Memoria Algoritmo A Estrella

Nerea Jiménez González
Yhondri Acosta Novas

Abril 2020

UNIVERSIDAD
COMPLUTENSE
MADRID

Índice

Índice de figuras

1. Detalles de la implementación

1.1. Lenguaje utilizado

Para la implementación, se utiliza el lenguaje JAVA.

1.2. Procedimiento seguido para la implementación

En un primer lugar, resolvemos el problema que se da en el enunciado de la práctica para poder utilizarlo de base.
Para la lectura del archivo, creamos una clase *Reader*, en la cual implementamos la lectura de los archivos de entrada.
Para la construcción del árbol final, implementamos una clase *Node*, cuyo constructor recibe un *String*, que será el valor seleccionado tras aplicar el **ID3**, y otro nodo para seguir en una futura recursión.
En la clase *ID3* es donde implentamos el algoritmo como tal.

El método id3 recibe 2 parámetros: *mData* que contiene los ejemplos de los atributos, y *restoAttributos* que contiene los atributos sobre los que se va a iterar.
Lo primero que nos encontramos en el método son los casos base, es decir, los casos que al cumplirse terminan esa rama recursiva y "regresa" desde donde se ha realizado la llamada.

- Caso base1: La lista restoAttributos está vacía, en este caso "regresamos"
- Caso base2: Todos los ejemplos de mData son del mismo signo, en este caso "regresamos

En caso de no cumplir ninguno de los casos base, seguimos.

1. Obtenemos el elemento que minimice el mérito, esto lo hacemos llamando al método *getBestAttribute*. El método *getBestAttribute* itera sobre cada uno de los atributos. Por cada atributo itera obteniendo el mérito y comparando quedándose en la variable *bestAttribute* el atributo con mejor mérito. Una vez ha terminado de iterar sobre todos los atributos, devuelve el mejor atributo obtenido
2. Obtenemos y guardamos en una lista (*noRepeatedValues*) los atributos del mejor atributo elegido en el paso anterior

3. En otra lista, filtramos el resto de atributos de la lista de atributos

4. En un map llamado *partitionDataMaxValue*, guardamos las filas de los ejemplos del mejor atributo elegido en el paso 1. Creamos un map llamado *nodes* que contendrá los nodos del árbol.

5. Iteramos sobre los valores de la lista *noRepeatedValues*. En este bucle hacemos la llamada recursiva que nos devolverá el subárbol (conjunto de nodos), de la rama de este atributo. El nodo devuelto en la llamada recursiva lo añadiremos al mapa Nodes creado en el punto 4

6. Finalmente regresamos devolviendo el Nodo compuesto del mejor atributo y el el mapa nodes que contiene el resto de nodos elegidos en esa rama.

Siguiendo estos pasos conseguimos la recursividad completa del algoritmo

1.3. Ampliaciones realizadas

Para la explicación de la recursividad, vamos a utilizar el atributo **Temperatura** como si fuese el atributo mejor de la iteración.
Una vez hemos elegido el primer nodo, aplicamos la recursividad para calcular los subárboles.
Calculamos los nuevos atributos a partir del nodo elegido siguiendo los siguientes pasos:

1. En la variable *bestAttribute* guardamos el valor de nodo atributo seleccionado, por ejemplo **Temperatura**

2. A continuación filtramos para quedarnos con los atributos de que aún no han sido seleccionados, siguiendo el ejemplo serían **Humedad, Viento y Jugar** y los guardamos en una lista de strings llamada *newRestoAttributos*

3. Obtenemos los atributos del atributo elegido, de **Temperatura** serían, caluroso, frío, templado y lo guardamos en un set de *Strings* llamado, *noRepeatedValue*s

4. Iterando sobre *noRepeatedValues*, dejamos en una lista llamada *partition* los atributos restantes de todos los elementos de *noRepeatedValues* excepto el mejor

Finalmente, devolvemos el valor de id3(partition, newRestoAtrtributos).

2. Código ejecutable

Se adjunta en el archivo formato zip, con nombre **Código**.

2.1. Simulación

Al ejecutar p2.jar se abre la siguiente ventana.

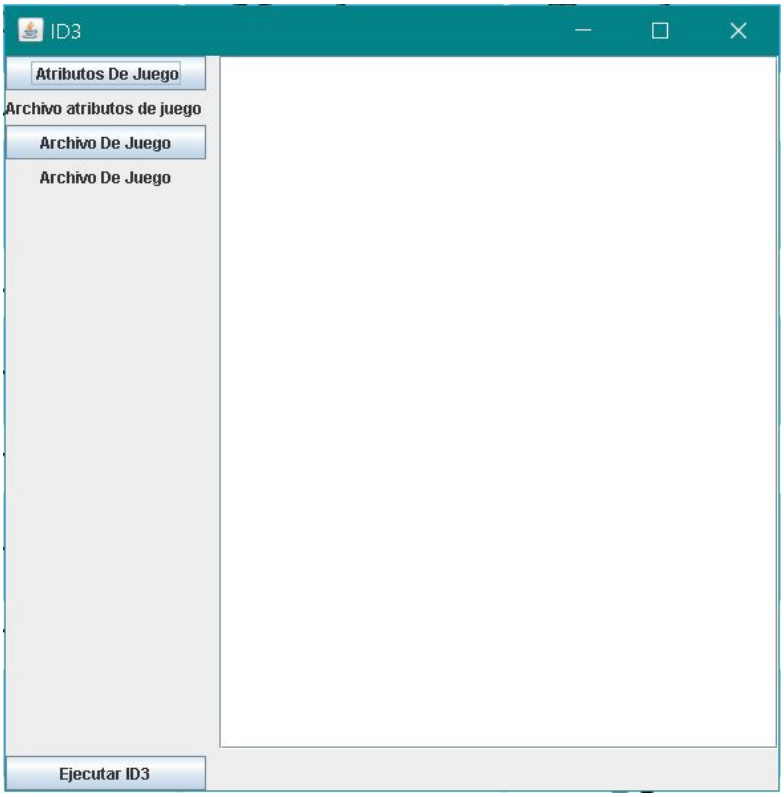

Figura 1: Ventana inicial

Elegimos los archivos en los cuales está la información para poder ejecutar el algoritmo.

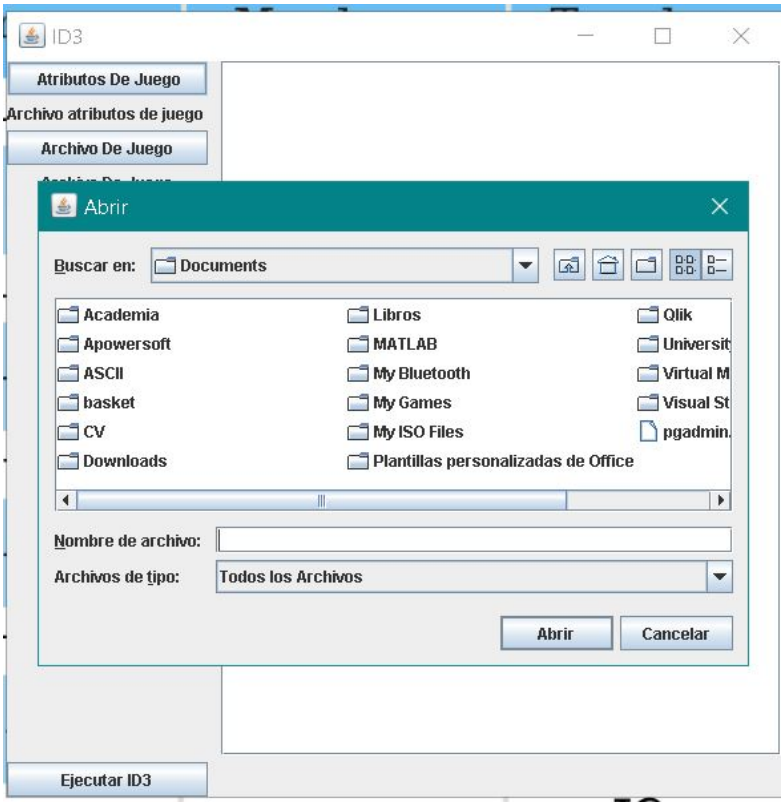

Figura 2: Ventana elección de archivos

Tras elegir ambos archivos, clickeamos el botón de Ejecutar ID3 y en la parte derecha de la ventana podemos ver el resultado tras aplicar el algoritmo. Hemos intentado recrear el árbol que se originaría:

- ->: indica el atributo seleccionado
- = : indica el valor que toma el atributo seleccionado

Si no se llega a seleccionar el valor Jugar, significa que no disponemos de suficiente información para continuar por esa rama.

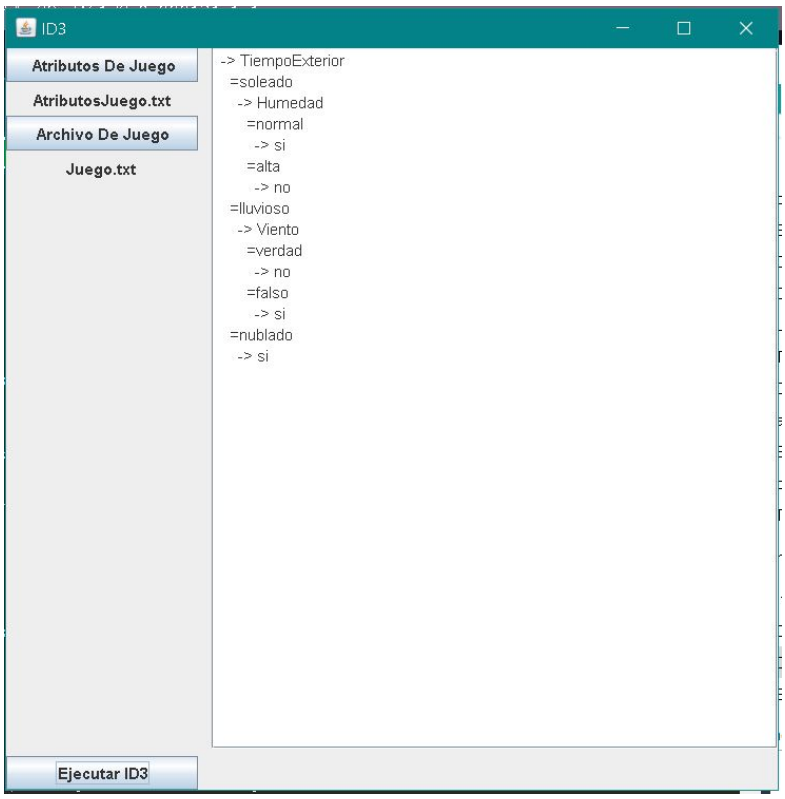

Figura 3: Ventana final

En este ejemplo, el primer atributo seria **TiempoExterior**. Según sea el valor de este atributo:

- **soleado:** en este caso no hemos acabado la recursión. El siguiente nodo en el árbol sería el atributo **Humedad**. Y con esta elección ya terminariamos esta rama.
- **lluvioso:** en este caso no hemos acabado la recursión. El siguiente nodo en el árbol sería el atributo **Viento**, y con esto terminariamos esta rama.
- **nublado:** siempre se juega si se tiene este valor.

De esta forma obtendriamos el árbol de decisión.

3. Manual de usuario

En el zip adjunto, hay un archivo ejecutable denominado **p2.jar**. Ejecutamos este archivo con doble click.
Una vez aparezca la venta, clickeamos en el botón *Atributos de Juego* para seleccionar el archivo que contiene los atributos del Juego. Después, clickeamos en el segundo botón *Archivo de Juego* para seleccionar el archivo de Juego.

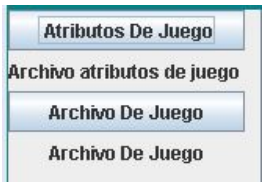

Figura 4: Botón para selección de archivos

Una vez seleccionados ambos archivos, clickeamos el botón de la esquina inferior izquierda *Ejecutar ID3*.

Figura 5: Botón para ejecutar ID3

Tras clickear este botón, en la parte derecha de la ventana nos apareceran los atributos y sus valores elegidos en cada iteración, como hemos visto en el apartado de Simulación.

Memoria Práctica 2
Abril 2020
Ult. actualización 29 de abril de 2020