UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE CIENCIAS

VNIVERSIDAD NACIONAL
AVTNOMA DE
MEXICO

# ANTITESTORES PARA EL CALCULO DE LOS TESTORES DE UNA MATRIZ

T E S I S

QUE PARA OBTENER EL TITULO DE

MATEMATICO PRESENTA

**ALEJANDRO MIER GONZALEZ CADAVAL**

México, D.F. 1993


**UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso**

CONTENIDO

**CAPITULO 1**

# IDEAS PRELIMINARES

## 1.1. Motivación.

La capacidad que tenemos los seres humanos para orientarnos en situaciones confusas es sorprendente. Parece que la evolución nos ha dotado de facultades intelectuales y perceptuales, que aun sin nuestra participación consciente, nos permiten extraer información relevante en contextos muy variados. Una nube o una mancha nos sugieren, de manera casi espontánea, un rostro o la figura de algún animal. Sin mucha dificultad reconocemos similitudes y diferencias en lugares donde no parece obvio que existan. Igualmente, podemos separar provechosamente bloques *significativos* dentro de una masa enorme y desorganizada de señales que afectan nuestra percepción. Al mismo tiempo somos capaces de procesar eficientemente información ambigua, errónea o incompleta. La identidad $E=mc^3$ es falsa, y sin embargo evoca de modo inmediato, el nombre de Albert Einstein (y de la identidad correcta).

Puede ser que lo que compartan estas habilidades radique en que todas ellas construyen o procesan bloques interrelacionados de información que llamamos *patrones*; y aun cuando lo anterior pudiese no ser totalmente cierto, el hecho de haber identificado (incluso arbitrariamente) como patrones al material básico sobre el cual operan nuestros procesos perceptuales y cognoscitivos ha resultado ser al menos, una simplificación muy fructífera. Ha servido como paradigma común, tanto a la Psicología Cognoscitiva, como a una buena parte de la Inteligencia Artificial.

La motivación que inspira a ambas es, sin embargo, muy distinta. A grandes rasgos, la Psicología Cognoscitiva se ocupa de indagar los modos y mecanismos mediante los cuales percibimos, formamos y procesamos patrones, mientras que una parte considerable de las investigaciones de la Inteligencia Artificial busca incorporar esta serie de habilidades a dispositivos no humanos. En otras palabras, se busca la manera de construir dispositivos artificiales (sistemas o máquinas) que se desempeñen con la misma flexibilidad y eficiencia en el manejo y formación de patrones, que sus contrapartes humanas. A esta rama de la Inteligencia Artificial se le conoce tradicionalmente como Reconocimiento de Patrones.

## 1.2. Contexto general.

Si los patrones fuesen en verdad la base de muchos de nuestros procesos mentales, podría pensarse que antes de iniciar cualquier investigación, deberíamos comenzar por definir lo que *es en realidad* un patrón. Sin embargo, en casi todas las ramas del Reconocimiento de Patrones, el punto de vista que se adopta es más funcional que filosófico. Básicamente consiste en dar una caracterización que sea al mismo tiempo intuitivamente plausible, así como fácilmente manipulable.

Bajo este enfoque, el universo de interés queda reducido a una colección de objetos y a un conjunto finito de atributos o variables. Un patrón resultará entonces, de identificar a un objeto con los valores particulares que toma en cada una de esas variables. Por ejemplo, en cierta discusión, puede ser suficiente que las variables a considerar sean: sexo, edad, peso, estatura y nacionalidad. Un patrón entonces, se obtendrá al evaluar a un determinado individuo en cada una de estas cinco variables.

Esta manera de codificar o representar a los objetos nos permite además, referirnos a los problemas principales que interesan al Reconocimiento de Patrones. Son esencialmente tres:

1. **El problema de la reducción de variables.**

2. **El problema de la clasificación.**

3. **El problema de la formación de cúmulos o clases.**

El primer problema consiste, como su nombre lo indica, en reducir el conjunto original de variables a uno más pequeño que permita seguir representando, sin ambigüedad, a todos los objetos de la colección. Como este trabajo se ubica dentro de esta problemática, en la siguiente sección lo trataremos con más detalle.

El problema de la clasificación parte de una colección finita de clases de patrones, y en donde dado un nuevo patrón $O^*$, la pregunta que busca contestar es la siguiente: ¿A cuál de todas las clases pertenece? Los problemas que se derivan de esta pregunta inicial se centran en crear los criterios y algoritmos más adecuados para clasificar.

En un problema de clasificación se tiene, por así decirlo, la mitad del problema resuelto: esto es, se tienen las clases. Sin embargo en muchas situaciones la tarea misma de especificar las clases es la que resulta problemática. Cuando esto sucede se

tiene un problema de formación de cúmulos. A pesar de las diferencias entre las distintas técnicas de agrupar objetos, todas ellas utilizan alguna forma de medir la similitud entre los patrones.

### 1.3. El problema de la reducción de variables.

En disciplinas tan diversas como la Medicina, la Biología, la Geología, etc., muchas veces resulta conveniente representar a los objetos de estudio como patrones. Esta forma de representar parte del supuesto de que cada objeto queda totalmente determinado por los valores que toma en cada una de las variables de dicha representación. En este contexto resulta que cada objeto *es* literalmente su descripción.

De esta suposición resulta además, que dos objetos cuyas descripciones sean iguales se consideren como el mismo objeto. El especialista, entonces, se ve en la necesidad de introducir un número suficiente de variables, que le permitan distinguir en la representación, a aquellos objetos que en su campo se consideran como diferentes. A esta última exigencia le llamaremos **Requisito de Representación (RR).**

De este modo, cuando en un principio se construye una representación, casi siempre se emplea un número elevado de variables que cumplan el RR. Más tarde, por razones de costo y tiempo, puede ser necesario reducir el conjunto original a un subconjunto propio pero que siga cumpliendo el RR. Es en esta situación donde resultan relevantes los conceptos de la Teoría de Testores. Conceptos que fueron introducidos por I.A. Cheguis y S.V. Yablonskii y que posteriormente fueron ampliados por Yu.I. Zhuravliov para atacar explícitamente el problema de la reducción de variables.

La Teoría de Testores se apoya en dos conceptos que resultan sumamente atractivos, tanto por su naturalidad como por su sencillez. Estos son, a saber, los conceptos de **testor** y **testor típico**. Para poder explicar adecuadamente los objetivos de este trabajo, los introducimos informalmente desde ahora y en el próximo capítulo los definimos con mayor precisión.

Supongamos que se tiene una muestra de objetos en la cual no existen dos iguales. Para simplificar el ejemplo, supongamos además que todas las variables sólo toman uno de dos posibles valores. ¿Qué pasa si ignoramos una variable? Es decir, ¿Qué sucede con las descripciones de los objetos vistos a la luz de todas las variables menos esa? Pueden pasar dos cosas: O aparecen al menos dos descripciones que se confunden, o bien las descripciones pueden seguirse distinguiendo. Si sucede esto último diremos que este subconjunto de variables constituye un

testor. Ahora bien, si continuamos este proceso, eventualmente llegaremos a un subconjunto de variables del cual no sea posible ignorar ningún rasgo sin obtener descripciones repetidas. A estos conjuntos *irreducibles* les llamaremos **testores típicos.**

A través de los testores típicos se han definido criterios que permiten reducir el espacio original de representación a uno más pequeño. La idea detrás de estos criterios consiste en averiguar el papel relativo que desempeña cada variable en la discriminación de los objetos. Serán eliminadas aquellas variables cuya presencia en *muchos* espacios de representación sea innecesaria. O dicho en otras palabras, se eliminarán aquellas variables que siempre (o casi siempre) puedan ser ignoradas sin provocar confusiones en las descripciones de los objetos. Esto, como es natural, sucede cuando ellas no aparecen en ningún (o en muy pocos) testores típicos.

Tradicionalmente, se ha utilizado el concepto de **peso diferenciante** de una variable para obtener una medida que permita decidir esta reducción. El peso diferenciante de una variable se define como el cociente del número de testores típicos en donde ella aparece, entre el número total de estos. Así las variables que es posible eliminar, serán aquellas cuyos pesos diferenciantes sean *bastante bajos.*

Calcular el peso diferenciante de las variables requiere resolver un problema previo: a saber, el de obtener todos los testores típicos. Debido a que el número de subconjuntos que potencialmente podrían ser testores típicos es del orden de $2^n$ cuando hay $n$ variables, se han diseñado varios algoritmos que reducen considerablemente los cálculos y permiten obtener a todos los testores típicos en un tiempo *razonable.*

Dependiendo de la manera en que obtienen los testores típicos los algoritmos se clasifican en:

**1. Algoritmos de escala interior.**

A grandes rasgos estos algoritmos construyen a todos los testores típicos. En realidad hacen más que eso, construyen una familia de conjuntos de variables dentro de la cual se encuentra la subfamilia de los testores típicos. Obtener de esta familia a la de los típicos no plantea ningún problema.

**2. Algoritmos de escala exterior.**

Estos algoritmos "revisan" la lista de todos los posibles conjuntos de variables, o lo que es lo mismo, la lista completa de sus vectores característicos (los vectores que resultan de aplicarle a cada conjunto su función característica). Como esta lista tiene longitud $2^n-1$, es claro que no resulta eficiente

generarla completa. De esta manera los algoritmos prueban en cada paso si el vector recién generado es o no un testor, y por la manera en que estos han sido ordenados, es posible realizar grandes saltos sobre muchos vectores que no pueden ser testores. Los algoritmos de esta categoría se distinguen unos de otros tanto por el orden en que son generados los vectores, como por los criterios específicos de salto.

## 1.4. Objetivos y antecedentes.

Este trabajo se centra exclusivamente en los algoritmos de escala interior para criterios de comparación *booleanos*. Una gran parte la discusión la realizamos, sin embargo, suponiendo que las variables son *booleanas*. Creemos que esta forma de proceder simplifica la presentación de las ideas y como veremos, no afecta nuestros resultados. En el capítulo 4 mostramos la manera de extender el caso de variables *booleanas* al de comparaciones *booleanas*. Nuestro propósito principal consiste en dar una caracterización completa del concepto de testor típico que facilite su obtención mediante una computadora. Esta caracterización y los dos algoritmos que discutimos se apoyan en un concepto nuevo y muy simple: el de **antitestor típico**. Sin embargo, la utilidad de los antitestores típicos es sólo teórica: mediante ellos simplificamos y unificamos. Simplificamos las definiciones y las demostraciones, y unificamos en uno solo, los teoremas de caracterización de cada algoritmo. Vale la pena comentar un poco ésto último.

Actualmente se tienen dos algoritmos de escala interior para construir testores típicos. Cada idea algorítmica se presenta, a grandes rasgos, bajo el siguiente esquema:

1. Definición de los conceptos *ad hoc* o propios del algoritmo.

2. Demostración de uno o varios resultados para caracterizar a los testores típicos a partir de estos conceptos *ad hoc*.

3. Demostración de uno o varios resultados que muestran que dentro de los conjuntos de variables construidos se encuentran los testores típicos.

Con esta forma de presentación, a nuestro juicio demasiado cargada de conceptos, no es posible explicar las diferencias (a nivel conceptual) que existen entre ambos algoritmos. Tampoco es posible saber si estas dos ideas algorítmicas agotan todas las posibles maneras de construir testores típicos o si existen formas radicalmente distintas (dentro de la clase de algoritmos de escala interior) de obtenerlos.

Nuestros resultados unifican y explican lo anterior a través del

concepto de antitestor típico. El esquema que adoptamos en este trabajo es el siguiente:

1. Definición de los conceptos de antitestor y de antitestor típico. Conceptos que son independientes de cualquier idea algorítmica.

2. Demostración de dos resultados de caracterización: el primero para testores y el segundo para testores típicos. El último de los cuales pone de manifiesto las dos (y sólo dos) ideas algorítmicas básicas que pueden seguirse en el caso de los algoritmos de escala interior.

3. Definición de los conceptos *ad hoc* de cada algoritmo, pero dados a partir del teorema de caracterización para testores típicos.

4. Demostración de que la familia de testores típicos se encuentra contenida en la familia de conjuntos que cada algoritmo construye.

Aun cuando en el capítulo 3 damos el pseudocódigo para cada algoritmo, nuestro interés no se encuentra en la parte computacional. No hemos hecho por tanto, ningún estudio de la eficiencia de cada algoritmo ni hemos realizado ningún tipo de estudio comparativo.

En el capítulo 4 presentamos algunas ideas, que aunque se salen de nuestro objetivo principal, consideramos apropiado discutir.

Finalmente conviene decir unas palabras respecto al estilo y tono del trabajo. Hemos preferido en todo momento, acercarnos a los conceptos paulatinamente; esto es, primero intuitivamente y después formalmente. Por lo cual no hemos evitado las redundancias, ni tampoco el empleo de un lenguaje a veces bastante informal. Respecto a las referencias, hemos decidido posponer cualquier indicación en el texto principal. En el apartado correspondiente mencionamos lo que hemos tomado de cada una de ellas.

**CAPITULO 2**

# TESTORES Y ANTITESTORES

En este capítulo presentamos las definiciones usuales de testor, testor típico y peso diferenciante. Asimismo introducimos los conceptos de antitestor y antitestor típico. Estos dos últimos son en realidad conceptos auxiliares y su principal utilidad reside en que a través de ellos podremos dar una caracterización muy simple y potente de los testores típicos. Posteriormente esta caracterización nos permitirá construir dos algoritmos para su obtención.

## 2.1. Conceptos básicos.

En todo lo que sigue supondremos que tenemos una muestra de objetos particionada en clases $K_1,\ldots,K_m$ y donde cada uno de los objetos se encuentra descrito por un conjunto R de variables o rasgos que sólo toman los valores 0 o 1. Para los subconjuntos de R nos reservamos las letras griegas $\alpha,\beta,\tau$ y $\omega$. Denotamos con $\alpha O$ a los valores que toma el objeto O cuando restringimos su descripición solamente a las variables que aparecen en $\alpha$.

Supondremos además, que nuestra partición se encuentra organizada en forma de matriz, los renglones representando a los objetos y las columnas a las variables. A esta matriz tradicionalmente se le conoce como matriz de aprendizaje (MA).

Ahora las definiciones:

**2.1.1.Definición.** Decimos que $\tau \subseteq R$ es un **testor** si para cualquiera $O \in K_i$ y $O' \in K_j$ con $i \neq j$ ocurre que $\tau O \neq \tau O'$.

**2.1.2.Definición.** Decimos que $\tau \subseteq R$ es un **testor típico** si $\tau$ es testor y para todo $\alpha \subset \tau$, $\alpha$ no es testor.

**2.1.3.Definición.** Sea $x \in R$, $T=\{\tau \subseteq R \mid \tau$ testor típico$\}$ y $T(x)=\{\tau \in T \mid x \in \tau\}$; el **peso diferenciante** de x es $\rho(x)=|T(x)|/|T|$.

**2.1.4.Ejemplo.**
Sea la siguiente matriz de aprendizaje en la cual hemos supuesto que las clases son unitarias:

| MA | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ |
|---|---|---|---|---|---|
| $O_1$ | 1 | 0 | 1 | 1 | 1 |
| $O_2$ | 0 | 1 | 0 | 1 | 1 |
| $O_3$ | 1 | 0 | 1 | 0 | 1 |
| $O_4$ | 1 | 1 | 0 | 0 | 0 |

Sus testores típicos son:
$\tau_1=\{x_2,x_4\}$, $\tau_2=\{x_3,x_4\}$, $\tau_3=\{x_1,x_4,x_5\}$.

Los pesos diferenciantes de las variables son:
$\rho(x_1)=\rho(x_2)=\rho(x_3)=\rho(x_5)=1/3$ y $\rho(x_4)=1$.

Aunque es la más utilizada en la práctica, la forma de medir el peso diferenciante que hemos presentado, no se encuentra exenta de problemas. En el capítulo 4 volveremos a ella para discutirlos.

Introducimos ahora los conceptos de antitestor y antitestor típico. El primero es simplemente la negación lógica del concepto de testor. El segundo tiene consecuencias más interesantes.

**2.1.5.Definición.** Decimos que $\omega \subseteq R$ es **antitestor** si existen $O \in K_i$ y $O' \in K_j$ con $i \neq j$ tales que $\omega O=\omega O'$.

De aquí que por definición la potencia de R (sin el vacío), queda dividida en dos clases ajenas: testores y antitestores.

**2.1.6.Definición.** Decimos que $\omega \subseteq R$ es **antitestor típico** si $\omega$ es antitestor y para toda $\alpha$ tal que $\omega \subset \alpha$, $\alpha$ no es antitestor.

Con esta última definición tenemos de manera inmediata la siguiente observación.

**2.1.7.Observación.** Dado $\alpha \subseteq R$, entonces $\alpha$ es antitestor $\Leftrightarrow$ $\alpha \subseteq \omega$ para algún $\omega$ antitestor típico.

Usando las ideas anteriores, podemos en este momento definir un algoritmo (ridículamente ineficiente) para obtener todos los testores típicos de una MA dada.

**Algoritmo I.**

**Paso 1.** Se obtienen de la MA todos los antitestores típicos. Esto es realizable en tiempo polinomial, pues el número de comparaciones (y por tanto de antitestores típicos) entre los objetos de todas las clases es polinomial.

**Paso 2.** Utilizando la observación 2.1.7. obtenemos a todos los antitestores.

**Paso 3.** Obtenemos a los testores eliminando de la potencia de R a todos los antitestores.

**Paso 4.** Nos quedamos con los testores típicos tomando a aquellos testores que no tienen contenido a ningún otro.

A pesar de que nadie se atrevería a usarlo en la práctica, este algoritmo ilustra de manera un poco tosca, la forma en que se vinculan antitestores típicos y testores típicos. Por otro lado, es claro que su tremenda ineficiencia proviene del hecho de que para obtener a los testores típicos necesita primero calcular a todos los antitestores y después a todos los testores. Sería deseable entonces, conocer otras propiedades de los testores típicos que nos permitieran construir algoritmos que alcanzaran su objetivo de manera más directa. Los resultados que presentamos a continuación son precisamente esas propiedades.

## 2.2. Teoremas de caracterización.

Como es natural caracterizamos primero al concepto de testor. A partir de ahora denotamos con $\Omega$ al conjunto de todos los antitestores típicos.

**2.2.1.Lema.** Sea $\alpha \subseteq R$, $\alpha$ es testor $\Leftrightarrow$ para toda $\omega \in \Omega$, $\alpha \cap (R-\omega) \neq \emptyset$.

**Demostración.**

$\Rightarrow$) Sea $\alpha$ testor y supongamos que existe $\omega_0 \in \Omega$ tal que $\alpha \cap (R-\omega_0)=\emptyset$. Entonces $\alpha \subseteq \omega_0$ y por la observación 2.1.7. $\alpha$ es antitestor, lo que es una contradicción.
$\Leftarrow$) La hipótesis implica que $\alpha$ no está contenido en ningún antitestor típico, y de nuevo por la observación 2.1.7., $\alpha$ no puede ser antitestor, luego $\alpha$ es testor. ∎

**2.2.2.Lema.** Sea $\tau$ un testor típico y $x \in \tau$, entonces existe $\omega_0 \in \Omega$ tal que $x \in R-\omega_0$.

**Demostración.**

Como $\tau$ es testor típico y $x \in \tau$, entonces $\tau-\{x\}$ es antitestor. Por la observación 2.1.7. existe $\omega_0 \in \Omega$ tal que $\tau-\{x\} \subseteq \omega_0$, por lo cual $x \in R-\omega_0$. ∎

Caracterizamos ahora a los testores típicos. Para eso definimos el concepto de conjunto compatible. Concepto que utilizaremos en el capítulo 3 para definir uno de nuestros dos algoritmos.

**2.2.3.Definición.** Decimos que $\alpha \subseteq R$ es un **conjunto compatible** si para cada $x \in \alpha$ existe $\omega \in \Omega$ tal que $\alpha \cap (R-\omega) = \{x\}$.

**2.2.4.Teorema.** Sea $\tau \subseteq R$ entonces $\tau$ es testor típico $\Leftrightarrow$
a) Para toda $\omega \in \Omega$ , $\tau \cap (R-\omega) \neq \emptyset$ (o que $\tau$ es testor).
b) $\tau$ es un conjunto compatible.

**Demostración.**

$\Rightarrow$) Sea $\tau$ testor típico, entonces se sigue a). Vemos que $\tau$ es compatible. Por el lema anterior, dada $x \in \tau$, sabemos que el conjunto $\mathbb{D}(x)=\{R-\omega | x \in \tau \cap (R-\omega),\ \omega \in \Omega\}$ es distinto del vacío. Supongamos que para todo $\beta \in \mathbb{D}(x)$ ocurre que existe $x(\beta) \in \tau \cap \beta$ con $x(\beta) \neq x$, entonces $(\tau-\{x\}) \cap \beta \neq \emptyset$ para las $\beta \in \mathbb{D}(x)$. Para los demás elementos de la forma $R-\omega$, $\omega \in \Omega$, que no se encuentran en $\mathbb{D}(x)$, $(\tau-\{x\}) \cap (R-\omega)$ es también distinto del vacío pues $\tau$ es testor. Y entonces por el lema 2.2.1., $\tau-\{x\}$ es testor, lo cual es una contradicción. Luego $\tau$ es compatible.
$\Leftarrow$) La parte a) es el lema 2.2.1. que implica que $\tau$ es testor. Sea $\alpha \subset \tau$, entonces existe $x_0 \in \tau-\alpha$. Como $\tau$ es conjunto compatible entonces existe $\omega_0 \in \Omega$ tal que $\tau \cap (R-\omega_0)=\{x_0\}$, luego $\alpha \subseteq \omega_0$ y por la tan socorrida observación 2.1.7., $\alpha$ es antitestor; es decir, $\alpha$ no es testor. De aquí que $\tau$ es testor típico. ∎

Lo que nos dice el teorema anterior es que todos los testores típicos se encuentran *metidos* de un modo muy especial, dentro de los complementos de los antitestores típicos. Por lo que la función de nuestros algoritmos será, por así decirlo, la de *entresacarlos*.

Sin embargo, antes de pasar a los algoritmos vemos cómo se ponen de manifiesto las dos propiedades del teorema en el ejemplo de la sección anterior. Para esto obtenemos los antitestores típicos y sus correspondientes complementos. La manera de obtenerlos es comparando cada uno de los objetos de la clase $K_i$ (para i=1,...,m) con todos los objetos de las demás clases. La comparación se hace rasgo a rasgo y los resultados se recogen en una matriz (MC). Si los valores coinciden se pone un 1 y si no, un 0. Para nuestro ejemplo queda:

| MC | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ |
|---|---|---|---|---|---|
| $O_1O_2$ | 0 | 0 | 0 | 1 | 1 |
| $O_1O_3$ | 1 | 1 | 1 | 0 | 1 |
| $O_1O_4$ | 1 | 0 | 0 | 0 | 0 |
| $O_2O_3$ | 0 | 0 | 0 | 0 | 1 |
| $O_2O_4$ | 0 | 1 | 1 | 0 | 0 |
| $O_3O_4$ | 1 | 0 | 0 | 1 | 0 |

Cada renglón es el vector característico de un antitestor, por lo que los antitestores típicos con sus complementos son:

| Antitestores típicos | Complementos |
|---|---|
| $\omega_1=\{x_4,x_5\}$ | $\{x_1,x_2,x_3\}$ |
| $\omega_2=\{x_1,x_2,x_3,x_5\}$ | $\{x_4\}$ |
| $\omega_3=\{x_1,x_4\}$ | $\{x_2,x_3,x_5\}$ |

Con los complementos y el teorema se ve porqué $\{x_2,x_4\}$, $\{x_3,x_4\}$ y $\{x_1,x_4,x_5\}$ son los únicos testores típicos. El conjunto $\{x_1,x_5\}$ es compatible, pero no es testor debido a que su intersección con el complemento de $\omega_2$ es vacía. Por otro lado, el conjunto $\{x_1,x_3,x_4\}$ es testor pero no es compatible, y por tanto no puede ser típico.

**CAPITULO 3**

# DOS ALGORITMOS

Del teorema de caracterización del capítulo anterior se desprenden con mucha naturalidad dos ideas algorítmicas diferentes. La esencia de ambas es explotar una de las dos propiedades de los testores típicos para construir conjuntos especiales de variables. Así, el primer algoritmo que discutimos (TE) construye cierto tipo de testores que hemos llamado testores escogidos, mientras que el segundo algoritmo (CC) construye un tipo especial de conjuntos compatibles conocidos como compatibles maximales. La forma en que presentamos a cada algoritmo es la siguiente:

1. Definimos el tipo de conjuntos que el algoritmo construye; testores escogidos para el primero y compatibles maximales para el segundo.

2. Demostramos en cada caso, lo que hemos denominado Teorema de Suficiencia. Esto es, la demostración de que dentro de la familia de conjuntos construidos se encuentra contenida la subfamilia de todos los testores típicos. Dicho en otras palabras, demostramos que el algoritmo *hace lo que debe hacer.*

Al final, en una sección aparte, damos los pseudocódigos para cada uno de ellos.

## 3.1. Algoritmo TE (testores escogidos).

Explicamos la idea de este algoritmo mediante un ejemplo.

**3.1.1. Ejemplo.** Supongamos que los siguientes conjuntos corresponden a los complementos de los antitestores típicos de cierta MA para la cual $|R| = 6$.

$$\beta_1=\{x_1, x_3\}$$
$$\beta_2=\{x_1, x_2, x_5\}$$
$$\beta_3=\{x_2, x_3, x_6\}$$
$$\beta_4=\{x_2, x_4, x_5, x_6\}$$

Es claro por el lema 2.2.1., que si escogemos de cada complemento $\beta_i$, una variable y al final las agrupamos en un conjunto, éste será un testor. Si ahora efectuamos todas estas maneras de escoger variables, el teorema 2.2.4. nos garantiza que dentro de la familia de testores obtenida, se encuentran todos los testores

típicos. Sin embargo con esta forma un tanto ingenua, se construyen *demasiados* testores. Buscamos entonces, mejorar esta idea inicial, definiendo una forma más eficiente de construirlos.

Sea $\Omega^*=\{\beta \mid \beta=R-\omega,\ \omega \in \Omega\}$ el conjunto de los complementos de los antitestores típicos. Y para que lo siguiente tenga sentido, lo suponemos ordenado de alguna manera. Las funciones que definimos a continuación serán el modo en que nuestro algoritmo construye testores escogiendo una variable de cada complemento.

**3.1.2.Definición.** Decimos que $\alpha$ es un **testor escogido** si existe una función $\phi$: $\Omega^* \longrightarrow R$ tal que para todo $i=1,\ldots,|\Omega|$, $\phi$ está definida como:

$\phi(\beta_i) = \phi(\beta_j)$ si para alguna $j < i$ $\phi(\beta_j) \in \beta_i$.

$\phi(\beta_i) = x$ con $x \in \alpha \cap \beta_i$, en otro caso.

Es claro que $Im\phi = \alpha$ y además $\alpha$ así construido es testor. De aquí que el nombre de "testor" en la definición sea apropiado.

En el ejemplo 3.1.1. la función que construye al testor típico $\{x_1,x_2\}$ es: $\phi(\beta_1)=x_1$, $\phi(\beta_2)=x_1$, $\phi(\beta_3)=x_2$, $\phi(\beta_4)=x_2$. Es fácil ver que el testor $\{x_1,x_2,x_4\}$ no es un testor escogido, pues no existe ninguna función $\phi$ que permita obtenerlo sin alterar el orden de los $\beta_i$'s.

El algoritmo TE *examina*, por así decirlo, todas estas formas (o funciones) de construir testores. En la práctica, sin embargo, sólo se observa al testor obtenido, pues la función que lo produce nunca se define explícitamente. Por otro lado, es importante señalar, que el conjunto de testores escogidos que se construyen depende del orden de los complementos en $\Omega^*$. Afortunadamente esto último no tiene ningún efecto sobre los testores típicos.

**3.1.3.Teorema de suficiencia.**

Sea $E=\{\alpha \subseteq R \mid \alpha$ testor escogido$\}$ y $T=\{\tau \subseteq R \mid \tau$ testor típico$\}$ entonces $T \subseteq E$.

**Demostración.**

Sea $\tau \in T$ y sea $B_\tau=\{\beta \in \Omega^* \mid \beta \cap \tau = \{x\},\ x \in \tau\}$; aprovechamos las propiedades del teorema 2.2.4. para definir la siguiente función:
Sea $\psi$: $\Omega^* \longrightarrow R$ tal que
$\psi(\beta) = x$ tal que $x \in \beta \cap \tau$, si $\beta \in B_\tau$
$\psi(\beta) = x$ con $x \in \beta \cap \tau$, si $\beta \notin B_\tau$

La función $\psi$ construye al testor típico $\tau$ pues $Im\psi = \tau$ .Con ella podemos definir una función $\phi$ que haga a $\tau$ un testor escogido.
Para $\beta_1$ definimos $\phi(\beta_1)=\psi(\beta_1)$.

Sea ahora $\beta_i \in \Omega^*$ $i=2,\ldots,|\Omega^*|$ y supongamos que para toda $j < i$ $\phi(\beta_j)$ está definida.
Para el caso en que exista alguna $j < i$ tal que $\phi(\beta_j) \in \beta_i$ definimos a $\phi(\beta_i)$ como $\phi(\beta_i)=\phi(\beta_j)$.
Para el caso contrario, i.e. cuando no existe $j < i$ para la cual $\phi(\beta_j) \in \beta_i$, entonces $\phi(\beta_i)=\psi(\beta_i)$. De esta manera $\phi = \psi$ y por lo tanto $Im\phi = \tau$, lo que hace a $\tau$ un testor escogido.■

Una vez que se han obtenido los testores escogidos, para quedarse con los testores típicos, el algoritmo TE busca a todos aquellos elementos de E que no tengan a ningún otro elemento de E como subconjunto propio.

### 3.2. Algoritmo CC (conjuntos compatibles).

Este algoritmo explota la propiedad de compatibilidad que tienen los testores típicos. La idea es construir cierto tipo de conjuntos compatibles y al final verificar cuales de ellos son testores. Aquellos que lo sean, por el teorema 2.2.4., serán testores típicos.

Dado cualquier conjunto compatible $\alpha$, siempre podemos ordenar a sus variables del siguiente modo:
Sea $\psi:\alpha \longrightarrow \Omega^*$ definida como:
$\psi(x)=\beta_i$ donde $\beta_i \in \Omega^*$ y es tal que $\beta_i \cap \alpha = \{x\}$ (como pueden existir varios complementos que cumplan lo anterior, tomamos uno sólo de ellos).

Los elementos de $\alpha$ quedan entonces ordenados con el subíndice del complemento que les asigne la función $\psi$. Es claro que $\alpha$ puede quedar ordenado de muchas maneras, pues para algún $x \in \alpha$ pudiesen existir varios $\beta_i$'s en $\Omega^*$ tales que $\beta_i \cap \alpha = \{x\}$. A dos compatibles que coincidan en su dominio pero que difieran en el orden, los consideraremos como diferentes. A continuación definimos los conjuntos compatibles que queremos que nuestro algoritmo construya.

**3.2.1.Definición.** Decimos que $\alpha$ compatible ordenado (por alguna $\psi$) es **maximal** ssi para cualquier $\beta_j \in \Omega^*$ con $j > i_{max}$ (el subíndice del último elemento de $\alpha$) le ocurre que $\beta_j \cap \alpha \neq \emptyset$
El algoritmo irá construyendo compatibles (avanzando sobre los $\beta_i$'s) hasta que sean maximales.

### 3.2.2. Teorema de suficiencia.

Sea $M=\{\alpha \subseteq R \mid \alpha$ es maximal$\}$ y $T=\{\tau \subseteq R \mid \tau$ es testor típico$\}$ entonces $T \subseteq M$.

**Demostración.**

Sea $\tau \in T$ y supongamos que $\tau \notin M$. Como $\tau$ es testor típico, por el teorema 2.2.4. es compatible (y sin pérdida de generalidad, ordenado). $\tau \notin M$ implica que existe $\beta_j > i_{max}$ tal que $\beta_j \cap \tau = \emptyset$. Pero entonces por el lema 2.2.1., $\tau$ no es testor. De esta contradicción se sigue que $\tau \in M$.■

## 3.3. Pseudocódigos.

### 3.3.1. Cálculo de los complementos.

Un paso común a ambos algoritmos es la obtención de los complementos de los antitestores típicos de la MA original. De esta manera, en lugar de calcular primero los antitestores, enseguida los antitestores típicos y finalmente sus complementos, podemos obtener directamente los complementos de los antitestores y después los complementos de los antitestores típicos. Esto es, ahorrarnos un paso.

El cálculo de estos complementos es como sigue:

**Paso 1.**

Se comparan (ver lo referente a los criterios de comparación *booleanos*.en el capítulo siguiente) todos los objetos de la clase $K_i$ con todos los objetos de las demás clases, y esto se hace para todas las clases. Se forma una matriz de comparación (MC) cuyas entradas son el resultado de estas comparaciones. Se pone un 0 para aquellas variables (columnas) donde los objetos coincidan y un 1 en otro caso. Al final se tiene una matriz cuyos renglones son los vectores característicos de los complementos de los antitestores.

**Paso 2.**

Se construye la matriz de los vectores característicos de los complementos de los antitestores típicos eliminando de MC a aquellos renglones cuyos correspondientes conjuntos contienen a algún otro conjunto (renglón). Si hubiese renglones repetidos nos quedamos con uno sólo de ellos. A la matriz resultante se le conoce como matriz básica (MB).

**Paso 3.**

En este momento ya podríamos *ejecutar* alguno de nuestros algoritmos. Este paso está descrito de manera muy informal y lo incluímos solamente para que el pseudocódigo que presentamos tenga sentido.

Para cada renglón de MB formamos el conjunto que representa. Cada conjunto los acomodamos en un renglón de una matriz, pero recorriéndolo hacia la izquierda. Finalmente rellenamos de ceros todas las posiciones a la derecha del último elemento del conjunto. En esta matriz habrá tantos renglones como elementos tenga $\Omega$ ($\Omega^*$) y columnas como variables tenga R. Cada renglón lo consideraremos como una $\beta_i \in \Omega^*$.

### 3.3.2. Algoritmo TE.

Para facilitar la lectura, explicamos lo que los siguientes objetos representan:

$\mathbb{F}$ y $\mathbb{C}$ son los conjuntos en donde se van guardando los índices de las filas y columnas respectivamente. $\mathbb{T}$ es el testor que se va construyendo, mientras que f y c son respectivamente, los índices de filas y columnas. Las $a_{fc}$ son las entradas de la matriz anterior y por supuesto representan a las variables.

**Paso 0.**

$\mathbb{C}=\{1\}$; $\mathbb{F}=\{1\}$; f=1; c=1; $\mathbb{T}=\{a_{fc}\}$ y Paso 1.

**Paso 1.**

f:=f+1;

Si f > $|\Omega|$ entonces Paso 3

Si f $\leq$ $|\Omega|$ entonces

  Si $\mathbb{T} \cap \beta_f \neq \emptyset$ repetir Paso 1

  Si $\mathbb{T} \cap \beta_f = \emptyset$ entonces

    $\mathbb{F}:=\mathbb{F} \cup \{f\}$; $\mathbb{C}=\mathbb{C} \cup \{1\}$ y Paso 2.

**Paso 2.**

c:=1;

f:=último elemento de $\mathbb{F}$;

$\mathbb{T}:=\mathbb{T} \cup \{a_{fc}\}$ y Paso 1.

**Paso 3.**

Guardar $\mathbb{T}$ en un archivo;

c:=c+1;

f:=último elemento de $\mathbb{F}$ y Paso 4.

**Paso 4.**

Si $a_{fc}$= 0 entonces

$\mathbb{F}:=\mathbb{F}-\{$último elemento de $\mathbb{F}\}$;

Si $\mathbb{F} \neq \emptyset$ entonces

$\mathbb{C}:=\mathbb{C}-\{$último elemento de $\mathbb{C}\}$;

$\mathbb{T}:=\mathbb{T}-\{$último elemento de $\mathbb{T}\}$;

f:=último elemento de $\mathbb{F}$;

c:=último elemento de $\mathbb{C}$;

c:=c+1 y repetir paso 4

Si $\mathbb{F} = \emptyset$ entonces Stop

Si $a_{fc} \neq 0$ entonces

$\mathbb{T}:=\mathbb{T}-\{$último elemento de $\mathbb{T}\}$;

$\mathbb{T}:=\mathbb{T} \cup \{a_{fc}\}$;

$\mathbb{C}:=\mathbb{C}-\{$último elemento de $\mathbb{C}\}$

$\mathbb{C}:=\mathbb{C} \cup \{c\}$ y Paso 1.

### 3.3.3. Algoritmo CC.

Los significados de $\mathbb{F}$, $\mathbb{C}$, f y c son como los del algoritmo anterior y $\mathbb{M}$ representa el maximal que se va construyendo.

**Inicio.**

c:=1; f:=1 y Paso 1.

**Paso 0.**

$\mathbb{C}=\{c\}$; $\mathbb{F}=\{f\}$; $\mathbb{M}=\{a_{fc}\}$ y Paso 1.

**Paso 1.**

f:=f+1;

Si $f > |\Omega|$ entonces Paso 3

Si $f \leq |\Omega|$ entonces

Si $\mathbb{M} \cap \beta_f \neq \emptyset$ repetir Paso 1

Si $\mathbb{M} \cap \beta_f = \emptyset$ entonces c:=1 y Paso 2.

**Paso 2.**

Si $a_{fc}$= 0 entonces Paso 4

Si $a_{fc} \neq 0$ entonces

Si $a_{fc} \in \beta_f$ para alguna $f \in \mathbb{F}$ entonces
$c:=c+1$ y repetir Paso 2
Si $a_{fc} \notin \beta_f$ para toda $f \in \mathbb{F}$ entonces
$\mathbb{M}:=\mathbb{M} \cup \{a_{fc}\}$; $\mathbb{C}:=\mathbb{C} \cup \{c\}$; $\mathbb{F}:=\mathbb{F} \cup \{f\}$ y Paso 1.

**Paso 3.**

Guardar $\mathbb{M}$ en un archivo;
$c:=c+1$;
f:=último elemento de $\mathbb{F}$;
$\mathbb{M}:=\mathbb{M}-\{$último elemento de $\mathbb{M}\}$;
$\mathbb{F}:=\mathbb{F}-\{$último elemento de $\mathbb{F}\}$;
$\mathbb{C}:=\mathbb{C}-\{$último elemento de $\mathbb{C}\}$ y Paso 2.

**Paso 4.**

Si $\mathbb{F} \neq \emptyset$ entonces
f:=último elemento de $\mathbb{F}$;
c:=último elemento de $\mathbb{C}$;
$c:=c+1$;
$\mathbb{F}:=\mathbb{F}-\{$último elemento de $\mathbb{F}\}$;
$\mathbb{C}:=\mathbb{C}-\{$último elemento de $\mathbb{C}\}$;
$\mathbb{M}:=\mathbb{M}-\{$último elemento de $\mathbb{M}\}$ y Paso 2
Si $\mathbb{F} = \emptyset$ entonces
Si $f+1 < |\Omega|$ entonces $c:=1$; $f:=f+1$ y Paso 0
Si $f+1 \geq |\Omega|$ entonces Stop.

**CAPITULO 4**

# IDEAS FINALES

En este capítulo ahondamos en algunos de los puntos que quedaron sugeridos en los capítulos precedentes.

## 4.1. Criterios de comparación booleanos.

En el capítulo 1 señalamos que una forma frecuente de representar a los objetos de estudio consiste en identificarlos con el conjunto de valores que toman al evaluarse en cada una de las variables fijadas por el especialista. En nuestra discusión no especificamos el tipo de estas variables, y esto podría haber dado la impresión de que todo lo que después dijimos, se aplicaba únicamente al caso en que las variables fuesen de naturaleza booleana. Como pronto veremos, esto no es así. Tanto los conceptos de testor, testor típico, antitestor y antitestor típico, asi como los teoremas de caracterización y los algoritmos, no se ven modificados sustancialmente cuando introducimos en la discusión variables de naturaleza distinta a la booleana.

Mostramos ahora, dos caminos que nos permiten seguir utilizando el esquema testor-antitestor dentro de este contexto más general. El primero, y que nosotros descalificamos inmediatamente, consiste en tratar a variables como, por ejemplo, la edad, el peso, la estatura, etc., que claramente no son booleanas, como si lo fueran. Cualquiera puede tomar al conjunto (ordenado) de posibles edades y trazar una linea que lo divida en dos clases ajenas. Cón esta manera de proceder es obvio que se simplifica el modelo, sin embargo, el precio que se tiene que pagar es el de la pérdida de "realismo". Además no parece fácil contestar a las siguientes preguntas. ¿Qué pasa con valores muy cercanos a la frontera y que se encuentran en clases distintas? ¿O qué pasa con valores muy alejados entre sí y que se encuentran dentro de una misma categoría?

El segundo camino que presentamos nos parece más razonable. Consiste, no en *booleanizar* las variables, sino en *booleanizar* la manera en que comparamos sus valores. Ilustramos esto mediante un ejemplo. Supongamos que una de nuestras variables es la edad medida en años. En sentido estricto, tener 34 años no es lo mismo que tener 36, sin embargo, en cierta discusión, una diferencia de dos años puede ignorarse. Esto significa que cuando la diferencia de edades de dos individuos no sea mayor de dos años, para propósitos prácticos, estas podrían considerarse como iguales. La

palabra que se utiliza en este caso no es "iguales", sino "coincidentes" o "semejantes". Lo anterior nos lleva a considerar la noción de semejanza entre valores y junto con ella la de semejanza entre objetos. Nótese que esta nociones se encuentran más cercanas a la manera en que trabaja el propio especialista; ya que es él quién decide, en última instancia, cuando dos valores deben o no considerarse como semejantes. Formalizamos ahora ambos conceptos.

**4.1.1.Definición.** Dos objetos O y O′ se dice que son semejantes en la variable $x \in R$, si $|\{x\}O - \{x\}O'| \leq \varepsilon_x$.

En notación, lo anterior se escribe como $s(\{x\}O,\{x\}O') = 1$ si son semejantes y $s(\{x\}O,\{x\}O') = 0$, en el caso en que no lo sean.
Al valor $\varepsilon_x$ que depende de x, se le llama umbral de semejanza.
Para el caso de variables booleanas este valor es naturalmente cero.

**4.1.2.Definición.** Dos objetos O y O′ se dice que son semejantes en $\alpha \subseteq R$ si $s(\{x\}O,\{x\}O') = 1$ para toda $x \in \alpha$.

La notación es ahora $s(\alpha O,\alpha O') = 1$, si son semejantes en todas las variables de $\alpha$ y $s(\alpha O,\alpha O') = 0$ en otro caso.

La definición de semejanza que hemos dado para los objetos no es más que un caso particular de lo que se conoce como funciones de semejanza. Existen muchas maneras útiles de definir la noción de semejanza y la forma en que nosotros la hemos presentado es lo que se conoce como *criterios de comparación booleana*. Bajo este enfoque particular, los conceptos, teoremas y algoritmos que hemos discutido, se pueden aplicar casi sin modificacion.

En las definiciones de testor y de antitestor deberán sustituirse las expresiones $\tau O \neq \tau O'$ y $\omega O = \omega O'$ por las nuevas expresiones $s(\tau O,\tau O') = 0$ y $s(\omega O,\omega O') = 1$ respectivamente. Esta ligera modificación no altera los conceptos de testor y de antitestor típicos. De aquí que para este caso un poco más general, los teoremas de caracterización no sufren ninguna modificación. Respecto a los algoritmos, lo único que cambia, es la manera de comparar a los objetos para obtener los complementos de los antitestores típicos. La comparación, como hemos apuntado arriba, se realiza ahora considerando los umbrales definidos para cada una de las variables.

## 4.2. Testores, antitestores y anticadenas.

Para una MA, la familia de antitestores típicos y la de los testores típicos, resultan ser, por definición anticadenas con la contención como el orden parcial. En cierto sentido podemos pensar que ambas anticadenas se encuentran *emparentadas*, pues la anticadena de antitestores típicos nos permite generar a la de los testores típicos. Si usamos el teorema 2.2.4. como punto de partida, podemos definir una forma más general de *generar* anticadenas a partir de anticadenas y que tal vez valga la pena investigar. Apuntamos aquí algunas ideas.

Sean $\mathcal{A},\mathcal{B} \subseteq \mathcal{P}(R)$ y $\mathcal{A}$ anticadena (con la contención),

**4.2.1.Definición.** Decimos que $\mathcal{A}$ **genera** a $\mathcal{B}$ si para todo $\beta \in \mathcal{B}$ ocurre que:
a) Para todo $\alpha \in \mathcal{A}$, $\beta \cap \alpha \neq \emptyset$.
b) Para todo $x \in \beta$ existe $\alpha \in \mathcal{A}$ tal que $\beta \cap \alpha = \{x\}$.

**4.2.2.Definición.** Decimos que $\mathcal{A}$ **genera maximalmente** a $\mathcal{B}$ si
a) $\mathcal{A}$ genera a $\mathcal{B}$.
b) Si $\mathcal{A}$ genera a $\mathcal{D}$ entonces $\mathcal{D} \subseteq \mathcal{B}$.

Los símbolos $g(\mathcal{A})=\mathcal{B}$ y $gm(\mathcal{A})=\mathcal{B}$ significan respectivamente, que $\mathcal{A}$ **genera** a $\mathcal{B}$ y que $\mathcal{A}$ **genera maximalmente** a $\mathcal{B}$.

**4.2.3.Lema.** Si $g(\mathcal{A})=\mathcal{B}$ entonces $\mathcal{B}$ es anticadena.

**Demostración.**

Sean $\beta_1$, $\beta_2 \in \mathcal{B}$, $\beta_1 \neq \beta_2$ y supongamos que $\beta_1 \subset \beta_2$. Esto significa que existe $x \in \beta_2-\beta_1$; por b) para esta x existe $\alpha \in \mathcal{A}$ tal que $\beta_2 \cap \alpha = \{x\}$. Como $\beta_1 \subset \beta_2$ entonces $\beta_1 \cap \alpha = \emptyset$ lo que contradice a). De este modo se sigue que $\mathcal{B}$ es anticadena.

La definición 4.2.2., junto con el lema anterior, permiten ver que $gm$ es función ($gm:\mathfrak{A}\longrightarrow\mathfrak{A}$, donde $\mathfrak{A}$ es la familia de las anticadenas de $\mathcal{P}(R)$). Resulta además que $gm$ es biyección.

Si ahora denotamos con $gm^{(n)}(\mathcal{A})$ a la enésima aplicación de $gm$ empezando en $\mathcal{A}$, podemos hacernos las siguientes preguntas (más bien borrosas):

1. Como $gm$ es una permutación, entonces dada una $\mathcal{A}$, anticadena arbitraria ¿podrá determinarse la N tal que $gm^{(N)}(\mathcal{A})=\mathcal{A}$?
2. ¿Podrá determinarse el número de ciclos en los cuales se descompone $gm$?
3. Y una metapregunta: ¿Qué relación podrían tener las respuestas encontradas, con los problemas combinatorios de la Teoría de

Testores o con las anticadenas en general?

No son todas las preguntas que podrían formularse y hasta ahora ignoramos todas las respuestas. Ignoramos incluso la dificultad que puedan plantear. Algunas de ellas no parecen tener una respuesta sencilla.

## 4.3. Peso diferenciante.

Discutimos ahora algunos de los problemas que plantea la definición 2.1.3. de peso diferenciante. Nuestro propósito no es dar una solución definitiva, sino proponer una vía que quizá permita definirlo de un modo más satisfactorio.

Ilustramos uno de sus inconvenientes mediante el siguiente ejemplo. Sean $\tau_1=\{x_2,x_4\}$, $\tau_2=\{x_3,x_4\}$ y $\tau_3=\{x_1,x_4,x_5\}$ los testores típicos de alguna MA. De acuerdo con la definición 2.1.3., los pesos diferenciantes de las variables son: $\rho(x_4)=1$ y $\rho(x_1)=\rho(x_2)=\rho(x_3)=\rho(x_5)=1/3$. Según estos cálculos $x_1$ y $x_2$ tienen el mismo peso diferenciante. Sin embargo, uno se inclinaría a pensar que $x_2$ debería tener un peso diferenciante mayor que el de $x_1$, ya que aun cuando ambas pertenecen a un sólo testor típico, el de $x_2$ es más pequeño. O dicho de otro modo, $x_2$ requiere de menos variables que le ayuden a discriminar a todos los objetos.

Podría entonces pensarse, que para resolver este problema, bastaría definir el peso diferenciante de manera que tomara en cuenta no sólo la cardinalidad del conjunto de los testores típicos en los que se encuentra cada $x \in R$, sino también su *longitud*. Por ejemplo:

Sea $x \in R$, T y T(x) como en la definición 2.1.3.

**4.3.1.Definición.** Sean $s(x)=\sum(|R|-|\tau|)$ para los $\tau \in T(x)$ y $M=\max(\{s(x)|x \in R\} \cup \{1\})$; el **peso diferenciante*** de x es $\rho^*=s(x)/M$.

De este modo $\rho^*(x_1)=\rho^*(x_5)=2/8$, $\rho^*(x_2)=\rho^*(x_3)=3/8$ y $\rho^*(x_4)=1$. Esta definición introduce una finura adicional que permite acercarse más a la idea intuitiva de lo que se entiende por peso diferenciante. Notamos además, que en el caso en que R mismo, fuese el único testor típico, la definición tradicional arroja el valor de 1 para todas las variables, mientras que esta definición da el valor de 0. Valor más acorde con lo que esperaríamos; a saber, que todas las variables tienen nula capacidad para diferenciar, y que sólo pueden hacerlo cuando estan acompañadas del resto.

Sin embargo no todo son buenas noticias, pues tanto esta definición como la anterior adolecen de un mismo defecto y que es bastante grave. Supongamos por ejemplo, que tenemos la siguiente colección de testores típicos: $\tau_1=\{x_3,x_4\}$, $\tau_2=\{x_2,x_5\}$, $\tau_3=\{x_1,x_5\}$, $\tau_4=\{x_6\}$. Claramente $x_6$ es una variable muy poderosa (en realidad la más poderosa), pues ella basta para discriminar a todas las clases u objetos. Su peso diferenciante, en cambio, calculado con cualquiera de las definiciones no refleja este hecho. Los pesos son: $\rho(x_1)=\rho(x_2)=\rho(x_3)=\rho(x_4)=\rho(x_6)=1/4$ y $\rho(x_5)=2/4$; $\rho^*(x_1)=\rho^*(x_2)=\rho^*(x_3)=\rho^*(x_4)=1/8$, $\rho(x_6)=5/6$ y $\rho^*(x_5)=1$.

El problema en general se centra en cómo *combinar* en una definición, el número de testores típicos en los que se encuentra una variable, con sus correspondientes longitudes, de manera que no se obtengan resultados contra-intuitivos.

Nos parece, por otro lado, que buscar *la fórmula* de peso diferenciante no es el camino adecuado. Podría resultar más provechoso definir primero, una serie de propiedades que reflejen lo que debe entenderse por una función de peso diferenciante, para poder después construir funciones adecuadas al tipo de problema que se intenta resolver (reducción de variables, clasificación, etc.).

## 4.4. Conclusiones.

Creemos que las secciones precedentes muestran que la definición de antitestor típico, a pesar de su simplicidad (o quizá precisamente por ella), resultó conveniente para los propósitos que nos planteamos. Creemos además, aunque pudieramos pecar de dogmáticos, que en lo que se refiere a los algoritmos de escala interior y para el caso exclusivo de comparaciones *booleanas*, no queda mucho trabajo teórico por hacer. Nos parece poco probable que exista alguna otra propiedad que caracterice a los testores típicos y que permita crear algún algoritmo realmente nuevo. Los algoritmos que presentamos seguramente pueden mejorarse dando definiciones más restrictivas, ya sea de los testores escogidos o bien de los compatibles maximales. Los algoritmos mejorados, sin embargo, explotarían alguna de las dos propiedades del teorema 2.2.4. o alguna combinación ingeniosa de ambas; pero siempre se les podría referir al teorema en cuestión.

## REFERENCIAS

1. Aguila Feros Luis, Ruiz Shulcloper José. Algoritmo MB para la elaboración de la información k-valente en problemas de reconocimiento. *Revista de Ciencias Matemáticas*. vol.V (1984) 89-101.

De este artículo tomamos el algoritmo MB modificando las definiciones de conjunto compatible y de compatible ordenado maximal para adecuarlas a nuestro enfoque. Este artículo por otro lado, generaliza el algoritmo CC original de E. V. Diukova al caso k-valente.

2. Bravo Martínez Adrián: Algoritmo CT para el cálculo de los tests típicos de una matriz k-valente. *Revista de Ciencias Matemáticas*. vol.IV (1983).

En este artículo aparece el algoritmo TE para el caso k-valente. Nosotros obtuvimos este algoritmo de manera independiente y por eso nos permitimos la libertad de rebautizarlo.

3. Ruiz Shulcloper José, Teoría Matemática del Reconocimiento de Patrones, (material para un curso de posgrado).

Estas notas fueron utilizadas por el Dr. Shulcloper para impartir, durante el segundo semestre de 1992, un curso de maestría en la Facultad de Ciencias. De ellas tomamos las definiciones básicas de la Teoría de Testores y nos familiarizamos con los algoritmos de escala exterior y con las funciones de semejanza. Este curso, junto con las notas, inspiraron gran parte de las ideas que aparecen en este trabajo.

Incluyo algunas referencias adicionales de la Teoría de Testores aunque ninguna fue utilizada directamente.

4. Un sistema de clasificación y de reconocimiento de patrones. R. Morales. Tesis de licenciatura en Matemáticas, Facultad de Ciencias, UNAM. 1988.

En este trabajo se desarrolla un algoritmo de escala exterior muy original.

5. Diukova, E.V., Acerca de un algoritmo para la construcción de los test típicos (en ruso). *Cibernética Matemática.* vol. I (1976).

6. Cheguis I.A. y Yablonskii S.V., Acerca de los test para esquemas eléctricos (en ruso). *Uspieji Matematicheskij Nauk.* 4(66) (1955) 182-184.
Aquí se introduce por primera vez el concepto de testor típico.

7. Zhuravliov Yu.I, Dmitriev A.N. y Krendeleiev F.P., Acerca de los principios matemáticos de la clasificación de objetos y fenómenos. *Diskretnyi Analiz.* **T7** (1966).

8. Ruiz Shulcloper J., Bravo Martínez A., Aguila Feros L. Algoritmos BT y TB para el cálculo de todos los tests típicos. *Revista de Ciencias Matemáticas.* **vol.VI** (1985) 11-18.

Este artículo presenta dos algoritmos de escala exterior.