MODELOS DE ORGANIZACIÓN DE BASES DE CASOS UTILIZANDO AGRUPAMIENTOS Y ÁRBOLES B

Natalia Martínez S.
María Matilde García L. *

RESUMEN

Las investigaciones actuales en el campo del razonamiento basado en casos se dirigen a facilitar el proceso de recuperación de los casos, módulo fundamental de un sistema basado en casos, para lo cual se desarrollan técnicas diversas entre las que se encuentran los esquemas indizados.

A fin de facilitar el proceso de creación de la base de casos y tomando en cuenta la necesidad de aumentar la velocidad en la recuperación de estos, y al no existir un procedimiento general que permita organizar las bases de casos, este trabajo propone tres modelos de organización que utilizan alguno de los enfoques propuestos en la literatura consultada.

ABSTRACT

The present researches in the field of case–based reasoning are headed to look for facilities in the process of cases recovering, which is a fundamental module in a case-based system.

To solve this problem several techniques are developed, among which the indexed schedules are found.

In order to facilitate the creation of the base of cases, three models of organizations are proposed in this paper. It is considered that there is no general process for the organization of the base of cases.

INTRODUCCIÓN

Al igual que otros sistemas basados en el conocimiento, los sistemas basados en casos (**SBC**) tienen dos componentes: la base de casos (**BC**) y el método de solución de problema (**MSP**) que usan el razonamiento basado en casos (**RBC**), donde se construyen las soluciones a nuevos problemas a partir de soluciones conocidas en casos similares al que se desea resolver. El RBC es un paradigma de solución de problemas que se focaliza sobre la recuperación, revisión y reutilización de soluciones de historias para resolver problemas nuevamente presentados [KOL93], [AAM94].

* Grupo de Inteligencia Artificial. Centro de Estudios de Informática Universidad Central de Las Villas. Santa Clara. Villa Clara. CUBA.

Considerando su forma, un caso puede ser un simple concepto o un conjunto conectado de subcasos; éste puede ser una instancia específica o generalizada [GUA94]. Una BC (memoria) es una estructura que representa y relaciona casos individuales o genes. El conjunto de problemas resueltos (que pudieron ser resueltos con éxito o no) forma la base de casos o memoria permanente del sistema. El diseño de la BC incluye la compilación del vocabulario de términos usados para describir los rasgos del problema, la selección de los rasgos apropiados para indexar los casos (si se usan índices) y la especificación de los esquemas usados para almacenar los casos. Lo esencial para todo el trabajo de un SBC es que todos los casos relevantes (semejantes, similares) al nuevo problema puedan recuperarse eficientemente desde la BC.

De forma común a todas estas representaciones está el problema de cómo organizar los casos en la base, organización que dependerá de la representación que se les de y que se encuentra estrechamente ligada al proceso de recuperación de casos semejantes. A esa organización se le denomina Modelo de la Base de Casos.

El modelo puede ir desde usar directamente como memoria una base de datos (relacional, jerárquica, etc.) hasta tener una organización propia de los elementos de la misma. Está claro que emplear una base de datos como memoria simplifica el trabajo de crear la misma pero afecta las propiedades deseadas para una memoria. Para obtener una organización con las características antes relacionadas todo indica que la solución descansa en tener una estrategia de generalización que cree y mantenga una jerarquía en la cual se organice la información, que sea incremental, es decir, a medida que se añaden nuevas instancias a la memoria se realizan mejores generalizaciones, y que sea pragmática en el sentido de que ningún elemento de la jerarquía se elimine por un simple contraejemplo; cuando se encuentren varias instancias que estén en desacuerdo con la generalización se deben apartar las partes malas y mantener las buenas de la generalización. Para esto se puede mantener para cada componente de la generalización un nivel de confidencia que indique cuantas veces fue confirmado éste y cuantas veces contradicho.

En esto intervienen dos aspectos principales del sistema: cómo están organizados los casos en la base y cómo se realiza la comparación entre las descripciones del caso y el problema por resolver. Por eso se dice que la recuperación tiene dos momentos principale: el acceso a los casos y la selección del, o los, casos similares

Lo esencial para el trabajo de un SBC es que todos los casos relevantes (similares, semejantes) al nuevo problema puedan ser recuperados desde la base de casos eficientemente.

La representación y manejo de grandes BC no son tareas triviales y no han sido suficientemente elaboradas hasta hoy. La mayoría de los trabajos sobre representación de los casos están relacionados con aplicaciones particulares.

1. DISEÑO DE DISTINTAS VARIANTES DE ORGANIZACIÓN DE BASES DE CASOS

En la búsqueda de la eficiencia en los sistemas basados en casos, se desarrollan distintas técnicas que faciliten un acceso rápido a los casos relevantes de determinadas aplicaciones en grandes bases de casos [LEN97]. Recientemente, y a través de la literatura consultada, vemos modelos que incluyen esquemas indizados, árboles k-d, modelos CRASH y redes de recuperación de casos.

Para lograr que la velocidad de recuperación no sea afectada por el volumen de la memoria permanente, la organización tiene que permitir acceso eficiente a los elementos más relevantes del problema en cuestión. Por eso la organización de la memoria se vincula estrechamente con la segunda componente del RBC. La medida en que se alcance una combinación eficiente de ambas componentes determinará el éxito o el fracaso del sistema con esta clase de razonamiento.

La manera más simple de identificar los casos más apropiados es realizando la búsqueda de los casos más cercanos o semejantes. Para ello se compara la descripción del nuevo problema (sus rasgos) con los correspondientes rasgos de cada

caso en la base, pero ésta es una operación costosa y el costo crece con el tamaño de la misma. Para evitar esta búsqueda exhaustiva se pueden organizar los casos en la base usando índices, los cuales usualmente son los rasgos más discriminantes de los casos. El recuperador compara solamente los índices con el nuevo problema y recupera solamente aquellos cuyos índices se corresponden con él. Una mejora adicional se logra si los índices se organizan jerárquicamente. De esta forma no se realiza la comparación de todos los casos exhaustivamente.

La selección del conjunto de rasgos es una cuestión central en el razonamiento basado en casos pues incide tanto en la definición del modelo para la base de casos como en el modelo de recuperación de éstos. El conjunto de rasgos determina qué información será almacenada en memoria para cada uno de sus elementos, la cual debe permitir la posterior recuperación de los casos semejantes dada la descripción de un nuevo problema. Dentro del conjunto de rasgos seleccionados, no todos poseen la misma importancia. Estas diferencias en la importancia de los rasgos tienen que ser consideradas al comparar dos objetos del mismo universo. La comparación de la descripción del problema por resolver, y que se utiliza como patrón de búsqueda con cada elemento de la memoria permanente, se realiza rasgo a rasgo. Para esto es necesario que una vez se hayan seleccionado los rasgos, para cada uno se determine su dominio de definición y luego el criterio de comparación de sus valores.

En este trabajo se proponen tres modelos de organización de la base de casos: estructura plana, usando la variante de índices a partir de una red neuronal que define los índices; árboles B y formación de grupos, atendiendo a los niveles de semejanza y utilizando para ello técnicas de reconocimiento de patrones.

1.1 Empleo de árboles B (B-tree) para organizar bases de casos

White en [WHI96] plantean que la mayoría de los sistemas de bases de datos proporcionan en la actualidad un grado modesto de escalabilidad y eficiencia al evaluar la similitud entre objetos utilizando uno o más rasgos. Estos rasgos se escogen cuidadosamente a fin de reducir al mínimo el tiempo requerido para computar la similitud entre objetos y reducir el espacio de almacenamiento requerido mientras se ofrece una medida de similitud exacta.

La idea más común utilizada para mejorar la eficiencia es emplear una jerarquía de representaciones que permita que la mayoría de los objetos en la base de datos puedan ser rápidamente eliminados de la consideración. Se escogió el empleo de árboles B para organizar bases de casos , pues los sistemas que manejan cantidades impresionantes de datos necesitan representaciones eficientes que permitan una búsqueda rápida; por lo general utilizan para ello los archivos indizados [HEK92].

Un archivo indizado consta de un conjunto de registros, donde cada uno contiene una clave única y algunos datos. Como mínimo, un archivo indizado debe contar con las siguientes operaciones:

- Búsqueda: Dada una clave, encontrar el dato asociado a ella.
- Inserción: Añadir un nuevo registro al archivo.
- Eliminación: Dada una clave, retirar el registro con esa clave.

Los árboles B representan una de las maneras más eficientes de implementar un archivo indizado [WIR87], [HEK92]. Un árbol B consta de un conjunto de nodos, donde cada uno puede tener hasta $2n$ registros y $2n+1$ hijos. El número n se llama orden del árbol. Cada nodo del árbol (con excepción de la raíz) debe tener al menos n registros. Esto garantiza que se usa al menos el 50% de la capacidad de almacenamiento. Además, un nodo no extremal que contenga m registros debe tener exactamente $m+1$ hijos.

Las estructuras de datos subyacentes reciben el nombre de árboles B y tienen las siguientes características (se dice que n es el orden del árbol B):

1. Cada página contiene a lo sumo $2n$ elementos (claves).

2. Cada página, excepto la de la raíz, contiene n elementos por lo menos.

3. Cada página es: una página de hoja, o sea que no tiene descendientes, o bien tiene $m+1$ descendientes, donde m es su número de claves en esta página.

4. Todas las páginas de hoja aparecen al mismo nivel.

Si el orden es ascendente, todas las claves de un nodo aparecen en orden creciente de izquierda a derecha, y para cada clave k, las claves del subárbol izquierdo son menores que k y las claves del subárbol derecho son mayores que k.

Los datos asociados a una clave pueden aparecer en el nodo, junto a la clave, o por separado, guardando únicamente su dirección (un apuntador) en el nodo. Esta última forma es más flexible porque puede permitir el uso de registros de longitud variable.

La propiedad más importante de un árbol B es que las operaciones de inserción y eliminación están diseñada para que el árbol permanezca balanceado todo el tiempo. En general, un árbol balanceado tiene una altura menor que la de uno no balanceado. Esto puede reducir en forma significativa el tiempo de búsqueda.

Otra cualidad positiva de la organización del árbol B es su conveniencia y economía en el caso de una actualización puramente secuencial de la base de datos completa. Cada página se mete en la memoria primaria exactamente una vez. Tal estructura incluye algoritmos relativamente simples de búsqueda, inserción y eliminación.

La figura 1 muestra la estructura de datos utilizada. Se empleó el tamaño más estándar, cuatro campos con cinco enlaces , pues para la búsqueda divide la cantidad de comparaciones por realizar entre cuatro, con lo que se hace mucho más rápida.

La clave mediante la cual nuestro modelo forma la jerarquía es una selección de los rasgos por su importancia, que da las opciones de utilizar alguno de los métodos expuestos en la literatura.

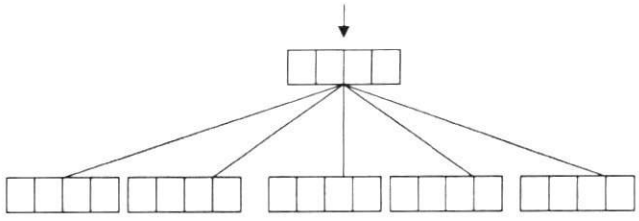

Figura 1. Estructura del árbol B

El problema con los árboles de búsqueda es que su rendimiento depende decisivamente de que la entrada sea aleatoria. Si éste no es el caso, el tiempo de ejecución se incrementa significativamente, hasta el punto en que los árboles de búsqueda se vuelven más costosos que las listas enlazadas.

1.2 Algoritmo para la organización de casos basado en los agrupamientos

La variante seleccionada para aplicar un sistema de índices fue dividir el fichero de casos en grupos mediante un algoritmo de agrupamiento o formación de clusters. No sólo basta tener un fichero de casos divididos en grupos, ya que de alguna forma se necesita seleccionar sobre que grupo se realizará la búsqueda. Una manera de solucionar este problema es tener un representante por cada grupo; de esta forma el módulo de recuperación compara el nuevo problema con el representante de cada grupo y la búsqueda se realizará sobre la clase del representante más semejante a él. La organización dada a la BC le garantiza al módulo de recuperación un acceso eficiente a los casos relevantes o candidatos a semejantes. Para recuperar los casos semejantes se realiza una búsqueda utilizando funciones de semejanza.

1.2.1 Algoritmo para dividir la base de casos en grupos

El algoritmo que seleccionamos para dividir la BC en grupos fue el Algoritmo CLASS perteneciente a un grupo de algoritmos en la rama de reconocimientos de patrones, aunque el mismo no se aplica tal y como se definió.

Existen varios algoritmos que nos permiten formar grupos. Tales son el caso de algoritmos de redes neuronales artificiales como Kohonen[HIL95], algoritmos de cluster lineal[RUI93] y el CLASS que se basa en la creación de grupos βo— compactos[RUI93], entre otros que pueden ser aplicados para resolver este problema. Se seleccionó el CLASS [RUI93] por ser un algoritmo sencillo, que necesita muy poca información sobre el dominio, y tiene poca complejidad desde el punto de vista computacional.

Descripción del Algoritmo CLASS con modificaciones

En este algoritmo serán utilizadas las siguientes notaciones:

MI → Matriz de *m* filas por *k* columnas, donde *m* es la cantidad de objetos y *k* la de rasgos de los objetos. En esta matriz se almacena el valor de cada objeto, donde un objeto para nosotros será un caso.

MS → Matriz de semejanza (*mxm*). El elemento *i,j* de esta matriz almacena un valor que corresponde con el grado de semejanza entre el objeto *i* y el objeto *j*. Donde: MS[i,i] = 1 y M[i,j] = M[j,i] si la función de semejanza empleada es simétrica.

I(Oi) → Descripción estándar del objeto *i*, en este vector están los valores asociados a los rasgos de objeto *i*; esta información se corresponde con una fila de la matriz.

β → Función de semejanza. Esta función tiene dos parámetros que son los objetos por comparar.
$\beta(I(O_i), I(O_j)) \rightarrow [0,1]$.

Algoritmo

Paso 1- *Se calculan los valores de β(I(Oi),I(Oj)) para todos los pares posibles de MI, es decir se forma MS.*

Paso 2- *En [RUI93] se plantean varias variantes para calcular βo, nosotros utilizamos:*

$$\beta_0 = \frac{1}{m(m-1)} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \beta(I(O_i), I(O_j))$$

Paso 3- *Se obtiene* $\beta^*_i = max\ \beta(I(Oi),I(Oj))$ *donde* $j=1..m,\ i \neq j$

Aquí i recorre las filas y j las columnas de la matriz MS.

Paso 4- *La magnitud* $\beta^*_{i\ (i=1..m)}$ *se compara con* β_0. *Si* $\beta^*_i < \beta_0$ *entonces el objeto* O_I *se considera un objeto aislado y conforma un conjunto aislado y unitario. Si* $\beta^*_i \geq \beta_0$ *se va al paso 5.*

Paso 5- *Para cada fila de la matriz MS se forma un conjunto compuesto por los índices t tales que* $\beta(I(Oi),I(Oj)) = \beta^*_i$. *Son formados los m conjuntos.*

Fin del algoritmo

La organización de la base de casos como resultado de la aplicación del algoritmo CLASS, sólo los primeros cinco pasos, es decir el primer nivel de agrupamiento, debido a que nos interesan los primeros compactos que son precisamente los de menor cardinalidad.

La aplicación de este algoritmo no presupone que se conozca previamente cuantos grupos se formarán. La estructuración no depende del orden en que se escojan los objetos para agruparlos y además para un β_0 *determinado siempre se obtendrá la misma agrupación.*

1.2.2 Selección del representante por cada grupo

Para seleccionar el representante de cada grupo seleccionamos el holotipo del grupo, que será el elemento de mayor tipicidad [RUI93], es decir el objeto al que más se parecen los demás objetos de su grupo.

Se seleccionó esta variante para la elección del representante por grupo a partir de la sugerencia de [RUI93] donde plantea la factibilidad de la combinación entre estos dos algoritmos, CLASS y HOLOTIPO.

Se calcula la tipicidad de cada objeto del grupo formado por:

$$t(O_i) = t_i = \frac{\beta_i}{\frac{1}{|G(O_i)|-1} \sum_{j \neq i} (\beta_i - \beta_{ij})^2}$$

donde

$j = 1. . . \mid G(O_i) \mid$

β_{ij} : *Forma abreviada de* $\beta(I(O_i),(I(O_j))$

$\mid G(O_i) \mid$: *Cardinalidad del grupo.*

$G(O_i)$: *Grupo formado a partir del algoritmo CLASS, objetos que conforman un* βo *– compacto.*

1.2.3 Resumen

La base de casos quedará como una estructura jerárquica de dos niveles , un primer nivel con los holotipos y un segundo nivel con los grupos determinados a partir de la aplicación del CLASS, como se muestra en la figura 2.

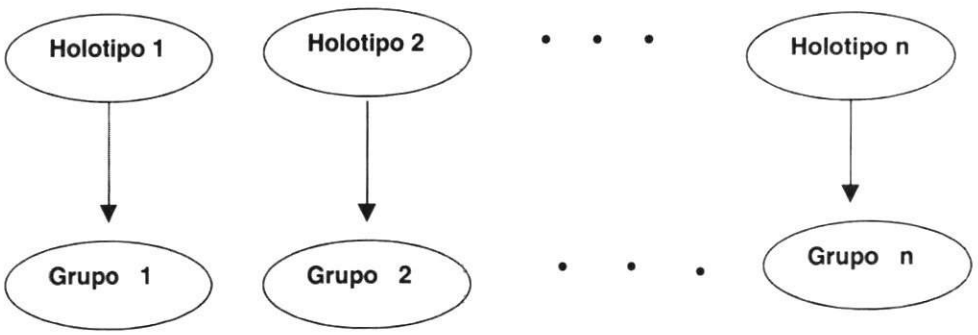

Figura 2. Niveles de agrupamiento según la semejanza

Para el acceso a los casos con mayor semejanza al nuevo caso se siguen estos pasos:

- Comparar el nuevo caso con todos los casos del primer nivel (holotipos de cada grupo), utilizando una función de semejanza.
- Seleccionar aquel cuya magnitud de semejanza sea mayor.
- Recuperar todos los casos correspondientes al holotipo seleccionado.

2. COMPORTAMIENTO ASINTÓTICO DE LOS ALGORITMOS PROPUESTOS

Hasta este punto hemos expuesto algunos modelos de organización de bases de casos sustentados cada uno de ellos en algoritmos clásicos o en algoritmos que proponemos como aspecto novedoso en este trabajo. En este epígrafe haremos un breve análisis de la complejidad computacional de cada uno de los modelos, que nos permita establecer una comparación entre los mismos, fundamentalmente en cuanto a tiempo de ejecución (complejidad Temporal , CT) y una pequeña referencia al espacio de memoria (complejidad espacial, CE). Para ello, y de manera informal, daremos un nombre a cada uno de los modelos que nos permita referirnos a ellos de forma rápida.

A1- Modelo de organización de Bases de casos utilizando Arboles B (B-Tree)

A2- Modelo de organización de Bases de casos utilizando Agrupamientos

2.1 Modelo de organización de bases de casos utilizando Arboles B (B-Tree)

La idea de este modelo, como se planteó anteriormente, es utilizar una estructura de árbol B para ir ubicando índices a la base de casos ya sea a la izquierda o a la derecha de cada nodo, en dependencia de la semejanza entre los casos. Para cada caso, la cantidad máxima de comparaciones que se hace no es mayor que la cantidad de niveles del árbol que se va creando, pues en cada nivel no se hace más de una comparación. Esto significa que en el peor de los casos la complejidad temporal de este algoritmo viene dada por la siguiente fórmula:

$$CT(A1) = O(M \log_2 M)$$

Para hacer un análisis de la complejidad espacial de este modelo debemos tener en cuenta que se trabaja

con la base de casos que debe estar almacenada en una base de datos y una estructura de datos para almacenar el árbol B que se crea. Podríamos incluir en este análisis el tamaño de los nodos del árbol pero según el tipo de problema este valor se mantiene constante y además no es objetivo de trabajo hacer un análisis detallado de la complejidad espacial del modelo.

2.2 Modelo de organización de bases de casos utilizando agrupamientos

En este modelo se utiliza un enfoque de reconocimiento de patrones para ir formando clusters con el conjunto de casos. Para ello se aplican los algoritmos CLASS y HOLOTIPO. El algoritmo CLASS se utiliza para formar los grupos. Aquí se calcula primero, la matriz de semejanza como resultado de la aplicación de la función de semejanza a todos los pares de casos. Esta es una matriz simétrica pues la semejanza entre el objeto Oi y el objeto Oj es la misma que la semejanza entre el objeto Oj y el objeto Oi (existen funciones de semejanza no simétricas que producen matrices no simétricas). En el peor de los casos este paso del algoritmo aporta un término a la expresión de la complejidad que es proporcional a M2 . Luego se encuentran los valores βi con los que se efectuará la comparación para ir formando los grupos. Esto significa que se hace, en el peor de los casos, un recorrido a la matriz de semejanza, lo cual aporta otro término proporcional a M2.

El algoritmo HOLOTIPO encuentra el holotipo de cada uno de los grupos formados; o sea el caso más representativo de cada grupo. Esto significa recorrer cada uno de los grupos y dentro de cada grupo, cada uno de los casos. Como la cantidad de casos es M, nunca se recorrerá un número de casos mayor que M; más bien, siempre se recorre los M casos. Esto aporta a la expresión final un término proporcional a M.

La expresión de la complejidad temporal de este modelo viene dada por la suma de los términos que aportan los procesos que explicamos anteriormente

$$CT(A2) = O(M^2) + O(M^2) + O(M)$$

Aplicando las reglas de la notación asintótica obtenemos la siguiente expresión para la complejidad temporal de este modelo:

$$CT(A2) = O(M^2)$$

La complejidad espacial de este algoritmo viene dada por la base de datos que contiene inicialmente los casos, una estructura de datos que almacena la matriz de diferencia y una estructura de datos para almacenar los grupos y los casos que pertenecen a cada grupo. Al igual que en los modelos anteriores no nos detenemos a hacer un análisis exhaustivo de la complejidad temporal pues no es objetivo de este trabajo.

BIBLIOGRAFÍA

[AAM94] AAMODT, AGNAR AND ENRIC PLAZA, *"Foundational issues, methodological variations, and system approach"*, AI Communications, 7(1), March 1994.

[GUA94] GUARDATI, S. *"Razonamiento Basado en Casos"*. Soluciones Avanzadas. Año 2 Nro. 13. Sept. 1994.

[HEK92] HEKMATPOUR, S., *"Guía para Programadores en C"*. Prentice-Hall Hispanoamerica, S.A., 1992.

[HIL95] HILERA, JOSÉ RAMÓN Y J. V. MARTÍNEZ. *"Redes Neuronales Artificiales. Fundamentos, Modelos y Aplicaciones"*. RA-MA Editorial, Madrid, España 1995.

[KOL93] KOLODNER, JANET L., *"Case-Based reasoning"*. Morgan Kaufmann Publishers, San Mateo, California, 1993.

[LEN97] LENZ, MARIO., *"Case Retrieval Nets applied to Large Case Bases"*. Dept. of Computer Science, Humbold University Berlin, 1997.

[RUI93] RUIZ, J., *"Modelos Matemáticos para el Reconocimiento de Patrones"*. Edit. UCLV, 1993.

[WIR87] WIRTH, N., *"Algoritmos y Estructura de Datos"*. Prentice-Hall Hispanoamerica, S.A., 1987.