.

# Implementación del SBR

## Plantillas

Se han definido las siguientes plantillas:

|  |
| --- |
| (deftemplate user  (slot id)  (slot name)  (multislot purchased\_products)  (multislot products\_tags)  )  (deftemplate product  (slot id)  (slot name)  (slot stock)  (multislot tags)  )  (deftemplate shop\_order  (slot user\_id)  (slot product\_id)  )  (deftemplate recommendation  (slot user\_id)  (slot product\_id)  ) |

La primera de ellas se corresponde con el usuario. El campo *id* sirve para identificar al usuario de manera interna, permitiendo el correcto funcionamiento del sistema. El campo *name* sirve para identificar el usuario de manera legible por un humano. El campo *purchased\_products* almacena un registro de los productos comprados por el usuario, para que estos no aparezcan en sus recomendaciones. El campo *products\_tags* es el registro de etiquetas de usuario explicado en la sección anterior.

La segunda de ellas se corresponde con el producto. Los campos *id* y *name* son análogos a los campos homónimos de la plantilla de usuario. El campo *stock* lleva cuenta de las existencias del producto en la tienda: si este campo es igual a cero, el producto está agotado. El campo *tags* es un registro que almacena las etiquetas del producto.

La tercera de ellas es una petición de compra. Básicamente, para comunicar al sistema que el usuario con *id x* ha comprado el producto con *id* *y*, se debe añadir un hecho de este tipo, con el campo *user\_id* puesto al valor *x*, y el campo *product\_id* puesto al valor *y*.

Por último, la cuarta es una recomendación. Los hechos correspondientes a esta plantilla serán generados automáticamente por el SBR a medida que se realizan compras. Si existe un hecho de este tipo donde el *user\_id* es *x* y el *product\_id* es *y*, entonces el sistema recomienda el producto *y* al usuario *x*.

## Reglas

Se han definido las siguientes reglas:

|  |
| --- |
| (defrule buy  ?sp <- (shop\_order (user\_id ?uid) (product\_id ?pid))  ?u <- (user (id ?uid) (purchased\_products $?pp) (products\_tags $?upt))  ?p <- (product (id ?pid) (tags $?pt) (stock ?pstock)) =>  (retract ?sp)  (modify ?u (purchased\_products $?pp ?pid) (products\_tags $?upt $?pt))  (modify ?p (stock (- ?pstock 1)))  )  (defrule invalid\_buy  ?sp <- (shop\_order) =>  (retract ?sp)  )  (defrule clear\_recommendations\_already\_purchased  ?u <- (user (id ?uid) (purchased\_products $?pp))  ?p <- (product (id ?pid))  ?r <- (recommendation (user\_id ?uid) (product\_id ?pid))  (test (member$ ?pid $?pp)) =>  (retract ?r)  )  (defrule clear\_recommendations\_out\_of\_stock  ?p <- (product (id ?pid) (stock 0))  ?r <- (recommendation (product\_id ?pid)) =>  (retract ?r)  )  (defrule out\_of\_stock  ?p <- (product (stock 0)) =>  (retract ?p)  )  (defrule update\_history\_recommendations  ?u <- (user (id ?uid) (purchased\_products $?pp) (products\_tags $?upt))  ?p <- (product (id ?pid) (tags $?pt))  (test (and (not (member$ ?pid $?pp)) (subsetp $?pt $?upt))) =>  (assert (recommendation (user\_id ?uid) (product\_id ?pid)))  ) |

La primera de estas reglas se encarga de procesar una solicitud de compra. Dada una solicitud, un producto, y un usuario, correspondientes entre sí, se añade el producto a la lista de productos comprados por el usuario, las etiquetas del producto al registro de etiquetas del usuario, se reduce el stock del producto en una unidad, y se elimina el hecho correspondiente a la solicitud de compra, ya que esta acaba de ser procesada.

La segunda regla se encarga de eliminar solicitudes de compra no válida, esto es, que referencien a un producto o usuario inexistentes. Como se puede ver, se trata de una regla que simplemente elimina la primera solicitud que encuentre. Este esquema funciona correctamente porque las solicitudes válidas ya se eliminan al ser procesadas por la primera regla, quedando solo las no válidas.

La tercera regla se encarga de eliminar las recomendaciones en caso de que éstas se correspondan con un producto ya comprado por el usuario. Esta situación se puede dar si un producto aparece en el panel de recomendación del usuario, y posteriormente, dicho usuario compra este producto. En esta situación, de no realizar ninguna acción a mayores, la recomendación seguiría apareciendo a pesar de que el usuario ya ha comprado el producto. Por tanto, esta regla corrige este tipo de situaciones, eliminado las recomendaciones que cumplan dichas condiciones.

La cuarta regla es similar a la tercera. En este caso, se eliminan las recomendaciones correspondientes a productos agotados. Esta situación se puede dar si un producto aparece en el panel de recomendaciones de un usuario, y otros usuarios realizan compras de tal forma que agotan las existencias de dicho producto.

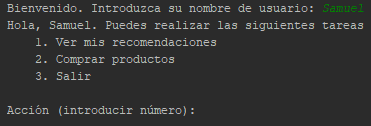
La quinta regla elimina el producto de la base de hechos cuando su *stock* llega a cero, dado que no se puede comprar.

Finalmente, la sexta regla es la que se encarga de realizar las recomendaciones. Básicamente, se comprueban las condiciones explicadas en apartados anteriores (que el usuario no haya comprado el producto, y que las etiquetas del producto sean un subconjunto del registro de etiquetas del usuario). Si se dan dichas condiciones, se añade la recomendación del producto para el usuario.

# Integración con Java

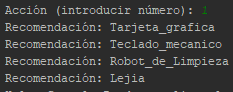
Se ha codificado una pequeña aplicación de consola en *Java* que permite probar de manera más cómoda el sistema de recomendaciones. Para ello, se ha usado la librería *CLIPSJini*, que permite integrar *CLIPS* con *Java*. Como la codificación en *Java* no es una competencia de esta asignatura, no se realizará ninguna explicación del código implementado. Si se dará, sin embargo, una breve explicación del funcionamiento de la aplicación, de cara al usuario.

Al ejecutar la aplicación, se cargará por defecto el archivo *CLP*, con todas sus reglas y la base de hechos inicial. Si se desea partir de una base de hechos diferente, bastará con modificar dicho archivo. Lo primero que se pide, es el nombre de usuario. Con el nombre de usuario que se introduzca, se podrán realizar compras y ver recomendaciones. En caso de que el usuario exista en la base de hechos, se asociará la sesión a dicho usuario. Si éste no existe, se añadirá el usuario a la base de hechos antes de iniciar la sesión.

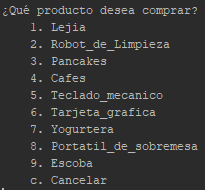


Una vez introducido el nombre de usuario, se mostrará el menú principal. Las acciones que se pueden realizar están etiquetadas mediante números. Para realizar una acción bastará con introducir el número correspondiente.

Por ejemplo, si se elige la acción *ver mis recomendaciones* para el usuario *Samuel*, se obtendrá lo siguiente. Nótese que dichas recomendaciones se realizan en base a los datos de la base de hechos inicial declarada en el fichero *CLP*.



Si se escoge la opción *comprar productos*, se mostrará un submenú similar al siguiente:



Bastará con teclear el número del producto correspondiente para realizar la compra. Si no se desea realizar la compra, se puede introducir el carácter “c”. Al realizar la compra se mostrará un mensaje de éxito y se actualizarán las recomendaciones en consecuencia.

Finalmente, para cerrar la sesión bastará con teclear el número correspondiente a la opción *salir* en el menú principal (3).

# Estructura de ficheros del proyecto

* *src*: carpeta que contiene el código fuente de la implementación en *Java*.
* *CLIPSJNI.dll*: librería necesaria para *CLIPSJNI* en Windows. En caso de ejecutar en otra plataforma será necesario copiar la librería correspondiente.
* *engine.clp*: código en *CLIPS* del sistema de recomendación.
* *RecommenderSystem.jar*: ejecutable en formato *JAR* de la aplicación codificada en *Java*.

Para ejecutar el proyecto usar el comando: *java –jar RecommenderSystem.jar*.