UNIVERSIDAD POLITÉCNICA DE MADRID
FACULTAD DE INFORMÁTICA

LISTAS *KBM2L* PARA LA SÍNTESIS DE CONOCIMIENTO EN SISTEMAS DE AYUDA A LA DECISIÓN

TESIS DOCTORAL

JUAN ANTONIO FERNÁNDEZ DEL POZO
DE SALAMANCA

Madrid, 2006

UNIVERSIDAD POLITÉCNICA DE MADRID

Facultad de Informática

Departamento de Inteligencia Artificial

TESIS DOCTORAL

LISTAS *KBM2L* PARA LA SÍNTESIS DE CONOCIMIENTO EN SISTEMAS DE AYUDA A LA DECISIÓN

Autor: Juan Antonio Fernández del Pozo Salamanca

Licenciado en Informática

Directora: Concha Bielza Lozoya

Licenciada en CC. Matemáticas y Doctora en Informática

Madrid, 2006

Listas *KBM2L* para la Síntesis de Conocimiento en Sistemas de Ayuda a la Decisión

Departamento de Inteligencia Artificial
Facultad de Informática
Universidad Politécnica de Madrid

Resumen

La construcción, evaluación y explotación de los Sistemas de Ayuda a la Decisión mediante Redes Bayesianas y Diagramas de Influencia, entre otros modelos de razonamiento, supone el manejo de tablas con información diversa. Entre ellas destacan las tablas de probabilidad condicionada, que especifican la relación probabilística entre variables, y las tablas de decisiones óptimas resultantes de evaluar el modelo. Dichas tablas, que pueden ser muy complejas, recogen conocimientos estructurados de los dominios de aplicación sobre un conjunto de variables del modelo gráfico probabilístico.

Bajo el nombre de *KBM2L* introducimos técnicas para construir la base de conocimiento de los sistemas de ayuda a la decisión. Tratamos de explotar la *lista KBM2L* como herramienta útil de representación del conocimiento del sistema, constituido por la estructura gráfica del modelo, los modelos de probabilidad y de utilidad y el resultado de la evaluación. La representación gráfica es cualitativa e intuitiva y por tanto, los usuarios del sistema tienen fácil acceso al conocimiento si son expertos en el problema. Por el contrario, los modelos cuantitativos de probabilidad y utilidad y la evaluación no muestran el conocimiento fácilmente. Por una parte, los modelos codifican gran parte del conocimiento numéricamente mediante numerosos parámetros, y por otra, las tablas de decisiones óptimas son de gran tamaño. Ambos aspectos impiden reconocer las variables y relaciones fundamentales que describen la representación del conocimiento y que explican los resultados de la inferencia o evaluación.

Mientras que las tablas se pueden considerar como entidades estáticas, las listas *KBM2L* son representaciones del conocimiento dinámicas. Una configuración concreta de la lista determina la capacidad de explicación del conocimiento, la eficiencia para resolver las consultas al sistema de ayuda a la decisión desde diversos puntos de vista, y la complejidad de memoria requerida para gestionar la base de conocimiento.

La estructura de la lista permite revelar la granularidad del conocimiento de las tablas mientras que las configuraciones nos muestran el papel que las variables del modelo tienen en la evaluación inferida. La granularidad posibilita articular procedimientos para estructurar y comprender mejor el conocimiento que el sistema alberga en sus tablas. El papel de las variables en los diferentes contextos y en el conjunto del modelo nos facilita un mecanismo de generación de explicaciones del conocimiento y de las propuestas que hace el sistema, así como el análisis de sensibilidad del propio modelo.

La memoria se estructura como sigue. Tras la introducción exponemos los fundamentos de la lista *KBM2L* como representación del conocimiento y describimos los problemas de búsqueda de la representación óptima y diferentes propuestas de solución. Nos enfrentamos a un problema de optimización combinatoria que abordamos en el marco de las actuales metaheurísticas con algoritmos y métodos adaptados a nuestro objetivo. A continuación, mostramos la aplicación de las técnicas a tablas de decisiones óptimas y a tablas de probabilidad condicionada del diagrama de influencia. Finalmente proponemos realizar un análisis de sensibilidad del modelo mediante la extensión natural de la lista *KBM2L* usual con los parámetros de interés.

KBM2L List for Knowledge Sinthesis on Decision Support Systems

Artificial Intelligence Department
School of Computer Science
Technical Unversity of Madrid

Abstract

The implementation, evaluation and exploitation of Decision Support Systems by means of Bayesian Networks and Influence Diagrams, among other reasoning models, imply the use of tables with diversified information. Among them we focus on the conditional probability tables that represent the probabilistic relationships among variables and the tables of the optimal decisions resulting from the model evaluation. The tables, that can be very complex, include structured knowledge from the application domains over a set of variables of the probabilistic graphical model.

Under the name of *KBM2L* we introduce a technique to build the knowledge base of the decision support system. We try to exploit the *KBM2L list* as a useful tool for the knowledge representation of the system that includes the model graph, the utility and probability models and the evaluation output. The graphical representation is qualitative and intuitive and then the users can easily access the knowledge if they are experts on the problem. On the other hand, the quantitative models of probability and utility and the evaluation output do not show easily the knowledge because it is coded numerically, in the case of these models, and due to the huge size of the optimal decision tables, in the case of the evaluation output. Both aspects do not allow us to recognize the main variables and relationships that describe the knowledge and explain the results.

While the tables can be regarded as static objects or entities, *KBM2L* lists are dynamic knowledge representations. A specific list configuration determines the ability of knowledge explanation, the efficiency to solve queries to the decision support system from many differents points of view and the memory complexity required to manage the knowledge base. The structure of the list allows us to reveal the granularity of knowledge from tables while the configurations show us the role of the model variables in the inferred evaluation. The granularity provides procedures to structure and understand better the knowledge that the system hosts in its tables. The role of the variables in the different contexts and in the whole model give us a mecha-

nism to generate explanations of knowledge and of the system proposals and also the sensitivity analysis of the model itself.

After the introduction we show the foundations of the *KBM2L* list for knowledge representation and describe the problems of the optimal representation search and several proposals of solution. We face a combinatorial optimization problem that is dealt with algorithms and methods adapted to our objective in the framework of metaheuristics. Next, we show the application of these techniques to optimal decision tables and conditional probability tables of the influence diagram. Finally, we propose to perform a model sensitivity analysis by means of the natural extension of the usual *KBM2L* list with the meaningful parameters.

Dedicatoria

A Rosa.

Agradecimientos

Quiero expresar mi agradecimiento a las personas que han contribuido, de diferentes formas, a hacer posible la realización de esta tesis.

Primero, mis compañeros y colegas: Concha Bielza Lozoya (directora de tesis), Sixto Ríos Insua, David Ríos Insua, Manuel Gómez Olmedo.

La tesis es la oportunidad para aprender a investigar científicamente y un paso necesario en la comunidad universitaria para realizarse profesionalmente. Las dificultades que se presentan son muy grandes: elegir el tema principal, reconocer los temas relacionados, acotar el alcance del trabajo, marcar unos objetivos, documentarse, definir la técnica para obtener resultados y mostrarlos en la tesis y en publicaciones. El apoyo constante a todas las iniciativas y la promoción de mi trabajo por parte de Concha ha sido determinante para el desarrollo de la tesis, las publicaciones y la participación en conferencias y reuniones.

Mis compañeros del departamento: Luis Baumela Molina, Joaquín Fernández Martín, Jacinto González Pachón, María Isabel Rodríguez Galiano, Antonio Jiménez Martín, Alfonso Mateos Caballero, Arminda Moreno Díaz y Miguel A. Virto García.

Algunos compañeros de la Universidad: Denis Bouyssou, Raúl Cabestrero Alonso, Javier Díez Vegas, Simon French, Jesús González Boticario, Finn V. Jensen, Pedro Larrañaga y sus colegas, Peter Lucas, Alexis Tsoukias,...

Segundo, los Proyectos de Investigación que han contribuido a las publicaciones, congresos, cursos y otros recursos:

* Una Metodología de evaluación de la Calidad, 1997. COOPERS & LYBRAND y DAG-DIA-FI-UPM.

* Nuevos Desarrollos en Sistemas de Decisión Inteligentes Basados en Diagramas de Influencia: Aplicaciones a la Ictericia Neonatal y a la Asistencia Respiratoria Extracorpórea, 1998-1999. DGESIC (PB97–0856) (Dirección General de Enseñanza Superior e Investigación Científica).

* Nuevos Enfoques para el Desarrollo del Sistema IctNeo de Ayuda a la Decisión para la Ictericia Neonatal, 1999-2000. UPM–A9902.

* Contribuciones al Desarrollo de Sistemas de Decisión Inteligentes. Comunidad de Madrid, 2000-2002. 07T/0027/2000-2002, con la participación de UPM, URJC, UCIII y HGM.

* Un Sistema de Ayuda a la Decisión Multiobjetivo en el Tiempo con Imprecisión. Ministerio de Ciencia y Tecnología, 2001-2002. DPI2001-3731.

* Definición, Construcción, Implantación y Explotación de una Medida de la Calidad de los Servicios en la Fundación Gil Gayarre (FGG) (MCS-FGG). Plan y Proyecto de Formación en la Calidad de los Servicios en la FGG (FCS-FGG). 2001–2006. Convenio.

* Hacia la democracia electrónica. Toma de decisiones complejas a través de Internet, 2003-2006. European Science Foundation.

* eDemocracy: Internet-Aided Complex Decision-Making, 2004-2007. Ministerio de Educación y Ciencia (Contrato n^o TSI2004-06801-C04-04).

* Red temática de decisiones multicriterio, 2005 y 2006 Dirección General de Investigación del Ministerio de Ciencia y Tecnología, BF M2002 - 11282 E.

* Red Temática de modelos gráficos probabilísticos y aplicaciones, 2006. Dirección General de Investigación del Ministerio de Ciencia y Tecnología.

Ha sido especialmente significativo que la Sociedad Española de Estadística e Investigación Operativa (SEIO) me eligiese como representante español para participar en el Euro Summer Institute, ESI-XIX, curso de Inteligencia Artificial y Análisis de Decisiones en Toulouse (2001). En este encuentro tuve la ocasión de presentar por primera vez mi trabajo junto a otros jovenes investigadores.

Finalmente, la Universidad Politécnica de Madrid me ha financiado directamente la participación en congresos, más de 10.

Tercero, los medios técnicos (tras ellos hay personas). A Sun por poner una potente herramienta como JAVA al alcance de la universidad. A los equipos de desarrollo de R, Linux, MikTex, GeNie, entre otros. A Microsoft, aunque Windows y Office me han dejado colgado muchas veces, también hacen fácil la edición y manipulación de gráficos y la organización de datos. Por supuesto a http://citeseer.ist.psu.edu/, el buscador www.google.com, los portales de las revistas y las páginas personales de investigadores y grupos mediante los cuales he encontrado gran cantidad de información y software muy fácilmente.

Por último, los recursos de conocimiento: los autores que he citado como referencias (y otros: Morris Kline, Michio Kaku, Ian Stewar, Ilya Prigogine, Carl Sagan, Roger Penrose, Martin Gard-

ner, Stephen Hawking, Daniel Peña, Sixto Ríos, A.N. Kolmogorov,...) pues en sus referencias están las raíces de mi interés y placer por la investigación.

Un capítulo aparte para la dimensión más personal: a Rosa, mis Padres y Hermanos, sin olvidar a los Amigos (algunos los he mencionado al comienzo).

Índice general

Abreviaturas y Acrónimos[1]

□	Marca el Final de un Ejemplo.
ADR	Atributos de Dominio Restringido en un Item de una Lista *KBM2L*.
AS	Análisis de Sensibilidad del Modelo.
BC	Base de Conocimiento del SAD.
CA	Conjuntos Aproximados (Rough Sets).
CMRS	Cambio de Modalidad de Respuesta Secuencial en la Lista *KBM2L*.
coKB	Entrada Restringida en una Tabla.
DdO	Dimensión de Observación.
DI	Diagrama de Influencia.
EKBM2L	Lista *KBM2L* Extendida con Parámetros del Modelo que Origina la MM.
FNC	Forma Normal Conjuntiva.
FND	Forma Normal Disyuntiva.
IctNeo	DI para la Gestión del Protocolo de la Icteria Neonatal.
KBM2L	MM que Representa Conocimiento y es Transformada en una Lista.
MGP	Modelos Gráficos Probabilísticos.
MM	Matriz Multidimensional.
MR	Modelo de Referencia en el AS.
PGNHL	DI para la Gestión del Linfoma Gástrico Primario no-Hodgkin
RB	Red Bayesiana.
SAD	Sistema de Ayuda a la Decisión.
TDO	Tabla de Decisiones Óptimas en un MGP.
TPC	Tabla de Probabilidad Condicionada en un MGP.
unKB	Entrada Desconocida en una Tabla.
XBase	Cambio de Base de una Lista *KBM2L*.

[1]Significado de las abreviaturas y acrónimos utilizados en el texto. En el capítulo 1 (*Introducción*) no hacemos uso de abreviaturas para facilitar la lectura.

Capítulo 1

Introducción

A lo largo de este capítulo pretendemos introducir brevemente los sistemas de ayuda a la decisión como el contexto de las técnicas y herramientas propuestas en la tesis para la síntesis de conocimiento. Describiremos desde un punto de vista general qué es un sistema de ayuda a la decisión, cómo es la base de conocimiento y las áreas de conocimiento relacionadas.

Revisaremos conceptos relativos a aquellas disciplinas que abordan de manera genérica muchos de los problemas a los que se enfrentan los sistemas de ayuda cuando realizan su trabajo en un dominio de conocimiento concreto. Por ejemplo: clasificar, aprender, seleccionar, filtrar, simplificar, generalizar, buscar, reconocer patrones,... Nos referimos a las áreas de ciencias de la computación e inteligencia artificial (aprendizaje automático – representación del conocimiento, modelos de razonamiento,...–, descubrimiento en bases de datos y minería de datos), análisis estadístico multidimensional de datos (regresión logística, componentes principales, análisis discriminante, análisis de conglomerados,...), la optimización y la lógica. El catálogo de áreas pretende describir los objetivos de cada una, y dar una idea general de la metodología y los algoritmos fundamentales utilizados.

Finalmente, completamos la introducción a la tesis con la definición de los objetivos y un bosquejo de sus contenidos.

1.1. Sistemas de Ayuda a la Decisión

Esta sección intenta mostrar las interrelaciones entre la gestión de conocimiento, la ayuda a la decisión y la Inteligencia Artificial. En primer lugar, ayudar a tomar decisiones es uno de los objetivos de la gestión del conocimiento. Por otra parte, tomar decisiones implica construir modelos de los problemas y procesos de decisión y la inteligencia artificial nos suministra técnicas y herramientas para obtener dichos modelos.

La gestión de conocimiento es importante en las organizaciones, grandes o pequeñas, porque el conocimiento de los problemas relacionados con sus actividades y fines, y sus soluciones, es muy volátil pues reside en las personas expertas, está desestructurado, oculto y disperso en múltiples documentos y bases de datos, y además, el entorno cambia y hay que rediseñar procesos y redefinir objetivos. La toma de decisiones, sea en un entorno estable o cambiante, es una actividad transversal en las organizaciones que precisa de una adecuada gestión del conocimiento mediante sistemas de ayuda a la decisión.

En este trabajo nos ocuparemos de los sistemas de ayuda a la decisión que utilizan como herramienta de modelización del conocimiento los diagramas de influencia. Se trata de un tipo de modelo gráfico probabilístico en el que las decisiones y los objetivos del problema se representan explícitamente junto a un modelo probabilístico de la incertidumbre.

1.1.1. Gestión de Conocimiento, Ayuda a la Decisión e Inteligencia Artificial

Este trabajo es una aportación a la gestión de conocimiento desde la Inteligencia Artificial y aplicado al Análisis de Decisiones en el contexto de los sistemas de ayuda a la decisión. Un sistema de ayuda a la decisión es un sistema informático que apoya el proceso de toma de decisiones, ayudando a los decisores a formar y explorar las implicaciones de los juicios y, por tanto, a tomar decisiones basadas en el entendimiento (French, 1998). Pensamos que es muy adecuado situar la tesis en el área de la inteligencia artificial y el análisis de decisiones. El título de la tesis: "Listas *KBM2L* para la Síntesis de Conocimiento en Sistemas de Ayuda a la Decisión", es la combinación de tres elementos. El primero es el *objetivo* de "Síntesis de Conocimiento", el segundo es el *método* de "Listas *KBM2L*" y el tercero es el *ámbito* de los "Sistemas de Ayuda a la Decisión". Se trata pues de gestión de conocimiento avanzada que utiliza técnicas de inteligencia artificial (Liebowitz, 2001).

La toma de decisiones debe apoyarse en la gestión de conocimiento (Holsapple, 2001) y en particular: en los aspectos organizacionales, el descubrimiento en bases de datos y la distribución de los sistemas de ayuda a la decisión. La gestión de conocimiento puede mejorar la comprensión de la ayuda a la decisión y las capacidades de los sistemas de ayuda a la decisión en tanto que se ocupan de la representación y proceso del conocimiento por medio de personas, máquinas, organizaciones y sociedades. La inteligencia artificial, los sistemas basados en el conocimiento y, en particular, los sistemas de ayuda a la decisión emplean técnicas de gestión de conocimiento para modelizar los problemas y hacer posible la toma decisiones como actividad intensiva en conocimiento. Nuestro trabajo está en la línea de conseguir sistemas de ayuda a la decisión capaces de gestionar enormes cantidades de información y conocimiento involucrados en la construcción

y explotación de los mismos.

La evolución de los sistemas de ayuda a la decisión (Shim et al., 2002) ha sido muy importante, observándose la tendencia a incorporar desde los años 70 hasta la actualidad, toda la tecnología y desarrollo organizacional disponible para resolver problemas cada vez más complejos.

Los sistemas de ayuda a la decisión consistían inicialmente en ayuda a decisores individuales, una base de datos centralizada y homogénea y un algoritmo de optimización específico. En la actualidad estos tres elementos (decisores, bases de datos y algoritmos de optimización) han variado considerablemente.

En cuanto a los decisores, los sistemas de ayuda a la decisión son usados por grupos de trabajo o equipos multidisciplinares, especialmente equipos virtuales (humanos, sistemas de ayuda a la decisión, agentes de adquisición, almacenamiento, comunicación y/o proceso de datos), que operan en un entorno de trabajo cooperativo. Por tanto, la definición, aplicación e impacto de los sistemas de ayuda a la decisión es cada vez más amplia. La tecnología de comunicaciones permite deslocalizar los sistemas, los datos y los usuarios, e internet pone en comunicación mediante interfaces flexibles y potentes a todos los agentes involucrados de forma muy eficaz.

Respecto de las bases de datos, los sistemas de ayuda a la decisión operan hoy con bases de datos heterogéneas y las tecnologías de descubrimiento en bases de datos y de minería de datos permiten explorar conjuntos de datos muy grandes y complejos. (Shaw et al., 2001) señalaban que conjugar la gestión de conocimiento y la minería de datos constituye la clave para mejorar la gestión de las relaciones con los clientes. En nuestro caso lo es para optimizar las interrelaciones entre los agentes involucrados en los procesos de decisión, es decir, los decisores, los sistemas de ayuda a la decisión y los usuarios. De este modo, la toma de decisiones es más eficiente y flexible. Pero sobre todo es más eficaz, es decir, clara y robusta en los juicios, lo cual sitúa a los sistemas de ayuda a la decisión considerados en el nivel 3 de ayuda, en la escala propuesta en (French, 1998). Según dicha escala, en el nivel 1, que es el más sencillo, el sistema facilita soluciones que son óptimas en el marco del modelo del problema. Por otra parte, en el nivel 2 hacemos abstracción del modelo y junto a las soluciones óptimas el sistema muestra diversos escenarios que mejoran la comprensión del problema. Por último, en el nivel 3 el sistema incorpora la posibilidad de representar eficazmente los juicios de los decisores y esto permite tomar cierta distancia con el problema y tener una perspectiva más rica y profunda del proceso de toma de decisión.

Por último, respecto de la optimización de los modelos, la perspectiva actual contempla tres facetas: la formulación, la solución y el análisis. En el pasado, el interés se centraba en la obten-

ción de las *soluciones* óptimas mediante los algoritmos más eficientes. Actualmente los sistemas de ayuda a la decisión se diseñan teniendo en cuenta la *formulación* del modelo. La modelización se apoya en el empleo de gráficos, lenguajes de especificación formal y en la integración entre herramientas y entornos (Druzdzel y Flyn, 2000). Existen algunos entornos que permiten, entre otras tareas, formular y evaluar modelos gráficos probabilísticos: (Hugin, 2002; GeNIe, 2003; Elvira, 2002), los paquetes *deal* (Bøttcher y Dethlefsen, 2003) y (Weihs et al., 2005) del entorno R (R Development Core Team, 2005) y la extensión de *weka* (Witten y Frank, 2005) para clasificadores basados en redes bayesianas realizada por (Bouckaert, 1995). Se permite de este modo la posibilidad de plantear diferentes modelos o prototipos y seleccionar los más adecuados en un contexto cambiante definido por nuevos decisores, factores ambientales, culturales, ... La solución de problemas cada vez más complejos ha necesitado de una evolución metodológica que combina los algoritmos directos tradicionales (programación matemática) y modernas técnicas importadas de aprendizaje automático (algoritmos genéticos,...), la simulación (recocido simulado,...) y otras metaheurísticas (búsqueda tabú,...). Asimismo, se nos presenta la posibilidad, y también la necesidad, de realizar un *análisis* de las soluciones del modelo y de los modelos posibles. Es decir, por una parte, analizar la sensibilidad de la solución respecto de los datos concretos de un caso, y por otra, respecto de las características estructurales (parámetros,...) del modelo.

En el futuro la tendencia será potenciar las características que permiten abordar problemas cada vez más complejos de forma fiable, segura y escalable (Chenoweth et al., 2004). Los tres aspectos mencionados implican, desde nuestro punto de vista, la *distribución* del sistema de ayuda a la decisión en cuanto a los datos, el proceso (Díez y Mira, 1995) y el servicio o acceso (Sanders et al., 1999). La distribución de un sistema de ayuda a la decisión requiere profundizar en las características de multiplataforma, de estandarización, de personalización por parte de los usuarios y de seguridad física y relativa a la integridad de datos, restricciones de acceso y disponibilidad remota.

Finalmente, respecto a las expectativas de la evolución de los sistemas de ayuda a la decisión, (Burstein et al., 2001) recoge de la 5^a Conferencia Internacional de Sistemas de Ayuda a la Decisión (ISDDD'99, Melbourne, Australia) la influencia en estos sistemas de la gestión de conocimiento, los métodos y técnicas de integración de bases de datos heterogéneas ("Data WareHousing") y el análisis de datos cualitativos/verbales, es decir, textos, ontologías,... Por otra parte (Zopounidis, 2002) pone de manifiesto que el análisis de decisiones multicriterio es actualmente la aproximación más potente para fundamentar la construcción de los sistemas de ayuda a la decisión que pretendan reflejar los criterios de los expertos y analistas y explicar

claramente su razonamiento. En última instancia, una decisión es tomada por una persona o el consenso de un grupo y, en general, no responde a un único criterio de idoneidad.

1.1.2. Sistemas de Ayuda a la Decisión y Modelos Gráficos Probabilísticos

Disponer de la base de conocimiento es tener "información sobre la información" que pone en conexión todas las piezas de forma coherente, explica y justifica las partes constitutivas del sistema, organiza de modo eficiente un resultado junto a otros posibles resultados y relaciona los hechos con el problema inicial y todas las soluciones posibles. Las técnicas de representación del conocimiento capturan primero el conocimiento del problema mediante un modelo, por ejemplo un modelo gráfico probabilístico, en cierto paradigma, por ejemplo el análisis de decisiones, y después presentan el resultado de la evaluación al sistema de ayuda a la decisión, cuya base de conocimiento puede entonces responder a las consultas de los usuarios. Sin embargo, el resultado de la evaluación y algunos elementos del modelo pueden ser muy complejos.

Con el modelo gráfico probabilístico construido para un problema de decisión que implementa el sistema de ayuda a la decisión, o parte del mismo, planteamos en esta tesis sintetizar el conocimiento y traducir ciertos elementos del modelo, por ejemplo tablas de probabilidad condicionada, y de la evaluación, por ejemplo tablas de decisiones óptimas, al lenguaje del problema, para mejorar la explotación de la base de conocimiento. La principal motivación es que las bases de conocimiento deben proponer junto a las alternativas de decisión óptimas y distribuciones de probabilidad, mecanismos de explicación (de la alternativa óptima), validación (de las políticas óptimas y de las distribuciones), aprendizaje (de nuevo conocimiento y refinamiento del disponible) y análisis de sensibilidad del modelo. Los aspectos de aprendizaje y análisis de sensibilidad quedarán únicamente esbozados en la tesis, ya que por su complejidad definen líneas futuras de investigación.

Para algunos problemas, normalmente aquellos que involucran consecuencias complicadas entre beneficios y riesgos de la decisión (el tratamiento clínico, la contaminación, la inversión,...), los expertos pueden usar la inferencia del sistema de ayuda a la decisión, por ejemplo tablas de decisiones óptimas, para obtener ayuda en la determinación de la mejor alternativa. Sin embargo, los expertos pueden encontrar difícil de aceptar las recomendaciones del sistema de ayuda a la decisión si no conocen el razonamiento que hay tras el resultado indicado por el mismo y la estructura coherente de conocimiento que sostiene todas y cada una de las propuestas registradas en la base de conocimiento (Henrion et al., 1991). El experto se preguntará probablemente: *¿por qué es esta propuesta óptima?* y *¿cúales son las reglas implícitas que subyacen al modelo del problema de decisión?* La respuesta a estas preguntas suministra una explicación a

las decisiones del sistema. Éstas pueden dar nuevas intuiciones dentro del problema y sintetizar el conocimiento que puede servir para validar el sistema de ayuda a la decisión.

Los sistemas de ayuda a la decisión, desde el punto de vista de los sistemas basados en el conocimiento, recogen conocimiento del dominio y de control. Para ello definen y mantienen tablas con datos de naturaleza diversa y complejidad exponencial. La información de control la constituyen los algoritmos (asistidos o no por el usuario) de captura de datos, de inferencia y de presentación de resultados. La información del dominio que modeliza un sistema de ayuda a la decisión puede ser de tres tipos:

1 *información estructural*, ya sea en forma de relaciones, dependencias, causalidad, correlación, preferencias, valor o cuantificación de las relaciones;

2 *información derivada* mediante inferencias o propagación de hechos, e

3 *información factual*, es decir, hechos o evidencias.

En general, puede haber incertidumbre en la información, es decir, relaciones probabilísticas y/o hechos probables, e imprecisión en la formulación del modelo de representación del conocimiento manifiesta en los parámetros y en el modelo de razonamiento, a causa de algoritmos de inferencia o evaluación aproximados, del sistema de ayuda a la decisión.

En esta tesis estamos interesados en la información del modelo de representación del conocimiento que usa el sistema de ayuda a la decisión. Este modelo se compone, desde el punto de vista de la toma de decisiones, de:

1. una medida de preferencia sobre los objetivos,

2. alternativas de decisión y

3. una medida de incertidumbre sobre las variables influyentes en las decisiones y estados del problema.

Consideramos por tanto, que la información estructural de un modelo consiste en un marco conceptual de variables y la especificación de la relación entre ellas (Druzdzel y Flyn, 2000).

Las informaciones derivadas y factuales conciernen a la explotación del sistema de ayuda a la decisión basado en el modelo y son obtenidas, respectivamente, del propio sistema y del contexto del problema.

La implementación de los sistemas de ayuda a la decisión conduce a estructuras de datos que tabulan las informaciones referidas. Unas veces las tablas son estructuras internas del sistema. Por ejemplo, las distribuciones de probabilidad, las funciones de utilidad, los casos o los ejemplos

del dominio, las enumeraciones de excepciones o las restricciones relativas a cierto algoritmo o procedimiento de inferencia genérico,... Otras veces las tablas son los resultados o inferencias que se suministran a los usuarios. Por ejemplo, las tablas de decisiones óptimas, explicaciones mediante reglas, las tablas de probabilidad condicionada a posteriori,... Nuestro principal objetivo es describir procedimientos para manejar estas tablas de forma eficiente y poniendo de relieve las relaciones entre las variables del problema y el papel que desempeñan en el proceso de toma de decisiones.

Las listas *KBM2L* propuestas son estructuras nuevas, basadas en la secuenciación de los datos del sistema de ayuda a la decisión, adecuadas para minimizar el espacio de almacenamiento en memoria de las tablas y al mismo tiempo para buscar la mejor organización del conocimiento que es útil para generar explicaciones. Ambos aspectos son interesantes para disponer de sistemas eficientes de consultas en los sistemas de ayuda a la decisión y ayudar en la educción de conocimiento probabilístico.

Como ya indicamos al comenzar la sección, el tipo de sistema de ayuda a la decisión que consideramos se basa en un modelo gráfico probabilístico y en particular hablamos de *diagrama de influencia*. Un diagrama de influencia (Howard y Matheson, 2005) es un formalismo moderno del análisis de decisiones, frecuentemente adoptado hoy día como base para la construcción de sistemas de ayuda a la decisión, y usado para estructurar y evaluar problemas de toma de decisiones. Los diagramas de influencia consisten en una estructura de grafo dirigida y acíclica y distribuciones de probabilidad y funciones de utilidad que modelizan, respectivamente, la incertidumbre y las preferencias relativas al problema. El resultado de la evaluación son las tablas de decisiones óptimas. Así, para cada decisión del diagrama de influencia, hay una tabla de decisiones óptima que muestra la alternativa mejor, es decir, de máxima utilidad esperada, para cada combinación de variables. El algoritmo de evaluación determina cuáles son estas variables.

(Gómez, 2004) muestra una panorámica de las aplicaciones a problemas reales de los diagramas de influencia en diversas áreas: agricultura, economía, gestión de recursos naturales y medio ambiente, industria, medicina...

La toma de decisiones y los sistemas de ayuda a la decisión son complejos por naturaleza. Cualquier problema de decisión de cierta entidad, por ejemplo, una secuencia de decisiones que tiene presente decenas de conceptos en incertidumbre y múltiples objetivos, implica la construcción de un modelo que será la representación de un número significativo de escenarios asociados al problema. Concretamente, en un diagrama de influencia, los conceptos inciertos muestran sus interrelaciones mediante tablas de probabilidad condicionada, la valoración de estados y

decisiones del problema es recogida en una función de utilidad y las decisiones óptimas están disponibles al calcular y maximizar la utilidad esperada. En las tablas de probabilidad condicionada y tablas de utilidad (las tablas de decisiones óptimas se derivan de las últimas) encontramos la combinatoria que es consecuencia de la complejidad del problema inicial que describe nuestro modelo.

Por otra parte, hay dos fuentes adicionales de complejidad para los sistemas de ayuda a la decisión que se apoyan en redes bayesianas o diagramas de influencia para representar el conocimiento. En primer lugar, los atributos, que por naturaleza son continuos, si se discretizan, dan lugar a atributos discretos con un dominio grande que incrementa la combinatoria y la discretización (Kozlov y Koller, 1997) y puede introducir sesgos y anomalías en los resultados (Cano et al., 2000). En segundo lugar, es interesante considerar parametrizaciones del modelo que lo hagan más flexible y adaptable a diversos escenarios y grupos de decisores, respecto de algunas probabilidades, valores de la función de utilidad,... Es decir, el análisis de sensibilidad que introduce en las tablas de decisiones óptimas nuevos atributos continuos y discretos. El modelo puede tener unas decenas de atributos internos especialmente sensibles, pero en general son miles de parámetros, como en los sistemas *IctNeo* y *PGNHL* que expondremos más adelante. Se trata de un problema de una complejidad extraordinaria, pues la sola evaluación de un modelo gráfico probabilístico es, en el peor caso, un problema NP-completo (Cooper, 1990).

1.2. Modelos Gráficos Probabilísticos para Toma de Decisiones Clínicas

De entre los diversos sistemas de ayuda a la decisión, hemos trabajado especialmente en toma de decisiones clínicas. Este tipo de sistemas es importante por el alcance que tiene su uso en las decisiones concretas de tratamiento y prevención de enfermedades y por el impacto que tiene la generalización de esta tecnología en la sociedad en general. Esencialmente permiten gestionar mejor gran cantidad de conocimiento y de recursos como tiempo, equipos, medicamentos, personal,... Pero lo más importante es que transforman la relación tradicional entre pacientes y profesionales sanitarios dotándola de nuevas y mejores posibilidades: de comunicación (vía correo electrónico), interacción (vía internet), fiabilidad (vía sistema de ayuda a la decisión), retroalimentación (vía educación) y gestión (con menos coste, errores y tiempo) (Ball y Lillis, 2001). Nuevamente, y en este contexto concreto de aplicación, vemos que la distribución del sistema de ayuda a la decisión es esencial para explotar todo el potencial del sistema con éxito.

En este sentido, nuestro trabajo desarrolla una actividad de minería de datos sobre los

modelos y los resultados de su evaluación (inferencia). Por ejemplo, la propuesta de (Shaw et al., 2001) relativa al mercado (bases de datos de transacciones) opera sobre los datos observados, mientras que nuestro trabajo tiene como objeto un sistema de ayuda a la decisión genérico basado en un modelo gráfico probabilístico. Para mostrar el método y resultados recurrimos a modelos gráficos probabilísticos de protocolos clínicos. Los problemas técnicos que hemos encontrado en la construcción y análisis de estos modelos han sido una importante fuente de inspiración en este trabajo.

El primer ejemplo real que se utiliza en esta tesis estudia la aplicación de las listas *KBM2L* para encontrar y analizar minuciosamente los tratamientos óptimos para el linfoma gástrico no-Hodgkin (Eidt et al., 1994) que propone el modelo de (Lucas et al., 1998). Éste es un problema clínico difícil, principalmente por la incertidumbre que conlleva. La lista *KBM2L* resultante suministra explicaciones de alto nivel de los tratamientos óptimos para la enfermedad, y además nos facilita encontrar relaciones entre grupos de variables y tratamientos.

El segundo ejemplo es un modelo del problema de gestión de la ictericia neonatal (Newman y Maisels, 1992) realizado en el Hospital General Universitario Gregorio Marañón (Gómez, 2002; Gómez et al., 2006). Éste es un problema clínico que depende de muchas variables e implica una secuencia considerablemente larga de decisiones. Además, la ictericia neonatal afecta a una proporción grande de recién nacidos lo cual conlleva costes importantes que podrían evitarse en muchas ocasiones. Este diagrama de influencia ha sido una de las motivaciones principales del desarrollo de este trabajo.

En estos problemas mencionados, los modelos y los resultados son explotados más eficientemente en la medida que:

1. extraemos patrones del modelo mediante nuestro *análisis de dependencias funcionales* (Date, 1986) que induce la estructura de la lista *KBM2L* sobre la base de conocimiento e *identificamos clases* o perfiles de pacientes (clientes);

2. *describimos los contextos* significativos del problema (estados del protocolo–tratamiento) y las situaciones potencialmente conflictivas, es decir, donde observamos recomendaciones por parte del sistema que discrepan con la experiencia para *detectar desviaciones* del comportamiento esperado del sistema de ayuda a la decisión;

3. proponemos una *visualización* del conocimiento que complementa al modelo gráfico probabilístico y facilita un mapa de las bases de conocimiento.

En problemas reales, las tablas de decisiones óptimas son muy grandes y virtualmente imposibles de utilizar en la práctica. Teniendo en cuenta que los tamaños de las tablas son exponenciales

en el número de variables, la tarea de encontrar explicaciones es compleja desde el punto de vista computacional. Transformando las enormes tablas de decisiones óptimas en tablas más compactas conseguimos ahorrar memoria. Optimizamos el almacenamiento de la base de conocimiento y a la vez mostramos las explicaciones (Fernández del Pozo et al., 2005). Las tablas compactas, cualquiera que sea su contenido de información y procedencia, ofrecen una visión más profunda de las tablas originales (Niermann, 2005).

1.3. Áreas de Conocimiento Relacionadas

En esta sección exponemos unos apuntes acerca de las áreas de conocimiento relacionadas más estrechamente con esta tesis, tales como:

1. el aprendizaje automático,
2. la estadística e investigación operativa,
3. los modelos de gestión de datos,
4. la lógica.

De la estadística e investigación operativa nos interesan algunas parcelas: análisis multidimensional de datos y análisis de decisiones. En la gestión de datos, que es una extensa área de la informática, sólo nos interesa la explotación de los datos de cara al descubrimiento en bases de datos y la minería de datos, y no la definición del modelo de datos (relacional (Fernández, 1981), orientado a objetos (Chen, 1976), deductivo (Gallarie et al., 1984),...) ni las técnicas de gestión de los datos (Date, 1986) (almacenamiento, transmisión, recuperación, compresión,...). La base de datos para nosotros se reduce a un conjunto de datos con estructura plana, de una tabla, igual que las *hojas de datos (data.frame)* del software estadístico (R Development Core Team, 2005), por ejemplo.

Todas estas áreas tienen desde hace años un desarrollo muy grande tanto en el plano teórico como aplicado, en muchas ocasiones, a la construcción de mejores sistemas de ayuda a la decisión (Shim et al., 2002). Por tanto, esta sección no pretende profundizar ni ofrecer tan siquiera una visión general de las mismas; sólo presentará algunas de las conexiones que nos parecen relevantes para comprender el alcance del desarrollo de la tesis y los conceptos preexistentes sobre los que descansa nuestra teoría.

Las áreas de conocimiento relacionadas con el trabajo presentan los siguientes temas de interés para la tesis:

1. Respecto a la lógica, fundamentalmente nos referimos a la representación proposicional del conocimiento que, por otra parte, es la base de los modelos indicados en el área de aprendizaje automático (árboles de clasificación, listas de decisiones,...):

   a) lógica proposicional (Cuena, 1985; Delahaye, 1988; Zantema y Bodlaender, 2000; Zantema y Bodlaender, 2002; Flach y Lachiche, 1999);

   b) programación lógica con restricciones (Jaffar y Maher, 1994; Sterling y Shapiro, 1994).

   El paradigma proposicional es completo y consistente para representar el conocimiento (Mira et al., 2000). Con tablas y listas *KBM2L* seguimos dicho paradigma, aunque consideramos otros paradigmas lógicos para extender el marco presentado en la tesis para resolver consultas a las bases de conocimiento.

   La consideración de restricciones, en un sentido lógico, permite definir una semántica operacional programable que hace más sencillo construir soluciones en los modelos lógicos. En los sistemas de ayuda a la decisión nos encontramos frecuentemente restricciones en la descripción general de las soluciones factibles que hay en la combinatoria de las configuraciones asociadas al marco conceptual del modelo. Las restricciones causan asimetrías que dificultan la representación y evaluación de los problemas de decisión (Bielza y Shenoy, 1999).

2. En el área de estadística e investigación operativa estamos interesados en la modelización de los problemas (optimización, modelos gráficos probabilísticos, análisis multidimensional de datos) (Ríos-García, 1995), fundamentada en el análisis de decisiones.

   a) Investigación operativa (Taha, 2002; Ríos-Insua et al., 2004);

   b) Análisis de decisiones (Raiffa y Schlaiffer, 1961; Keeney, 1982; Clemen, 1996; Goodwin y Wright, 1998; Ríos-Insua et al., 2002);

   c) Modelos gráficos probabilísticos (Howard y Matheson, 2005; Pearl, 1988; Jensen, 2001);

   d) Análisis multidimensional de datos (Everitt y Dunn, 1992; Manly, 1994).

3. En cuanto al aprendizaje automático mencionamos las siguientes representaciones del conocimiento:

   a) árboles de clasificación (Quinlan, 1986; Clark y Niblett, 1989; Murthy, 1998);

   b) conjuntos aproximados (rough sets) (Wong et al., 1986; Pawlak, 1997);

   c) redes de neuronas artificiales (Mitchell, 1997);

   d) redes bayesianas (Pearl, 1988; Jordan et al., 1999; Castillo et al., 1997);

   e) otros modelos: listas de decisiones (Rivest, 1987), oblivious decision graphs (Kohavi, 1994), tablas de decisión clasificadoras (Kohavi y Sommerfield, 1998);

4. Los modelos de datos y la gestión de datos se dirigen a las estructuras de datos de los modelos gráficos probabilísticos para realizar:

   a) descubrimiento en bases de datos (Agrawal et al., 1993; Agrawal et al., 1996; Imielinski y Mannila, 1996; Fayyad, 1997; Starr et al., 1998; Piatetsky-Shapiro, 2000);

   b) minería de datos (Holsheimer y Siebes, 1994; Fayyad et al., 1996; Glymour et al., 1997; Haughton et al., 2003).

Las áreas de conocimiento consideradas están interesadas en estructuras de información parecidas a las tablas de los modelos gráficos probabilísticos como bases de datos, registros de patrones,... Por otra parte, los objetivos son análogos al que perseguimos en la tesis: resumir las tablas facilitando el almacenamiento y gestión, extraer conocimiento como reglas o restricciones, reconocer patrones, identificar independencias y relevancias,...; explicando el contenido de información de dichas estructuras. Unas áreas plantean cómo obtener estructuras de representación eficientes, explicar los datos, aprender de los datos,...; otras trabajan sobre modelos de decisión e inferencia, y otras utilizan técnicas semejantes de optimización y/o búsqueda. Pensamos que un objetivo central de estas áreas y de la tesis es revelar que las estructuras o representaciones del conocimiento y de la información pueden no ser óptimas debido al tamaño, la complejidad de la inferencia y de la explicación. Por ejemplo, con los árboles de clasificación o las redes bayesianas no sólo interesa un modelo que se ajuste bien a la información de los datos disponibles para construir el modelo; además, se busca que sea sencillo, comprensible, intuitivo y representativo del problema.

En el resto de la sección detallamos más la relación con las cuatro áreas de conocimiento citadas.

1.3.1. Aprendizaje Automático: Clasificación, Inferencia y Reconocimiento de Patrones

Encontrar explicaciones a las inferencias y a la propia estructura del modelo de representación del conocimiento es una meta que persiguen todas las áreas de la inteligencia artificial.

Extraer reglas, patrones importantes y tendencias y comprender qué dicen los datos, el llamado aprendizaje de los datos, es una tarea genérica de muchos modelos del aprendizaje

automático. Por consiguiente, es fácil suponer que nuestra propuesta en esta tesis tiene cierto parecido con alguna de estas técnicas. Por ejemplo, en clasificación (supervisada) del aprendizaje automático, deseamos aprender una aplicación de un vector de atributos en una variable clase. En nuestro caso, esta variable puede ser, por ejemplo, la alternativa óptima del problema de toma de decisiones. Así, nuestra aproximación parece tener cierta analogía con técnicas de extracción del conocimiento tales como las usadas en la construcción de árboles de clasificación (Duda et al., 2001). Clasificadores basados en árboles, como CART (Breiman et al., 1993), ID3 (Quinlan, 1986), C4.5 (Quinlan, 1993), muestran tener una particular utilidad con datos no métricos y sin información a priori acerca de la forma apropiada del clasificador (Duda et al., 2001). Sin embargo, nuestra aproximación presenta grandes diferencias como veremos en la sección 3.7.2.

La idea central de este trabajo es constatar que el contenido de una tabla no es conocimiento hasta que esté organizado de cierta manera, como un libro cuyo conocimiento no es sólo el contenido (las letras, palabras, signos, fórmulas, figuras,...) sino cómo está estructurado todo el material. A diferencia de los árboles de clasificación, nuestras listas no son construidas sino reorganizadas. El objetivo de los árboles es maximizar una puntuación de pureza de la clase, la cual no tiene sentido para nuestro "clasificador"porque éste no permite clasificaciones incorrectas. Para nosotros, todos los casos están ya correctamente clasificados y queremos explicar por qué están clasificados de esa manera. Traducimos este problema a encontrar la lista *KBM2L* más corta.

Además, las tablas de decisiones óptimas y las tablas de probabilidad condicionada, respectivamente, se construyen para algún otro fin diferente de la explicación buscada, es decir, son el resultado de la evaluación y la cuantificación de las relaciones de incertidumbre entre variables de un diagrama de influencia. Así, los datos no se recogen usando estrategias eficientes para responder a preguntas específicas; son datos observacionales frente a datos experimentales. Más aún, las filas de nuestras tablas recorren todas las configuraciones posibles de las variables. Y no puede haber repeticiones. Evidentemente no proceden de la clásica muestra de prueba cuidadosamente seleccionada en el laboratorio, que encontramos frecuentemente en aprendizaje automático y estadística. Por tanto, los valores desconocidos aparecen sólo en la variable clase, a diferencia de los métodos basados en árboles, con valores desconocidos en las variables. Esto ocurre cuando las tablas de decisiones óptimas sólo incluyen un subconjunto de la solución del problema completo, debido a problemas computacionales, conduciendo a políticas desconocidas. También ocurre en una tabla de probabilidad condicionada que está parcialmente asignada. Veremos a lo largo de la tesis cómo tratamos con estos valores.

Encontramos otras conexiones posibles entre nuestro trabajo y los conjuntos aproximados

(Pawlak, 1997). Éstos se usan para la inducción de reglas en aprendizaje automático o como técnica para la eliminación de redundancias en análisis de datos. El algoritmo de construcción de conjuntos aproximados se basa en la teoría de conjuntos y en la topología y es un algoritmo de aprendizaje inductivo. Nosotros estamos más interesados en un algoritmo deductivo.

Los árboles de clasificación, las redes bayesianas, los modelos lineales generalizados (Agresti, 1996), las listas de decisiones y los grafos de decisión inconsciente son clasificadores que se pueden combinar con la lista *KBM2L*. Ésta puede organizar las reglas de clasificación obtenidas o caracterizar los grupos que determina el clasificador entre los datos de entrada. Además, podemos usarla para comparar clasificadores que dan lugar a distintas listas. Por último, podemos usar la lista *KBM2L* para clasificar (véase el ejemplo 3.12 en el capítulo 3).

Las técnicas del vecino más próximo (K-NN) (Mitchell, 1997) tocan de lleno con el planteamiento de la lista, que tiene en cuenta la distancia entre celdas de memoria, pero la distancia en la lista *KBM2L* es unidimensional y realmente sólo considera la adyacencia. Estas técnicas son la base de algoritmos de agrupación o "cluster" y de optimización con búsqueda local. La lista *KBM2L* realiza agrupación de datos y el proceso de optimización se apoya en un sistema de entornos de proximidad entre listas que proponemos en la sección 3.4.

Respecto de las redes de neuronas artificiales (Mitchell, 1997), destacamos que se pueden usar para reconocimiento de patrones. En este sentido conectan con la técnica presentada en la tesis mediante listas *KBM2L* pues, por ejemplo, la lista detecta perfiles o prototipos de pacientes en las tablas de decisiones óptimas. Las redes neuronales son muy eficientes pero tienen dificultades para explicar los resultados. Además, encontramos una analogía entre la regla de activación de las neuronas y el cálculo del *desplazamiento* de los casos de la lista (véase la sección 2.2.1). La neurona pondera las señales de entrada y la lista pondera el valor de los atributos de la tabla. En el primer caso resulta la señal de salida si se supera el umbral. En el segundo caso se obtiene como respuesta el contenido de la tabla. El mecanismo de la lista es más restrictivo pero es trazable y la explicación es transparente.

Mencionamos las máquinas del vector soporte (Cristianini et al., 2001; Burges, 1998), pues se trata de una técnica de clasificación reciente que está siendo aplicada con éxito en minería de datos a problemas de muy alta dimensionalidad, como la clasificación de documentos de texto y de datos bioinformáticos. De forma análoga a las listas *KBM2L*, las máquinas del vector soporte pueden obtener la clasificación mejor en un espacio de mayor dimensionalidad que el de los datos originales.

Para terminar, haremos mención al software relacionado con la tarea de clasificación. Los algoritmos de aprendizaje automático en JAVA (Witten y Frank, 2005) y los algoritmos de cla-

sificación (Elvira, 2002) son dos ejemplos de implementaciones de algoritmos del aprendizaje automático. Las versiones más recientes de paquetes estadísticos clásicos incorporan herramientas de aprendizaje automático para explorar los datos (S-PLUS, 2003). La representación gráfica de datos multidimensionales es importante en la fase de exploración y el software de aprendizaje automático debe permitir la presentación gráfica de los datos (Swayne et al., 1998; Brunk et al., 1997).

1.3.2. Estadística e Investigación Operativa: Modelos Gráficos Probabilísticos y Análisis Multidimensional de Datos

La estadística nos da una perspectiva más analítica del modelo de los datos mientras que el aprendizaje automático es más descriptivo y exploratorio (Glymour et al., 1997).

En este trabajo encontramos varias cuestiones que conectan con la estadística. En primer lugar nos enfrentamos a datos multidimensionales masivos que pretendemos describir y resumir. (Niermann, 2005) propone una aproximación al problema de descripción de las tablas multidimensionales de datos que presenta grandes analogías con nuestra propuesta para las tablas involucradas en la construcción y evaluación de los modelos gráficos probabilísticos. En concreto, destaca que cuando la tabla se presenta con un orden adecuado de sus filas (casos) y columnas (dimensiones–atributos) se revelan patrones intrínsecos de la información y el resultado o contenido de la misma es mucho más fácil de leer e interpretar.

En principio no deseamos reducir la dimensión, aunque como consecuencia de la transformación de las tablas en listas *KBM2L* pueden desaparecer dimensiones en toda la lista o en algunos contextos por irrelevancia. La reducción de la dimensión en análisis multidimensional de datos se puede realizar, por ejemplo, mediante componentes principales y análisis discriminante (Manly, 1994), pero en las tablas que consideramos no hay repeticiones y la correlación entre variables es propiamente una relación lógica en nuestro caso.

En parte, la descripción de los datos la planteamos mediante la construcción de agregados de datos y la consiguiente clasificación de los mismos. La clasificación puede ser supervisada o no supervisada según sean o no conocidas las clases. Nosotros siempre trabajamos con datos clasificados correctamente por el modelo gráfico probabilístico. Es decir, siempre tenemos una clasificación supervisada de partida para la construcción de la base de conocimiento. En concreto, como hemos señalado en la sección anterior, podemos ver el resultado de evaluar un diagrama de influencia, las tablas de decisiones óptimas, como una aplicación del vector de atributos en una variable de clasificación, la alternativa óptima del problema de toma de decisión. Se trata de un clasificador supervisado parametrizado por un modelo probabilístico y una función de

utilidad. Para esta tarea podemos considerar las técnicas de análisis multidimensional de datos de regresión logística y los modelos lineales generalizados.

Mencionamos también el software empleado en el análisis multidimensional de datos: (R Development Core Team, 2005; S-PLUS, 2003), y para estudiar modelos gráficos probabilísticos para análisis de decisiones y clasificación: (GeNIe, 2003; Hugin, 2002; Elvira, 2002; Witten y Frank, 2005).

1.3.3. Bases de Datos: Descubrimiento en Bases de Datos y Minería de Datos

La discusión entre técnicas de aprendizaje automático y estadística sugiere establecer nuestro marco en el campo de la minería de datos, véase (Hand et al., 2001). La minería de datos es un área interdisciplinar que ha crecido rápidamente y cuyas herramientas juegan un papel importante en aprendizaje automático, reconocimento de patrones, tecnologías de bases de datos, estadística (Glymour et al., 1997), entre otras áreas.

La minería de datos se puede enmarcar en las áreas del aprendizaje automático y estadística, pero nosotros consideramos que esta técnica responde realmente a la necesidad de estudiar grandes masas de datos para comprender su naturaleza u origen. En este contexto, la estadística puede ser sólo una herramienta que da soporte a los resultados, o proporciona modelos de los datos compatibles con dichos resultados. El aprendizaje automático, por su parte, facilita posibles formalismos de presentación del resultado de la exploración de los datos que realiza la minería de datos, en busca de patrones, hipótesis y conocimiento.

Más aún, podemos ver la minería de datos como parte del proceso de descubrimiento en bases de datos (Fayyad et al., 1996; Piatetsky-Shapiro, 2000) donde, frente al cálculo intensivo, es más relevante la consulta intensiva (Starr et al., 1998). Aunque es raramente documentada fuera de la literatura de las bases de datos, los típicos conjuntos de datos masivos en minería de datos demandan una especificación explícita de las estrategias de gestión de los datos, es decir, la forma en la cual los datos son almacenados, indexados, accedidos,... En esta tesis tendremos en cuenta estas cuestiones.

La minería de datos pretende encontrar patrones en los datos (Cios et al., 1998) y se lleva a cabo de forma empírica, y a diferencia del análisis de datos tradicional, no presenta hipótesis a priori sino que las genera a partir de los propios datos. Por otra parte, la minería de datos trabaja con conjuntos de datos de gran tamaño que están registrados sin el propósito de ser analizados. En definitiva se desencadena un proceso de descubrimiento que puede conducir a resultados (hipótesis) erróneos. Por esto la minería de datos se enmarca en el descubrimiento en bases de datos que reúne diversas técnicas para preprocesar y limpiar los datos y validar

los resultados (Fayyad, 1997). La minería de datos se ocupa de diferentes tipos de datos (binarios, numéricos, nominales,...), y propone modelos y patrones (reglas, árboles,...) mediante los que resuelve problemas de clasificación, agrupación,... Debido a la naturaleza de los problemas reales es imprescindible que los modelos contemplen la incertidumbre, y la obtención del modelo adecuado, por ejemplo, una red bayesiana, una red de neuronas artificiales, o un árbol de clasificación, implica realizar una búsqueda mediante algoritmos de aprendizaje y optimización combinatoria.

(Haughton et al., 2003) recogen una revisión del software para minería de datos que ilustra de forma colateral las características usuales de las diferentes tareas propias de un análisis de esta clase. Realiza una primera aproximación a estas técnicas mediante las posibilidades de importación de datos en diversos formatos, su descripción estadística y los gráficos de exploración. En segundo lugar aborda la construcción de modelos explicativos y predictivos tales como: regresión, árboles de clasificación, redes bayesianas, redes de neuranas artificiales,... Por último, todos los paquetes permiten definir dos conjuntos de datos a partir del conjunto inicial: el conjunto de datos de *entrenamiento* que permite derivar el modelo y el conjunto de datos de *test* que permite validar el resultado. Entre los paquetes de software más potentes para realizar minería de datos están SAS (http://www.sas.com/technologies/analytics/datamining/) e IBM (http://www-306.ibm.com/software/data/iminer/), que por supuesto facilitan las tareas de descubrimiento en bases de datos y DataWareHousing (gestión conjunta de bases de datos con arquitecturas y/o paradigmas diferentes y geográficamente dispersas).

Queremos señalar la importancia de dos tareas esenciales en descubrimiento en bases de datos y minería de datos. Éstas son la *Selección de Variables* y la *Selección de Casos* (Freitas, 2002). Las dos tratan de simplificar el conjunto de datos conservando esencialmente el conocimiento que se quiere extraer. Ambas presentan una gran semejanza con aspectos clave propuestos en la tesis. La selección de variables con la explicación, porque pretende reducir la dimensionalidad de los casos al subconjunto de atributos relevantes en la consulta al sistema de ayuda a la decisión. La selección de casos con la construcción de la lista *KBM2L*, porque precisamente trata de determinar un subconjunto de los datos (casos) que representa a todos. En nuestro caso, además, podemos recuperar el conjunto original, al menos en teoría.

Para terminar, esperamos con las técnicas propuestas aportar alguna solución al problema de la explotación de los resultados derivados del descubrimiento en bases de datos y de la minería de datos. Es habitual encontrarse con cientos o miles de reglas de asociación o con modelos muy complejos (redes bayesianas, redes de neuronas, modelos de regresión,...) imposibles de validar que discrepan en las respuestas. Actualmente, el análisis de los resultados que permita la toma

de decisiones y la recomendación de acciones guiadas por dichos resultados es un tema abierto.

1.3.4. Lógica

La lógica tiene una importancia fundamental en inteligencia artificial desde sus comienzos. Sustenta de forma explícita, en mayor o menor medida, la representación del conocimiento y la inferencia en todas las herramientas de representación utilizadas en los modelos de razonamiento (Cuena, 1985) y espacios de hipótesis del aprendizaje automático (Mitchell, 1997).

En nuestro caso destacamos que nuestra propuesta se puede encuadrar claramente en una representación del conocimiento mediante la lógica proposicional (Holsheimer y Siebes, 1994), véase la sección 3.7. Más aún, la lista *KBM2L* puede considerarse una implementación de un compilador de conocimientos de un cierto problema (Kautz y Selman, 1991). Esta representación tiene la ventaja de tener un cálculo consistente, completo y decidible (Mira et al., 2000) basado en la regla de resolución (Robison, 1965). Por otra parte, es menos expresiva, por ejemplo, que la lógica de predicados de primer orden, que es semidecidible, o las lógicas de orden superior, que son incompletas (Gödel, 1976).

La lista *KBM2L* pretende superar el inconveniente de la explosión combinatoria de la enumeración explícita de las proposiciones y suministrar un procedimiento de explicación fundamentado en la lógica proposicional. Ésta es la clave de la transparencia de los resultados que muestra la lista, pues representa el conocimiento en el mismo marco conceptual del modelo gráfico probabilístico, pero de forma concisa y destacando las relaciones lógicas entre conceptos que subyacen a las relaciones probabilísticas originales.

Finalmente, en relación con el diseño y verificación automática de circuitos integrados *VLSI*, (Hu y Dill, 1993) presentan estructuras para manejar la complejidad de los sistemas combinacionales y secuenciales (hardware que evalúa operaciones aritméticas y lógicas e implementa autómatas finitos). La lógica booleana se emplea para representar las funciones aritmetico–lógicas y los autómatas muy complejos mediante enormes tablas. Algunas soluciones adoptadas en este campo son bastante similares a las que hemos propuesto para la representación del conocimiento en los modelos gráficos probabilísticos en este trabajo. En este sentido observamos en (Kohavi, 1995b) una propuesta de espacio de hipótesis del aprendizaje automático que importa herramientas de la tecnología digital para el aprendizaje inductivo de clasificadores supervisados. En nuestra propuesta no hemos partido de las herramientas de representación y diseño de sistemas digitales, pero vemos una cierta convergencia en los aspectos formales, pues compartimos problemas y objetivos. La sección 3.7 mostrará con mayor detalle las conexiones entre las propuestas de la tesis y las representaciones proposicionales.

1.4. Objetivo y Estructura de la Tesis

Este trabajo propone mostrar técnicas de representación del conocimiento sobre las tablas del modelo de los problemas de decisión. La utilización de estas técnicas supone una mejora en la explotación del modelo al que dotan de capacidades de explicación y análisis de sensibilidad. Por otra parte, la construcción del diagrama de influencia es asistida por la organización de sus distintas tablas con las listas *KBM2L*, facilitando la adquisición y estructuración del conocimiento y la validación. El objetivo de la tesis es *gestionar las tablas y extraer la información de forma suficientemente comprimida para poder suministrar fácilmente explicaciones.* Para ello disponemos el desarrollo siguiente por capítulos.

En el capítulo 2 exponemos los fundamentos de las técnicas de representación, aportando un vocabulario, sintaxis y semántica. La sección 2.1 delimita las fuentes de la información que manejamos, las tablas multidimensionales, que representan de forma simple, relaciones, funciones y datos del modelo. La sección 2.2 describe la matriz multidimensional como estructura de datos intermedia en la que se apoya la transformación que logra la representación propuesta. La sección 2.3 define las listas *KBM2L* como implementación eficiente de una base de conocimiento para un sistema de ayuda a la decisión. La sección 2.4 explota la estructura de la lista y sus componentes para dotar de expresividad y capacidad de explicación a la lista. En la sección 2.5 proponemos la representación gráfica de las listas, que junto a diferentes vistas de la información proporciona un medio de visualización de grandes conjuntos de datos. La sección 2.6 muestra una posible implementación de las listas mediante reglas de gestión de los datos involucrados en las operaciones básicas que se pueden hacer con las listas, para construirlas y mantenerlas. La implementación propuesta determina, en parte, la semántica operacional de la representación y las capacidades relacionadas, como aprendizaje y explicación.

El capítulo 3 se ocupa del problema del almacenamiento óptimo de las tablas mediante listas *KBM2L*. La sección 3.1 define qué entendemos por almacenamiento óptimo y cuál es la solución al problema. La sección 3.2 expone los detalles del cambio de base o de configuración de una lista *KBM2L*, ya que es el procedimiento básico de mejora del almacenamiento. La sección 3.3 recoge varias heurísticas relacionadas con la búsqueda de las soluciones descritas en la sección 3.1. La sección 3.4 desarrolla una estrategia de búsqueda local, de optimización, basada en la conocida *heurística de entorno variable*, adaptada a nuestro problema. La sección 3.5 propone un *algoritmo genético* como estrategia de búsqueda global de la solución. La sección 3.6 muestra sobre un conjunto de listas con tamaños diversos los resultados de experimentar con los algoritmos propuestos. La sección 3.7 muestra una comparación de la representación mediante listas *KBM2L* con otras estructuras, como las matrices dispersas y algunas representaciones

proposicionales del conocimiento.

El capítulo 4 trata de ilustrar el uso de las listas *KBM2L* con las tablas de decisiones óptimas. La sección 4.1 describe cómo se transforma y optimiza una tabla de decisiones óptimas y un conjunto de dichas tablas. La sección 4.3 plantea un sistema de consultas a la base de conocimiento del sistema de ayuda a la decisión, en nuestro caso una lista *KBM2L*, donde la ayuda consiste en facilitar la información al usuario con el menor coste. En las secciones 4.2 y 4.4 recurrimos a dos sistemas de ayuda a la decisión reales (*PGNHL* e *IctNeo*) para aplicar las técnicas que introducimos en este trabajo.

El capítulo 5 trata de ilustrar el uso de las listas *KBM2L* con las tablas de probabilidad condicionada. Las secciones 5.1 y 5.2 definen la transformación y correspondencia entre las tablas de probabilidad condicionada y dos formas alternativas y equivalentes de listas. La sección 5.3 propone varias aplicaciones de la representación mediante listas de las relaciones de probabilidad condicionada, a la educción, validación y extracción de conocimiento. En las secciones 5.4 y 5.5 ilustramos los métodos propuestos en la sección 5.3 con los sistemas de ayuda a la decisión *PGNHL* e *IctNeo*, respectivamente.

El capítulo 6 pretende abordar al análisis de sensibilidad del modelo original. La sección 6.1 expone brevemente el estado de la cuestión y la motivación. La sección 6.2 introduce la extensión de la lista *KBM2L* para realizar análisis de sensibilidad del modelo. La sección 6.3 propone el algoritmo y el análisis de los resultados. La sección 6.4 ilustra mediante un ejemplo sencillo las dos secciones precedentes.

El capítulo 7 resume las conclusiones y la continuación posible de este trabajo. La sección 7.1 recoge los resultados y aportaciones de este trabajo. La sección 7.2 plantea un conjunto de posibles desarrollos para una investigación futura.

En resumen, nuestro método está digirido a la tarea de construir, mejorar y hacer accesible el sistema de ayuda a la decisión. Concretamente, a gestionar el almacenamiento, la consulta y el análisis de las bases de conocimiento. Suponemos que hay unos sistemas de ayuda a la decisión, en cierta organización, y los queremos dotar de mayor capacidad de síntesis, explicación y aprendizaje.

Capítulo 2

Fundamentos de la Lista *KBM2L*

Este capítulo 2 presenta los fundamentos de esta tesis y desarrolla nuestra propuesta de representación del conocimiento en el área de los sistemas de ayuda a la decisión (SAD), más concretamente relacionados con los modelos gráficos probabilísticos (MGP). El objetivo es analizar la complejidad de memoria de las bases de conocimiento (BC) tabulado y proponer ciertas técnicas de aprendizaje automático que permitan abordar la representación de diferentes fuentes de conocimiento sintetizando las tablas mediante las listas *KBM2L* que aquí introducimos.

En la sección 2.1 el punto de partida es la información tabulada multidimensionalmente, su acceso mediante índices y el almacenamiento mejorado mediante listas ordenadas de proposiciones. El almacenamiento optimizado de la BC es el objeto del próximo capítulo.

A lo largo de las restantes secciones desarrollamos la semántica de las listas *KBM2L*, su capacidad de representación, y la sintaxis, así como los detalles de la implementación, estructuras de datos y gestión básica de la información mediante dichas listas.

Específicamente, en la sección 2.2 consideramos matrices multidimensionales (MM) que recogen la información de las tablas. Una tabla puede ser almacenada matricialmente de muchas formas y proponemos una serie de técnicas para transformarla en una BC eficiente para el SAD.

En la sección 2.3 nos centramos en el concepto y aplicación de las listas *KBM2L* que descansa sobre dos ideas: la proyección unidimensional de la tabla, que define un orden en la información, y la coalescencia de respuestas adyacentes, que determina agregados o ítems de información. El resultado es una representación compacta de la tabla.

En la sección 2.4 analizamos la estructura de la lista: los casos (datos atómicos) y los ítems, las piezas de conocimiento que definen la lista *KBM2L*. Describimos el acceso a la información mediante índices y caracterizamos las componentes del índice de los ítems. El análisis de los índices nos permite por una parte describir la dinámica de la tabla al considerar diferentes

matrices, y por otra, desarrollar una explicación del contenido de la BC.

En la sección 2.5 presentamos gráficas (espectros y mapas) de las listas. Se trata de representaciones gráficas de datos multidimensionales que complementan las diferentes vistas de la BC.

Finalizamos, en la sección 2.6 desarrollando en detalle la gestión de casos e ítems que implementa la estructura de la lista y hace posible su construcción.

2.1. Tablas Multidimensionales

La construcción, evaluación y explotación de un SAD mediante redes bayesianas (RB) y diagramas de influencia (DI), entre otros modelos de razonamiento aproximado, supone el manejo de tablas con información diversa. Algunas de estas estructuras de datos son: las tablas de probabilidad condicionanada (TPC) y las tabla de decisiones óptimas (TDO) resultantes de evaluar el modelo. Dichas tablas, que pueden ser muy complejas, recogen conocimientos estructurados de los dominios de aplicación.

Las tablas son un marco natural para la adquisición y representación del conocimiento en el área de los sistemas de información. Las tablas son uno de los espacios de hipótesis del aprendizaje automático más simples que podemos encontrar (Kohavi, 1995a). Básicamente registran condiciones, reglas, casos,... Permiten recolectar información sobre un dominio que puede ser posteriormente clasificada, por ejemplo mediante un árbol (Murthy, 1998), grupos (clusters) (Manly, 1994), RB (Buntine, 1996),...

Debemos considerar sin embargo, que una tabla equivalente a un árbol de clasificación tiene complejidad exponencial respecto de dicho árbol (Zantema, 1999), en función de su profundidad y su factor de ramificación. Similar resultado obtenemos respecto de una RB (Cooper, 1990). Es decir, la explotación de la tabla mediante una RB o un árbol es una tarea más sencilla que manejar la tabla explícitamente, pero el aprendizaje o construcción de la RB o el árbol es complejo.

El espacio de almacenamiento asociado a una tabla de datos con atributos discretos y dominio finito es el producto de las dimensiones o cardinal del dominio de los atributos. En general, el producto cartesiano de los conjuntos que definen los dominios es inmenso. En definitiva, la construcción de las tablas en problemas reales tampoco es una tarea sencilla.

En esta tesis, describiremos procedimientos para manejar las tablas y extraer conocimiento de las estructuras. El almacenamiento compacto de la información es posible si comprimimos los ficheros de las tablas, pero debemos indicar que no estamos interesados en comprimir los datos (Davies, 2002). Nuestro objetivo es estructurar el conocimiento y resumir la información para

presentarla al usuario o a ciertos componentes del SAD de forma eficiente.

2.1.1. La Tabla Multidimensional

Desde el punto de vista del ordenador convencional, la tabla es la estructura de datos más importante. Las instrucciones y los datos de los procesos están tabulados tanto en la memoria principal como en los discos. Otras estructuras de datos auxiliares proporcionan estructuras lógicas de acceso y manipulación de los contenidos específicos (árboles, listas,...). Parece interesante, desde el punto de vista computacional, tener en cuenta la estructura de *tabla* como referencia para gestionar la información, procesarla de forma óptima y extraer conocimiento, especialmente si debemos trabajar con grandes cantidades de datos.

Vamos a introducir algunos conceptos para la representación, almacenamiento y explotación del conocimiento tabulado. En primer lugar vamos a formalizar la representación definiendo: la tabla, el esquema, la base, el índice de acceso a la información y la respuesta (su dominio y las etiquetas ignorancia (unKB) y restricción (coKB)). Otros conceptos, como las operaciones que permiten modificar (ampliar, reducir, combinar,...) tablas y dar fundamento algebraico a transformaciones y operaciones entre listas, se introducirán en el capítulo 3, que se dedica a exponer la dinámica de la lista *KBM2L*.

Tablas, Esquema y Respuesta En primer lugar vamos a formalizar algunos conceptos usuales en el área del aprendizaje automático, que podemos encontrar por ejemplo en (Kohavi, 1995b). Hemos observado algunas diferencias entre conceptos que se expresan con términos idénticos en el análisis de decisiones y el aprendizaje automático. Por ejemplo, la tabla de decisión en análisis de decisiones es una representación para un problema de toma de decisiones, mientras que en aprendizaje automático es una tabla de verdad (o su generalización para atributos no binarios) con valores *falso* y *cierto*. El árbol de decisión es un clasificador en el contexto del aprendizaje automático, mientras que en análisis de decisiones es una representación equivalente al DI. En cualquier caso, los conceptos siguientes son de uso común aplicados a tablas de datos, observaciones y proposiciones.

La **tabla**, considerada como estructura de datos, consta de un conjunto de **atributos** (dimensiones) que determinan un concepto, propiedad o acción. Al conjunto de atributos lo llamaremos **esquema** de la tabla. Al conjunto de valores que puede tomar un atributo lo denominamos **dominio**:

Esquema: $\{Atrib_0, Atrib_1, \ldots, Atrib_{\delta-1}\}$
Dominio: $Atrib_i : \{v_{i,0}, v_{i,1}, \ldots, v_{i,\delta_i-1}\}$

donde hay δ atributos, con δ_i $\forall i = 0, \ldots, \delta_i - 1$ valores distintos en cada respectivo dominio. En general, consideramos esquemas y dominios finitos, y por tanto, desde el punto de vista de la lógica, la tabla es disyunción de conjunciones proposicionales, es decir, un modelo en forma normal disyuntiva (FND).

Llamaremos **respuesta** al contenido de la tabla, que puede ser simple (escalar) o compuesta (vectorial o conjunto) y cuantitativa, cualitativa o mixta. Cada línea o fila de la tabla es un **ejemplo**, **instancia**, **caso** o **proposición** (conjuntiva de los atributos) del conocimiento recogido en la misma. Planteamos que la respuesta frente a los atributos de la tabla se puede interpretar como conocimiento de cierto dominio, una proposición. Por ejemplo, "si la analítica es positiva, los antecedentes del paciente son desfavorables y la edad es crítica, aunque no presente síntomas claros, se le ingresa en observación". En el dominio médico el *ingreso en observación* es un valor de la respuesta. Las respuestas (y dominios) son: analítica (positiva,...), antecedentes (desfavorables,...), edad (crítica,...) y síntomas (claros,...).

En los problemas reales nos encontramos que las tablas pueden tener algunas proposiciones cuya respuesta no es conocida (la tabla es incompleta) o cuya semántica es inconsistente (la tabla es asimétrica en el sentido de que ciertas combinaciones de atributos que determinan una entrada no son posibles). Consideramos un valor especial del dominio de la respuesta llamado **unKB** para las entradas con respuesta desconocida en las tablas incompletas. Nos parece interesante considerar otro valor especial llamado **coKB** para las entradas imposibles en las tablas que presentan restricciones en la combinatoria (producto cartesiano) de los valores de sus atributos. La naturaleza de las restricciones puede ser semántica (referida a atributos del esquema original de la tabla) o sintáctica (referida a atributos artificiales). En los diferentes ejemplos que mostramos, denotamos estas etiquetas numéricamente con los valores enteros -1 para *unKB*, y -2 para *coKB*, y codificamos el dominio de la respuesta y de los atributos con 0,1,2,..., siempre que no se indique otra notación.

Base **e Índice** Llamamos **base** (*Base*) al vector cuyas componentes son los atributos o dimensiones del esquema ordenadas conforme a una secuencia (natural o convencional). Para el orden de los atributos en el esquema de la tabla dado inicialmente, la *Base* es llamada **canónica**. Nótese que el factorial del cardinal del esquema es, en principio, el número de *Bases* diferentes que podemos considerar. Denotamos con i_j al atributo componente de la *Base* que está en la posición j-ésima, $j = 0, 1, \ldots$ Así, una *Base* de 5 atributos es $[0, 1, 4, 3, 2]$, y se tiene $i_0 = 0$, $i_1 = 1$, $i_2 = 4$, $i_3 = 3$, $i_4 = 2$. Cuando sean pocos atributos, si es más claro, utilizaremos los nombres que tengan en el esquema. De esta forma, una *Base* de 3 atributos con esquema $\{A,B,C\}$ es $[A, C, B]$, donde $i_2 = B$.

Llamamos **índice** al vector coherente con la *Base* cuyas componentes son valores del dominio de los atributos. Los índices se interpretan como coordenadas en cierta *Base* para el contenido de la tabla. En las tablas los índices son conocidos, no siendo así en un conjunto de datos, donde cualquier observación puede tener algún atributo desconocido. Es decir, todas las componentes del índice toman algún valor del dominio. Denotamos con c_j los valores de las componentes del índice y como hemos indicado antes, con i_j a las componentes. Por ejemplo, en una *Base* con 3 atributos $[Atrib_0, Atrib_1, Atrib_2]$, el índice tiene 3 componentes i_0, i_1 y i_2. Normalmente codificamos los valores del dominio del índice mediante números enteros del 0 a $\delta_i - 1$. Si los dominios son respectivamente $\{a_{0,0}, a_{0,1}\}$, $\{a_{1,0}, a_{1,1}, a_{1,2}\}$ y $\{a_{2,0}, a_{2,1}, a_{2,2}, a_{2,3}\}$, entonces un índice genérico es (c_0,c_1,c_2) donde c_i puede tomar los valores de su respectivo dominio y que codificamos $a_{i,j}$ $\forall j$.

La representación de la información tabulada tiene una interpretación lógica, operativa y geométrica. Las tablas almacenan valores que representan objetos descritos mediante un conjunto de atributos del esquema que implican su comportamiento o propiedades (interpretación lógica). La tabla es equivalente a otras representaciones, como árboles de clasificación y funciones lógicas (Mitchell, 1998). La información se almacena como contenido en la correspondiente celda de la tabla asignando el valor determinado por el conjunto de atributos evaluados en ciertos valores (interpretación operativa). La asignación es la acción que fija la respuesta correspondiente en una celda de la tabla.

A las dos interpretaciones anteriores añadimos otra de carácter geométrico. El vector de atributos evaluados o instanciados se corresponde unívocamente con las coordenadas enteras de las celdas de la tabla en el **sistema de referencia** asociado al esquema. Los atributos son las dimensiones de la tabla. El contenido se interpreta como la propiedad de la celda o punto cuyas coordenadas determina el índice en el espacio de representación de la información. La celda o punto es el espacio para almacenar la respuesta asociada a una instancia de los atributos del esquema. Una celda es referenciada por su índice y viceversa.

Sobre la interpretación geométrica construiremos la estructura de la lista *KBM2L*, que se introducirá más adelante. La interpretación lógica nos conducirá a la formulación de explicaciones, véase la sección 2.4.1, y en la interpretación operativa encontraremos el procedimiento de realización de consultas a la BC, véase la sección 4.3.

Las tablas representan la relación explícita entre el contenido o respuesta y el esquema. Además, tal y como veremos, expresan o sostienen relaciones implícitas entre los atributos del propio esquema: importancia relativa, relevancia, restricciones, explicaciones, reglas, afinidad de atributos y valores del dominio, patrones de respuesta, dependencias probabilistas,... Ambas

relaciones, explícitas e implícitas, son conocimiento. Las primeras son observables (datos) mientras que las segundas se extraen de la estructura y organización de los datos. El Cuadro 2.1 muestra el esquema y disposición de la información por filas.

Cuadro 2.1: Tabla genérica de *Respuesta*; $\{r_k\}$ es el contenido de la tabla y $\{v_{i,*}\}$, donde $*$ es el valor del *Atributo* índice

$Atrib_0$	$Atrib_1$	$Atrib_2$	...	$Atrib_{\delta-1}$	Respuesta
...	...	...	...	...	...
$v_{0,*}$	$v_{1,*}$	$v_{2,*}$	...	$v_{\delta-1,*}$	r_k
...	...	...	...	...	...

Veamos mediante ejemplos algunas de las ideas y conceptos anteriores.

Ejemplo 2.1. Consideramos una TDO, asociada a una decisión sencilla T, que tomamos en función de dos atributos con información escalar. T tiene tres alternativas que llamamos t_0, t_1 y t_2. Los atributos A y B tienen 4 y 5 valores respectivamente. Los valores de A son $\{a_0, a_1, a_2, a_3\}$ y los valores B son $\{b_0, b_1, b_2, b_3, b_4\}$. Para simplificar la notación escribimos el subíndice y por el contexto sabemos a qué se refiere. A continuación mostramos la tabla de doble entrada que expresa el valor de T en función de A y B.

$[A, B]$	0	1	2	3	4
0	0	0	0	0	1
1	1	1	1	1	1
2	1	1	1	1	1
3	0	0	1	2	1

Por ejemplo, dados los atributos $A = a_2$ y $B = b_3$ se propone tomar la decisión t_1. La respuesta (información de la tabla) es escalar y cualitativa. □

Ejemplo 2.2. Consideremos la tabla de dos atributos con información vectorial. La siguiente tabla contiene información vectorial mixta. El esquema es el de la tabla anterior. Representa una supuesta TDO proporcionando dos decisiones y la utilidad máxima esperada. Así, la respuesta tiene tres componentes: la utilidad máxima esperada, la primera y la segunda decisión de cierto problema de decisión.

$[A, B]$	0	1	2	3	4
0	(0.05,0,0)	(0.12,0,1)	(0.10,0,2)	(0.32,0,1)	(0.02,1,0)
1	(0.25,1,0)	(0.14,1,2)	(0.80,1,0)	(0.34,1,2)	(0.0,1,0)
2	(0.02,1,0)	(0.17,1,0)	(0.21,1,0)	(0.42,1,0)	(0.12,1,3)
3	(0.01,0,0)	(0.43,0,0)	(0.32,1,3)	(0.71,2,1)	(0.20,1,0)

Se trata de una extensión de la tabla descrita en el anterior ejemplo. Por ejemplo, dado el atributo $A = a_2$ y $B = b_3$, se propone desarrollar la política (t_1, t_0) con utilidad máxima esperada 0.42. La respuesta es un vector de componentes cualitativos y una valoración cuantitativa, en este caso la utilidad esperada. □

Las tablas con respuesta vectorial pueden utilizarse como representación del conocimiento en cadenas de decisiones, por ejemplo al evaluar un DI con varios nodos de decisión.

2.1.2. La Matriz Multidimensional

Dada una tabla que se define mediante un esquema y asignaciones de respuesta a las celdas, las coordenadas de una celda dependen de la *Base*. Veremos que la *Base* supone una gran ventaja, pues para referenciar cada celda no hay que mencionar los pares $(ATRIBUTO, valor)$, sino que el orden de los valores manifiesta la correspondencia con los atributos. La enumeración de los atributos permite ordenar parcialmente las celdas (y respuestas). Además, con la enumeración de los dominios de los atributos este orden es total.

Una tabla se puede representar matricialmente de varias formas según la *Base* elegida, dado el esquema. Al seleccionar una *Base* obtenemos una expresión matricial multidimensional de la tabla y se fijan las coordenadas de las celdas y sus respuestas. En las secciones 2.2 y 2.3 describimos cómo precisamente mediante matrices, el almacenamiento en memoria de una tabla puede transformarse en una lista.

Pretendemos construir una estructura para representar eficientemente tablas mediante matrices. Como veremos posteriormente, tratamos de minimizar el espacio que asignamos a los elementos de la matriz en su estructura óptima que depende de la *Base*.

Insistimos en que el mencionado sistema de referencia asume que existe un orden implícito en los atributos, el de una *Base*. También debe existir un orden en la lista de valores que tiene el dominio de cada atributo. Ambos órdenes son convencionales y arbitrarios y determinan la *Base*. Tal vez el orden del dominio de los atributos, cuantitativos o cualitativos, sea relevante. Las diferentes *Bases* implican diferente almacenamiento y manipulación de la tabla pero con idénticos contenidos y complejidad en memoria. La consideración de ambos órdenes, en los atributos y en los dominios de valores respectivos, es una idea fundamental.

La búsqueda de la *Base* más eficiente no es un problema sencillo. Lo abordaremos como un problema de optimización combinatoria y dedicamos el próximo capítulo a su análisis y a presentar soluciones. El problema de encontrar una tabla de tamaño mínimo equivalente a una tabla dada es NP-completo (Zantema, 1999). La **equivalencia** entre tablas se define en el sentido de que ambas tablas representan el mismo conocimiento, por ejemplo la misma función lógica. Hay muchas transformaciones de una tablas que dan lugar a tablas equivalentes, por ejemplo, reordenar filas y columnas (atributos) (Niermann, 2005) en tablas de datos multivariantes (no son tablas completas y hay repeticiones). Nosotros, en nuestro caso, proponemos reordenar las columnas (atributos) determinando el orden en las filas. Veremos que obtenemos resultados muy interesantes para las TDO que encontramos en los DI.

2.2. Matrices de una Tabla

La BC es el sistema de información usado por el SAD para sugerir las acciones recomendadas y suministrar explicaciones. Si ésta debe ser eficiente (responde a preguntas y construye explicaciones satisfactorias), debe estar implementada con una estructura próxima al dominio del problema más que al formalismo de representación y/o inferencia. En el caso de los SAD que nos interesan, el conocimiento está representado mediante un DI. Esta herramienta incluye tablas diversas como las TPC y las del resultado de la evaluación, las TDO, que son un conjunto de tablas que relacionan las decisiones del problema con los escenarios o estados descritos mediante las variables. Las tablas son la materia prima de la BC que proponemos.

Tratamos de encontrar las relaciones inducidas por la respuesta entre los atributos que determinan la tabla. Éstas sirven en primer lugar para minimizar el espacio de almacenamiento y en segundo lugar para explicar de modo práctico el contenido de la propia tabla.

Las tablas tienen un tamaño exponencial en el número de atributos de su esquema. Por ejemplo, en el SAD *IctNeo* sobre ictericia neonatal (Gómez, 2002), algunas de las TDO exceden la capacidad de almacenamiento de cualquier sistema informático actual y ello motivó querer expresar su contenido por medio de otra representación, que minimice el tamaño. También las TPC pueden ser muy grandes para obtenerlas mediante una asignación manual sistemática, o a partir de un conjunto de datos poco significativo, véase la sección 5.3.1. Por otra parte, es interesante tratar de resumir el contenido y extraer características de las tablas que puedan analizar los expertos, véanse las secciones 5.3.3 y 5.3.2.

Es decir, la representación, la adquisición y extracción del conocimiento tabulado en los DI son las cuestiones que pretende abordar este trabajo. En este sentido, tenemos motivaciones similares a la tesis de (Kohavi, 1995b), si bien, en aquélla el enfoque es estadístico y de apren-

dizaje automático muy general. Nosotros nos centramos en el análisis de decisiones y los SAD mediante el espacio de hipótesis propuesto en la sección 2.3.

2.2.1. Conceptos Básicos

Comenzamos con algunas definiciones. El orden de los atributos en las tablas originales depende de cómo se hayan obtenido éstas. Si se trata de una TDO, depende básicamente de la secuencia de borrado seguida al resolver el DI (Shachter, 1986). Si se trata de una TPC, el orden lo establece quien construye/aprende la tabla. Un cambio en el orden altera la posición de la enumeración de variables del esquema, pero las proposiciones de la tabla quedan invariantes. Si imponemos el orden de una cierta *Base* (los componentes del esquema y los respectivos dominios de valores) podemos definir el índice de acceso a las posiciones del contenido o respuesta.

Cuando establecemos un sistema de referencia podemos enumerar el contenido de la tabla sobre una MM. Ahora hay un elemento primero, un elemento último y una relación de adyacencia, que no es exactamente de proximidad pues las componentes del índice (valores del dominio de los atributos) no son necesariamente cuantitativas, aunque estén codificadas como números enteros. De modo que el contenido de la tabla es almacenado en celdas con coordenadas $\overrightarrow{c} = (c_0, c_1, \ldots, c_{\delta-1})$ en la $MM[c_0, c_1, \ldots, c_{\delta-1}]$. En una MM los valores están ordenados por medio de una aplicación $f : N^{\delta} \rightarrow N$, donde

$$f(\overrightarrow{c}) = c_0 * \prod_{k=1}^{\delta-1} \delta_k + c_1 * \prod_{k=2}^{\delta-1} \delta_k + ... + c_{\delta-2} * \delta_{\delta-1} + c_{\delta-1} = q \tag{2.1}$$

que provee el **desplazamiento** q de la posición de un valor con ciertas coordenadas en el vector unidimensional respecto del primer elemento (en la posición cero) en una *Base* dada. Luego, el desplazamiento de un dato es una coordenada relativa a la propia tabla que además depende de la *Base*. Esto es lo que hace usualmente un compilador de lenguaje de programación de alto nivel para alojar los elementos de la matriz en la memoria física (RAM o Fichero). El almacenamiento consiste en tratar la MM como un vector unidimensional y utilizar el desplazamiento como referencia o localización de los datos. Para mayor detalle acerca de estructuras abstractas de datos y su implementación se puede consultar la referencia clásica (Knuth, 1968) y un manual de programación, por ejemplo (Kernighan y Ritchie, 1991).

Para cada atributo índice, δ_i es el cardinal de su dominio y los productos $W_{j-1} = \prod_{k=j}^{\delta-1} \delta_k$, los llamamos **pesos** de los atributos. $\overrightarrow{w} = (W_0, W_1, \ldots, W_{\delta-1})$ es el vector de pesos y sus componentes son calculadas de modo recurrente según:

$$W_j = \delta_{j+1} * W_{j+1}, j = 0, 1, \ldots, n-1, W_{\delta-1} = 1. \tag{2.2}$$

El vector *Base* determina el $\overrightarrow{w}$ y diferentes *Bases* implican, en general, $\overrightarrow{w}$ diferentes para los atributos. La forma funcional del desplazamiento es un producto escalar del vector índice y el vector de $\overrightarrow{w}$:

$$\overrightarrow{c} * \overrightarrow{w}^T = q \tag{2.3}$$

La expresión (2.3) es igual que (2.1) pero más compacta y recurriremos a ella en el capítulo 4 para formalizar las consultas a las BC construidas mediante listas *KBM2L*.

Llamamos **desarrollo** de un índice a la instanciación iterada de todos los valores de sus componentes en el orden convenido en la *Base*. El desarrollo de los índices permite recorrer la matriz a través de sus dimensiones de forma sistemática según indica la *Base*. Simbólicamente, en la componente c_j su desarrollo es $for(c_j)$. Si se desarrollan todos los índices, entonces se recorre la matriz conforme está almacenada en la memoria linealmente. Por ejemplo el índice (c_0, c_1) de dominio binario tiene el desarrollo:

$\{(0,0),(0,1),(1,0),(1,1)\} = \{for(c_0)\{for(c_1)\}\}$.

Los W_j son los coeficientes que multiplican las coordenadas en (2.1), igual que las potencias de la base en la expresión del valor numérico en un sistema posicional (interpretación algebraica). El significado de W_j, desde el punto de vista del acceso y asignación de memoria, es el número de celdas que separan dos valores que solamente difieren en una unidad de la coordenada j (interpretación geométrica), es decir, los valores $(c_0, c_1, \ldots, c_j, \ldots, c_{\delta-1})$ y $(c_0, c_1, \ldots, c_j + 1, \ldots, c_{\delta-1})$.

Nótese que hay dos posibles implementaciones de (2.1): ordenar la magnitud de los pesos de mayor a menor de izquierda a derecha *(desplazamiento$_{LR}$)* y otra equivalente, de derecha a izquierda *(desplazamiento$_{RL}$)*. Ambas formas de desplazamiento están relacionadas mediante el orden simétrico de los atributos, y el mismo orden de valores de los respectivos dominios. Es decir, una *Base* $B_a = [0, 1, 2, \ldots, \delta - 1]$ tiene una *Base* simétrica $B_a^{-1} = [\delta - 1, \delta - 2, \ldots, 0]$. Entonces $\overrightarrow{w}_{LR} = (\delta_1 * \delta_2 * \ldots * \delta_{\delta-2} * \delta_{\delta-1}, \ldots, \delta_{\delta-1}, 1)$ y $\overrightarrow{w}_{RL} = (1, \delta_{\delta-1}, \ldots, \delta_1 * \ldots * \delta_{\delta-2} * \delta_{\delta-1})$, y se puede calcular el desplazamiento como

$$q_{LR,B} = \overrightarrow{c}_{LR,B_a} * \overrightarrow{w}^T_{LR,B_a} = \overrightarrow{c}_{RL,B_a^{-1}} * \overrightarrow{w}^T_{RL,B_a^{-1}} = q_{RL,B_a^{-1}}.$$

Aquí lo que importa es entender en qué orden se desarrollan los índices para dar lugar a un desplazamiento determinado. En caso contrario no sabemos qué significa el que un atributo tenga un W_j alto. Así, esta igualdad aclara que un W_j alto, el último siempre en el sentido RL, equivale a ordenar la tabla primero por ese último atributo. Tras la observación anterior vemos que en realidad hay $\delta!$ posibles implementaciones de (2.1), que calculan el mismo desplazamiento. Esta simetría inherente al orden de la información indexada muestra que cambiar el orden de los atributos, la *Base*, no altera el orden de las celdas de información si el W_j se cambia solidario con el atributo. Todas las configuraciones de la información tabulada son, en este sentido, iguales. La idea que proponemos es establecer el W_j de los atributos de acuerdo con el orden de la *Base* y fijar el orden de magnitud de los W_j. La implementación elegida, que es arbitraria, es LR. El orden elegido aquí es el que está en analogía con el sistema de asignación de valor absoluto a la representación numérica posicional a los números naturales, en el orden lexicográfico usual. Recordemos, por ejemplo, que el valor de 15 se obtiene de la expresión $1 * 10 + 5 = 15$.

El orden de la *Base* sabemos que es convencional. El orden inicial o natural de los atributos se asocia a la *Base* canónica. Como consecuencia, una vez fijada una asignación de W_j (LR u otra) el mayor o menor W_j de los atributos en la *Base* también es convencional. Pero los desplazamientos de las celdas de información cambian cuando realizamos un cambios de base ($XBase$), véase la Figura 2.1, y en la proyección unidimensional de la MM, las celdas cambian la adyacencia que tienen entre ellas en las diferentes *Bases*.

La Figura 2.1 muestra el desplazamiento de seis *Bases* para un esquema de tres atributos A, B y C con 2, 3 y 4 valores, respectivamente, para cada uno. Consideramos el desplazamiento de la *Base* canónica [A,B,C] (en la diagonal) comparado con el desplazamiento asociado a las *Bases* [B,A,C], [C,B,A], [A,C,B], [C,A,B] y [B,C,A]. Las gráficas muestran la posición relativa de las celdas en las distintas *Bases*. Una celda tiene por adyacentes a las (una o dos) que tienen un desplazamiento más próximo.

También podemos calcular el desplazamiento q por medio de la expresión recursiva:

$$\alpha_{k+1} = (\alpha_k + c_k) * \delta_{k+1}, \alpha_{\delta-1+1} = q \tag{2.4}$$

donde $\alpha_0 = 0$, $\delta_{\delta-1+1} = 1$ y $k = 0, 1, 2, \ldots, \delta - 1$. La expresión algebraica (2.1) es útil para recuperar un dato de la tabla, realizar una pregunta a la BC, porque los W_j muestran la importancia de los atributos y la estructura del espacio de representación del conocimiento para todas las instancias del índice o pregunta del usuario. Por otra parte, la expresión recursiva (2.4) es mejor para el cálculo porque es menos compleja (computacionalmente) y permite el

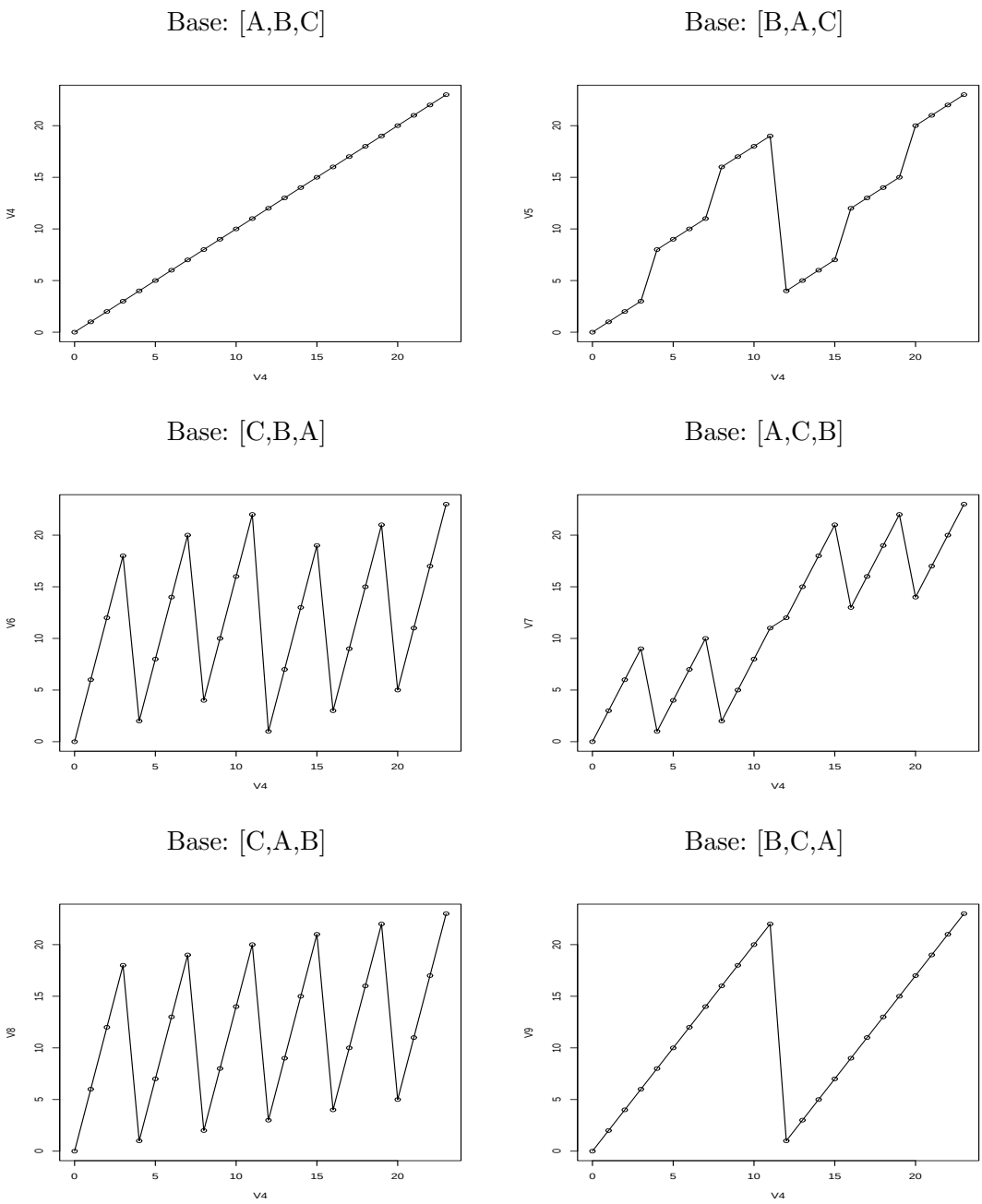

Figura 2.1: Cambio de desplazamiento asociado al $XBase$, la $Base$ canónica es [A,B,C] y el desplazamiento es $c_0 * 4 * 3 + c_1 * 4 + c_2$. Observamos que la $Base$ determina la relación de adyacencia entre celdas

$XBase$ fácilmente, pues no es necesario recalcular los W_j. Llamamos desplazamiento *pregunta* (sección 4.3) a la expresión (2.1) y desplazamiento *interno* (sección 2.6) a la expresión (2.4).

Así, dada una celda $\overrightarrow{c}$ podemos calcular su posición en memoria q y recuperar su valor. Y

a la inversa: el índice de cada celda puede reconstruirse a partir del desplazamiento, dada cierta $Base$: si $f_B(c_0, c_1, \ldots, c_{\delta-1}) = q$, entonces $f_B^{-1}(q) = (c_0, c_1, \ldots, c_{\delta-1})$, que se calcula mediante operaciones aritméticas con el algoritmo iterativo siguiente. Dados q y $\overrightarrow{w}$:

$$c_j = Ent((q - q_{j-1})/W_j)$$

donde $q_{j-1} = \sum_{k=0}^{j-1} c_k * W_k$ y $q_{-1} = 0$. Se trata en resumen de un mecanismo de asignación de memoria (puntero), respecto de una dirección relativa, posicional.

En el caso especial de que todos los atributos tengan igual número de valores, el vector de $\overrightarrow{w}$ es invariante frente al $XBase$ o permutación de los atributos. Esta situación es interesante y puede plantearse *cuadrar*, es decir, homogeneizar el cardinal de los atributos al máximo del esquema, u otro valor que sea mayor, mediante la introducción de restricciones sintácticas y dimensiones artificiales. Se trata de un problema abierto que describiremos con mayor profundidad en la sección 5.2 del capítulo 5. También es posible hacer que todos los atributos sean binarios vía descomposición del dominio, véase (Fernández del Pozo y Bielza, 2002b). Así, la estructura de las matrices es regular y su manipulación sintáctica (el $XBase$) es más simple, y su semántica (explotación de la tabla) flexible. Concretamente en minería de datos (Fayyad et al., 1996) (Mannila, 2000) es habitual transformar el conjunto de datos a un esquema binario que permite trabajar a los algoritmos más eficientemente con independencia del dominio de aplicación. Análogamente, los algoritmos genéticos tradicionales (Holland, 1975) codifican la información en binario aduciendo razones teóricas (maximización del número de esquemas (Goldberg, 1989)) que garantizan la eficacia del algoritmo. En cualquier caso, las ventajas computacionales (velocidad de proceso y sencillez del diseño de algoritmos) al modificar el esquema pueden ser inconvenientes en las tareas de interpretación de las estructuras resultantes.

2.2.2. Configuraciones de una Tabla

Una primera observación nos muestra dos clases de $XBase$: permutación de atributos y permutación de dominios. Y la combinación de ambas permutaciones. Una segunda nos sugiere la descomposición del esquema (orden del índice) en componentes (por ejemplo dos componentes, una con las $(i_0, \ldots, i_{\lfloor(\delta-1)/2\rfloor})$ y la otra con el resto). Una tercera nos sugiere la descomposición de atributos o **derivación** del esquema, donde un atributo da lugar a atributos que se derivan del dominio original. Los esquemas nuevos deben ser compatibles con el original, que debe poder ser recuperado de forma biunívoca. Por ejemplo, el atributo A con valores a_1, a_2 y a_3 se descompone en $A1$ con valores a_1 y $a_2 \vee a_3$ y $A2$ con valores a_2 y $a_1 \vee a_3$, véanse más ejemplos en (Fernández del Pozo y Bielza, 2002b). Claramente hay infinidad de posibilidades en las tres aproximaciones mencionadas, que además podemos combinar.

Más concretamente, dada una MM que almacenamos en una lista de celdas de memoria consecutivas, podemos preguntarnos: ¿de cuántas maneras podemos almacenar la misma matriz? ¿Qué relación hay entre dicho número y todas las posibles permutaciones de las celdas?

No vamos a abordar todas las transformaciones citadas, dada su complejidad semántica. Nos centraremos casi siempre en una versión simplificada de la primera clase de $XBase$: permutar los atributos del esquema dejando los órdenes de los dominios inalterados. Relajar esta restricción introduce una gran complejidad como veremos a continuación. Por otra parte, la primera clase de $XBase$ ha dado buenos resultados en las aplicaciones que hemos realizado, véanse los capítulos 4 y 5, posiblemente porque los esquemas contribuyen a una representación adecuada del conocimiento.

Consideramos la siguiente clasificación de los $XBase$ respecto una $Base$ de referencia o canónica. Si el esquema permanece intacto en una $Base$, el $XBase$ lo llamamos **natural**, aunque cambie el orden canónico de los dominios. Dados δ atributos tenemos $\delta!$ bases naturales, donde tan sólo permutan los atributos. Al orden de atributos lo llamamos **base simple**. Si fijamos el orden de los dominios, entonces los $XBase$ posibles son **simples**. Si consideramos la permutación de los órdenes en los dominios de los atributos, el número de imágenes en la memoria que representan cierta información concreta es como máximo $\delta! \prod_{i=0}^{\delta-1} \delta_i!$.

Ejemplo 2.3. Sea el esquema $\{A, B, C\}$ con $\delta_A = 2$, $\delta_B = 3$ y $\delta_C = 4$. Entonces, $3! * (2! * 3! * 4!) = 6 * 2 * 6 * 24 = 1728$ es el número de $Bases$ que se obtiene al permutar el orden delos atributos y el orden de los dominios. Observamos que es mucho mayor que el número de $Bases$ simples: $3! = 6$. □

Así, dada una matriz de δ dimensiones binarias: $\delta! \prod_{i=0}^{\delta-1} 2 = \delta! * 2^{\delta}$, es el número de formas distintas en que podemos almacenar la misma información siguiendo la primera clase de $XBase$. Es una cantidad muy superior a $\delta!$ asociada a la restricción anterior que deja el orden de los dominios fijo. El número de celdas de la matriz anterior es 2^{δ} y los datos pueden estar ordenados en memoria de $(2^{\delta})!$ formas diferentes. Pero de todas estas configuraciones (permutaciones) de la información, solamente unas pocas corresponden con la tabla y muchas son idénticas si la respuesta es discreta y con un número pequeño de modalidades. El ejemplo mostrado en la Figura 2.1 da los valores siguientes:

Permutan los atributos	$3!$	6
Permutan los atributos y sus dominios	$3! * 2! * 3! * 4!$	1728
Permutan las celdas de la MM	$(2 * 3 * 4)! = 24!$	$\sim 620 * 10^{21}$

En la sección 2.3 definiremos la estructura que explota las configuraciones para compactar las tablas.

Ejemplo 2.4. Tomemos la siguiente tabla con dos atributos con el esquema $\{A, B\}$, dominio binario de A: $\{a_0, a_1\}$ y B: $\{b_0, b_1\}$ y consideremos como configuraciones las dos posibles *Bases*: $[A, B]$ y $[B, A]$, sin permutar el dominio, y las *Bases*: $[A_{1,0}, B_{0,1}]$, $[A_{0,1}, B_{1,0}]$ y $[A_{1,0}, B_{1,0}]$, donde permuta el orden de los dominios. El conjunto de configuraciones posibles contempla además las *Bases*: $[B_{1,0}, A_{0,1}]$, $[B_{0,1}, A_{1,0}]$ y $[B_{1,0}, A_{1,0}]$, es decir, 8 *Bases*. T representa la respuesta.

Base $[A, B]$			*Base* $[B, A]$			*Base* $[A_{1,0}, B_{0,1}]$			*Base* $[A_{0,1}, B_{1,0}]$			*Base* $[A_{1,0}, B_{1,0}]$		
A	B	T	B	A	T	$A_{1,0}$	$B_{0,1}$	T	$A_{0,1}$	$B_{1,0}$	T	$A_{1,0}$	$B_{1,0}$	T
a_0	b_0	t_0	b_0	a_0	t_0	a_1	b_0	t_2	a_0	b_1	t_1	a_1	b_1	t_3
a_0	b_1	t_1	b_0	a_1	t_2	a_1	b_1	t_3	a_0	b_0	t_0	a_1	b_0	t_2
a_1	b_0	t_2	b_1	a_0	t_1	a_0	b_0	t_0	a_1	b_1	t_3	a_0	b_1	t_1
a_1	b_1	t_3	b_1	a_1	t_3	a_0	b_1	t_1	a_1	b_0	t_2	a_0	b_0	t_0

Se trata de la misma información de la tabla original pero con distintas configuraciones asociadas a las *Bases*, que a su vez determinan las secuencias de respuestas adyancentes correspondientes. □

Desde el punto de vista geométrico, el $XBase$, en términos del orden de los atributos, es la aplicación de simetrías y rotaciones a los puntos o celdas de información de la tabla, en el sistema de referencia, que conserva las distancias entre celdas, $distancia(celda_i, celda_j)$, siempre que la distancia considerada sea una función que agrega la diferencia entre las coordenadas homólogas. En principio, la distancia de Hamming (Hamming, 1950) es la más apropiada, dado que ya hemos comentado que el índice es un símbolo y no es necesariamente cuantificable, no tiene norma o módulo como un vector. Podemos utilizar la distancia de Hamming entre dos celdas para establecer una relación de adyacencia multidimensional. Pero es muy complicado y de dudosa utilidad práctica. Sin embargo, en la proyección unidimensional que proporciona el desplazamiento, la relación de adyacencia es única: dos celdas de la MM son adyacentes si sus respectivos desplazamiento son consecutivos. La adyacencia depende de la *Base*, véase la Figura 2.1. Veremos que ha resultado de cierta utilidad para construir y explotar las BC. Otras distancias posibles son: L1 (Manhattan), L2 (euclídea usual), Max,...

El $XBase$ no afecta a la semántica de la información recogida en la tabla. La información y el conocimiento, supuestamente recogidos en la tabla, constituyen los invariantes frente a $XBase$ en los que se centra nuestro interés. Si existe un $XBase$ que transforma una matriz en otra, ambas matrices son equivalentes. La semántica de los atributos permite establecer la

equivalencia de manera directa. Pero responder a la pregunta de si dos MM son equivalentes es un problema muy complejo como veremos en el capítulo 3. Sintácticamante se trata de conocer los vectores que permutan las componentes homólogas entre *Bases*. Así se almacena y recupera la misma información, reordenando el índice de acceso conforme a la *Base* en la cual se expresa la tabla, en modo matricial, respecto de cierta *Base* de referencia. Esto se verá en detalle en la sección 3.2.

Para establecer correspondencias entre órdenes diferentes, identificamos el $XBase$ con un vector de longitud igual al número de dimensiones de la MM y cuyas componentes son los números ordinales de los atributos de la *Base*. En el caso del orden inicial o canónico, a la *Base* $[0, 1, 2, \ldots, \delta - 1]$ la llamaremos identidad. Cualquier permutación de la *Base* identidad genera una base de atributos de la matriz. La *Base* identidad determina un orden convencional en los atributos del índice de la *Base* canónica. Véase un ejemplo gráfico en la Figura 2.2, donde I es un índice en la *Base* inicial, J está en la *Base* canónica, K está en la *Base* nueva, índice es la referencia a la componente del índice y la reasignación de valores del índice es:

1. $J[i] \leftarrow I[B1[i]]$
2. $K[i] \leftarrow J[B2[i]]$

Hemos tomado del léxico del lenguaje Java, R o C la notación usual del acceso al valor de una componente índice de un vector.

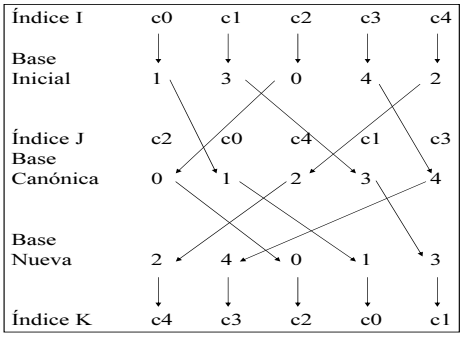

Figura 2.2: $XBase$

En resumen, para considerar la equivalencia entre tablas en distinta *Base* se debe conocer la permutación entre sus atrbutos que proporciona un orden a partir de otro y observar que los contenidos son idénticos, pero con diferente disposición relativa de las filas.

Podemos pensar que la *Base* no afecta, en principio, al espacio o número de celdas que

necesita la estructura de datos de la tabla. Tampoco revela nada nuevo relativo a la organización de la información.

Sin embargo, desmentiremos las dos afirmaciones del párrafo anterior. Estudiaremos que la *Base* determina el orden y el espacio de almacenamiento lógico y la recuperación de la información y está relacionada estrechamente con el conocimiento que se representa en las tablas y la utilización eficiente de la información. El conocimiento se manifiesta en dos facetas interrelacionadas: contenidos (respuestas) y organización (dependencias entre atributos).

Mediante ejemplos, pongamos en juego los conceptos anteriores y mostremos los aspectos importantes.

Ejemplo 2.5. A continuación mostramos un esquema de dos dimensiones.

1. Sea la matriz siguiente (utilizada en el ejemplo 2.1) cuyo esquema es $\{A, B\}$. Sean i_0 y i_1 los componenetes del índice de los atributos en la $Base$ $[A, B]$. El peso de i_0 es $W_0 = \delta_1 = 5$ y el peso de i_1 es $W_1 = 1$.

$[A, B]$	0	1	2	3	4
0	0	0	0	0	1
1	1	1	1	1	1
2	1	1	1	1	1
3	0	0	1	2	1

   La función desplazamiento, con la $Base$ $[A, B]$, es $f_{[A,B]}(i_0, i_1) = i_0 * \delta_1 + i_1 = i_0 * 5 + i_1$.

2. Sean i_0 y i_1 los componentes del índice de los atributos en la $Base$ $[B, A]$. El peso de i_0 es $W_0 = \delta_1 = 4$ y el peso de i_1 es $W_1 = 1$. La función desplazamiento, con la $Base$ $[B, A]$, es $f_{[B,A]}(i_0, i_1) = i_0 * \delta_1 + i_1 = i_0 * 4 + i_1$.

$[B, A]$	0	1	2	3
0	0	1	1	0
1	0	1	1	0
2	0	1	1	1
3	0	1	1	2
4	1	1	1	1

La matriz se almacena linealmente según determina $f(i_0, i_1)$ que depende de la $Base$ (que determina los W_i) y del cardinal del dominio de los atributos – índice de la siguiente forma:

$[A, B]$	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	0	0	1	2	1

donde $\delta_0 = 4$ y $\delta_1 = 5$, en la $Base$ $[A, B]$ y

$[B, A]$	0	1	1	0	0	1	1	0	0	1	1	1	0	1	1	2	1	1	1	1

donde $\delta_0 = 5$ y $\delta_1 = 4$, en la $Base$ $[B, A]$.

A continuación mostramos el desplazamiento de las celdas de memoria de ambas matrices, véase el Cuadro 2.2. El orden de desarrollo de los índices es $for(\overrightarrow{i}) = \{for(i_0)\{for(i_1)\}\}$.

Ambas $Bases$ dan idéntico contenido a las tablas. La primera que corresponde con $[A, B]$, está en un orden natural (creciente en las posiciones de memoria) de los elementos, por ello la llamamos canónica. La segunda corresponde con $[B, A]$ y es transposición de la $Base$ canónica. Si disponemos la tabla conforme a la cabecera:

(i_1, i_0)	$f_{[B,A]}(i_1, i_0)$	respuesta

la $Base$ $[B, A]$ aparecería en el orden natural, sería la $Base$ canónica, y su transpuesta no. Vemos que, de hecho, llamar canónica a una $Base$ es arbitrario.

Podemos escribirlo también de modo resumido, con el formato $\langle(f_{[A,B]}(i_0, i_1), respuesta)$:
$\langle(0,0)$ $\langle(1,0)$ $\langle(2,0)$ $\langle(3,0)$ $\langle(4,1)$ $\langle(5,1)$ $\langle(6,1)$ $\langle(7,1)$ $\langle(8,1)$ $\langle(9,1)$ $\langle(10,1)$ $\langle(11,1)$ $\langle(12,1)$ $\langle(13,1)$
$\langle(14,1)$ $\langle(15,0)$ $\langle(16,0)$ $\langle(17,1)$ $\langle(18,2)$ $\langle(19,1)$
donde se puede observar el orden ($\langle$) del contenido de la tabla bidimensional en la memoria lineal. □

Ejemplo 2.6. A continuación mostramos un esquema de tres dimensiones.

Sea una tabla cuyo esquema es {A,B,C} y la matriz $MM[i_0, i_1, i_2]$ de 3 dimensiones tal que los dominios son $\{a_0, a_1\}$, $\{b_0, b_1, b_2\}$ y $\{c_0, c_1, c_2, c_3\}$ respectivamente. La $Base$ es $[A, B, C]$ donde las componentes son los índices de acceso a la matriz y representan el valor que toman ciertos atributos o factores. Los pesos son: $W_0 = 3 * 4 = 12$, $W_1 = 4$ y $W_2 = 1$.

El elemento $v_{0,1}, v_{1,1}, v_{2,1}$ se almacena en $q = 1 * 12 + 1 * 4 + 1 = 17$, posición del vector lineal. Esta posición es el desplazamiento respecto a la posición del primer elemento del vector de celdas de memoria, $v_{0,0}, v_{1,0}, v_{2,0}$ cuyo $q = 0$. Las $Bases$ posibles son $3! = 6$:

A,B,C	A,C,B	B,A,C	B,C,A	C,A,B	C,B,A

Análogamente al caso anterior se producen diferentes asignaciones de memoria en las distintas $Bases$. □

En resumen, se ha tratado de dar algunas ideas sobre cómo se puede almacenar cierta información o conocimiento tabulado y acceder al mismo. Lo hemos planteado en términos de la

Cuadro 2.2: Desplazamiento de las celdas de memoria

(i_0, i_1)	$f_{[A,B]}(i_0, i_1)$	respuesta	$f_{[B,A]}(i_0, i_1)$	respuesta
(0,0)	0*5+0=0	0	0*4+0=0	0
(0,1)	0*5+1=1	0	1*4+0=4	1
(0,2)	0*5+2=2	0	2*4+0=8	1
(0,3)	0*5+3=3	0	3*4+0=12	0
(0,4)	0*5+4=4	1	4*4+0=16	0
(1,0)	1*5+0=5	1	0*4+1=1	1
(1,1)	1*5+1=6	1	1*4+1=5	1
(1,2)	1*5+2=7	1	2*4+1=9	0
(1,3)	1*5+3=8	1	3*4+1=13	0
(1,4)	1*5+4=9	1	4*4+1=17	1
(2,0)	2*5+0=10	1	0*4+2=2	1
(2,1)	2*5+1=11	1	1*4+2=7	1
(2,2)	2*5+2=12	1	2*4+2=10	0
(2,3)	2*5+3=13	1	3*4+2=14	1
(2,4)	2*5+4=14	1	4*4+2=18	1
(3,0)	3*5+0=15	0	0*4+3=3	2
(3,1)	3*5+1=16	0	1*4+3=7	1
(3,2)	3*5+2=17	1	2*4+3=11	1
(3,3)	3*5+3=18	2	3*4+3=15	1
(3,4)	3*5+4=19	1	4*4+3=19	1

expresión de las tablas como matrices en cierto esquema y *Base*. Los $XBase$ con permutaciones en el orden de atributos del esquema (y otros tipos de $XBase$ no tratados aquí mediante descomposición o composición del dominio de los atributos iniciales y permutación de dominios) mantienen invariante la información pero implican diferentes organizaciones y complejidad de los elementos en la memoria.

A continuación desarrollamos formalmente una estructura que nos permitirá conseguir los mismos objetivos con mayor generalidad y eficiencia utilizando $XBase$.

2.3. *KBM2L*, una Representación Matricial Implícita de la Tabla

Hemos visto que una tabla es un vector de datos en la memoria de un ordenador convencional. Cada dato o respuesta tiene una posición o desplazamiento asociado al índice que tiene en la tabla.

Como ya indicamos en la sección 2.1, en sistemas complejos, estas estructuras de datos como tablas son enormes, pues se trata de matrices (arrays) multidimensionales de orden el producto de los dominios de los atributos. Por ejemplo, una tabla que tiene dos atributos A y B con dominios respectivos de 4 y 5 valores tiene $|A| * |B| = 4 * 5 = 20$ elementos. Una tabla con 40 atributos binarios tiene $2^{40} \approx 10^{12}$ elementos.

Podemos plantearnos sistemas donde el esquema tenga decenas de atributos, δ, con dominios de valores muy grandes $(2, 4, 8, 16, 32, 64, \ldots, 1024, \ldots)$. El espacio de almacenamiento es $(\ldots,$ $(2^8)^\delta$, $\ldots$, $(2^{10})^\delta$, $\ldots,)$ infinito a todos los efectos. En inteligencia artificial nos encontramos habitualmente con problemas de representación y búsqueda cuya complejidad es exponencial debido a la combinatoria, por ejemplo, en los árboles de decisión. Hacemos notar que las aplicaciones no se circunscriben al análisis de decisiones y SAD, aunque es en este área donde se han desarrollado los proyectos relacionados con esta tesis. Ésta es la razón de situar la exposición en el marco del análisis de decisiones y proponer ejemplos relativos a TDO y TPC.

El almacenamiento de la información tabulada en el sistema de ficheros de un ordenador, su acceso y manipulación son problemas que planteamos abordar mediante una estructura de datos para la matriz que representa a la tabla. La estructura será una lista construida a partir del almacenamiento matricial de la información tabulada, que llamaremos *KBM2L* (véase la próxima subsección). Su nombre nos indica que la BC multidimensional se transforma en una lista (en inglés "Knowledge Base Matrix To List").

Las dos hipótesis relativas al conocimiento tabulado que permiten explotar la estructura en las tablas se concretan en:

1. el dominio de los valores almacenados toma un número discreto y reducido de modalidades y
2. el conocimiento presenta patrones de regularidad, sobre el espacio de representación de la tabla.

Bajo las anteriores hipótesis que llamaremos, respectivamente, **representatividad discreta** y **granularidad de la información**, relativas al conocimiento en las áreas concretas a las que hace referencia la tabla, ésta se puede tratar como una matriz muy compacta o resumida.

Estas hipótesis se pueden asumir razonablemente con mucha frecuencia en análisis de decisiones. Muchos problemas de decisión se pueden abordar mediante variables discretas que permiten representaciones tabuladas– representatividad discreta– del conocimiento. Por otra parte, en los procesos de decisión, la información está condensada en escenarios o contextos (granos), es decir, si se toma una decisión en un caso, en casos similares se toma la misma o parecidas decisiones –granularidad de la información–.

En general, las preferencias, la estructura de relaciones representada en los diagramas de influencia y la evaluación del modelo conducen a patrones de regularidad y granularidad sobre el espacio de estados del problema. Ésta es una observación empírica, pero tiene cierta justificación en las estructuras de datos que soportan la representación del conocimiento (tablas multidimensionales), en el algoritmo de evaluación (maximización de la utilidad esperada) (Shachter, 1986) o en la independencia del contexto (Boutilier et al., 1996; Cano et al., 2000; Poole y Zhang, 2003; Heckerman, 1990), entre otras razones. Posiblemente la razón general sea el principio de Pareto (formulado en 1895 por Vilfredo Pareto, un economista italiano) que es de naturaleza empírica, y sostiene que el conjunto de variables influyentes en un cierto problema casi siempre puede restringirse al 20 % y sin embargo explican más del 80 % del resultado o comportamiento observado. En el análisis de decisiones, además, la asimetría inherente a los problemas (Bielza y Shenoy, 1999; Shenoy, 2000) introduce restricciones que implican granularidad en las TDO . La granularidad está relacionada con el proceso de modelización y en concreto con la estructuración de los problemas y la modularidad de las distintas componentes de un modelo de representación del conocimiento. Por ejemplo en los MGP (GeNIe, 2003), el aprendizaje automático (Quinlan, 1986; Mitchell, 1997), o en la ingeniería del software (Pressman, 1993), por citar otras áreas.

La granularidad del contenido de la tabla depende del problema y de la organización del contenido de la tabla que determina su *Base*. La representatividad discreta del conocimiento es un requisito para poder observar cierta granularidad y posibilidad de optimización en el almacenamiento. Si algún atributo o la información es intrínsecamente continua, o tiene un dominio muy grande, es difícil que se puedan encontrar granos de información. Si no se presenta granularidad y/o la respuesta es aleatoria o con mucho ruido puede existir conocimiento en la tabla, pero la granularidad no se revela útil para extraerlo. Para representar el conocimiento asociado a dichas fuentes de información tal vez sea más adecuada una ley de probabilidad. Por otra parte, si el conocimiento se ajusta bien a una función, ecuación, o a una formulación lógica simple, ellas mismas dan lugar a una representación muy eficiente aunque no se presente apenas granularidad.

Si el contenido de la tabla presenta cierta granularidad podemos almacenar un represen-

tante de cada grano. El **ítem** representa al conjunto de casos adyacentes en la métrica del desplazamiento (véase el comentario en la sección 2.2 acerca del punto de vista geométrico del $XBase$) que tienen la misma respuesta. El concepto de ítem presenta cierta analogía con el concepto de aproximación inferior de los conjuntos aproximados, conocidos como *Rough Sets* (Pawlak, 1997; Pawlak, 2002).

Proponemos una técnica que usa las representaciones matriciales de las tablas para identificar los granos. En dicha estructura de datos planteamos reagrupar los casos que constituyen contextos o granos. Intuitivamente se trata de grandes conjuntos de respuestas o de datos iguales. Recordemos que diferentes *Bases* constituyen tablas equivalentes desde el punto de vista de su contenido semántico, pero pueden tener muy diferente uso de memoria, si es posible reconocer unos pocos granos que representen a las entradas o casos de la tabla. La construcción de los granos no es trivial, y necesita de heurísticas, técnicas de optimización estocástica y una considerable potencia computacional.

La menor y más óptima de todas las estructuras tiene propiedades muy interesantes computacionalmente, por una parte, y desde el punto de vista de la ingeniería del conocimiento, por otra. Si esto es así a menudo, la estructura es muy eficiente como soporte del conocimiento que infieren los sistemas. Reordenar la información repercute en el almacenamiento y acceso a la misma. Además, ciertas organizaciones de los datos proporcionan conocimiento relativo al esquema de la tabla, que ya hemos comentado (dependencias entre grupos de atributos, afinidad entre atributos, relevancia y explicación del contenido de la tabla mediante subconjuntos del esquema,...). En definitiva, disponemos de una herramienta de descripción y análisis multivariante de la tabla multidimensional.

Nos preguntamos qué grado de coalescencia (versus dispersión) tendrá una lista *KBM2L*, es decir, qué expectativas tenemos al tratar una tabla con la técnica propuesta. El grado de coalescencia de la lista debe ser de orden polinómico respecto del logaritmo de desplazamiento máximo (el tamaño de la tabla) y proponemos dos argumentos que sostienen esta afirmación. El primero es de carácter práctico. La MM es una estructura de complejidad exponencial y la lista asociada sólo es factible construirla si tiene un tamaño a la medida de los recursos (computacionales). En otro caso no se observa coalescencia y es imposible resumir la MM. Por otra parte, argumentamos sobre la propia naturaleza de la información resumida en la lista en forma de ítems: es conocimiento. Aquí el patrón de medida del tamaño es la capacidad humana de conocer y transmitir el conocimiento. Si una BC tiene tamaño exponencial, aunque teóricamente pueda ser computable, supera la capacidad humana (individual y colectiva) y no podríamos, por ejemplo, validarla ni alterarla.

El mayor nivel de coalescencia puede que no se obtenga en la dimensión lineal de la lista y debamos considerar otras alternativas de representación, véase el capítulo 7.

A continuación, mostraremos el modo de compactar la información y veremos cómo se pueden explotar eficientemente las estructuras *KBM2L* propuestas con diversos propósitos, centrándonos en la extracción de reglas, es decir, generalizaciones de las proposiciones de la tabla.

2.3.1. Lista *KBM2L*

Los ítems representan granos de conocimiento o conjuntos de casos con la misma respuesta. Si el contenido de la tabla presenta cierto nivel de granularidad podemos almacenar un solo valor en lugar de un conjunto de casos. Elegimos como representante al caso de mayor desplazamiento en la secuencia de iguales. El desplazamiento depende de la *Base* y por tanto el representante es, en general, otro si ésta cambia.

Hasta aquí, podemos ver que potencialmente podemos comprimir en cierta medida la tabla. De hecho, este procedimento es similar al utilizado para la compresión de las imágenes en el formato GIF (Graphics Interchange Format, ©CompuServe Incorporated) y el formato TIFF (Tag Image File Format, Adobe Software), véase (Foley et al., 1993). El algoritmo de compresión utilizado es el LZW (Ziv y Lempel, 1977) revisado, propiedad de Unisys. Se trata de una compresión poco eficiente para las matrices de píxels de las imágenes, pero sencillo de implementar, de bajo coste y sin pérdida. Es muy usado en las páginas html del WWW en internet para gráficos. En compresión de datos, imágenes por ejemplo, hay procedimientos mucho más eficentes, pero sólo en cuanto al ratio compresión y velocidad de compresión/expansión, tienen un componente estadístico y los hay con factores de pérdida (jpeg (Joint Photographic Experts Group), mp3 (MPEG-1 Audio Layer-3), mpeg (Moving Picture Experts Group)). Los algoritmos de compresión de datos, que se fundamentan en la *Teoría de la Información* (Ziv y Lempel, 1977): los conocidos ARJ^{TM} (©1999–2003, ARJ Software Inc.), gzip (GNU zip), RAR (©2002-2003, Eugene Roshal) y otros relativos a grandes bases de datos (Apte, 2003; Ooi et al., 2000; Wang et al., 2001), no tienen en cuenta la semántica de los datos; en ningún caso consideran el conocimiento implícito en los datos. Esta última consideración se ha mencionado al comienzo del capítulo.

Recuperando el hilo de la sección, cabe decir que si consideramos otra *Base* tenemos el mismo conocimiento pero cambia la granularidad, y por tanto, los requerimientos de memoria de la lista final de ítems. El objetivo es encontrar la *Base* que minimiza el número de ítems y manifiesta los granos de conocimiento de mayor calidad en el sentido de agrupar el mayor número de los casos afines, es decir, con la misma respuesta. Esto, a diferencia de los algoritmos

de compresión, ofrece un medio de explicación, y muestra relaciones entre grupos de atributos y respuestas de la tabla.

Como ejemplo, sea el contenido de una tabla representada y almacenada como la siguiente MM:

$$\langle(0,\sharp)\ \langle(1,\sharp)\ \langle\ldots<(p-1,\bar{x})\ \langle(p,x)\ \langle\ldots\langle(q,x)\ \langle(q+1,\bar{x})\ \langle\ldots\langle(W_0*\delta_0-1,\sharp)$$

donde cada celda de la tabla es representada con el par $(desplazamiento, respuesta)$, W_0 es el peso del primer atributo, δ_0 es el cardinal de su dominio, el signo $\langle$ muestra el orden entre posiciones de memoria usado para almacenar los pares, igual que el ejemplo 2.4 de la sección 2.2, y $\sharp$ denota una modalidad cualquiera de la respuesta. Ahora supongamos que las celdas entre las posiciones p y q, donde $p < q$, contienen la misma respuesta x.

La lista **KBM2L** para esta tabla incluye todas las posiciones entre p y q en un solo ítem, ahorrando espacio de memoria. Introducimos el símbolo | para denotar el final de la secuencia de casos de un ítem. Sin tener en cuenta otros posibles granos de conocimiento, la lista será:

$$\ldots\langle p-1,\bar{x}|\langle q,x|\ldots\langle W_0*\delta_0-1,\sharp|$$

Así, la notación $\langle$*desplazamiento, respuesta*| que usamos refleja dos ideas. Primeramente, el desplazamiento de los ítems es estrictamente creciente, y en segundo lugar, resume un conjunto de celdas adyacentes con el mismo valor de respuesta y diferente valor de respuesta del ítem siguiente y anterior, si existen. La lista se puede expresar de forma equivalente mediante índices a la que denominaremos como notación de índices, en lugar de mediante desplazamientos, es decir:

$$\langle q,x|\equiv\langle\overrightarrow{c},x|$$

Axiomas *KBM2L*. Las condiciones formales de consistencia de la lista *KBM2L*, donde p_j es el desplazamiento del ítem j, y m_j es la respuesta de sus casos, son:

1. Todos los desplazamientos son no negativos y no pueden superar el tamaño de la tabla asociada $W_0 * \delta_0 - 1$, el cual determina el desplazamiento del último ítem de la lista en cualquier *Base*. $0 \leq p_j \leq (W_0 * \delta_0 - 1)$, $\forall$ ítem j en la lista *KBM2L*.

2. Los desplazamientos son estrictamente crecientes: $p_i < p_{i+1}$, $\forall i$.

3. La información o respuesta de contextos adyacentes es diferente: $m_i \neq m_{i+1}\,\forall i$. Si las respuestas son idénticas, los casos serán unidos en un único ítem por definición de lista *KBM2L*.

Ejemplo 2.7. (Cont. del ejemplo 2.5) Sea la siguiente matriz en la *Base* $[A, B]$:

$[A, B]$	0	1	2	3	4
0	0	0	0	0	1
1	1	1	1	1	1
2	1	1	1	1	1
3	0	0	1	2	1

Su almacenamiento lineal se puede representar mediante el desplazamiento:

$$\begin{array}{lllll} \langle(0,0) & \langle(1,0) & \langle(2,0) & \langle(3,0) & \langle(4,1) \\ \langle(5,1) & \langle(6,1) & \langle(7,1) & \langle(8,1) & \langle(9,1) \\ \langle(10,1) & \langle(11,1) & \langle(12,1) & \langle(13,1) & \langle(14,1) \\ \langle(15,0) & \langle(16,0) & \langle(17,1) & \langle(18,2) & \langle(19,1) \end{array}$$

o índice (i_0, i_1):

$$\begin{array}{llll} \langle((0,0),0) & \langle((0,1),0) & \langle((0,2),0) & \langle((0,3),0) \\ \langle((0,4),1) & \langle((1,0),1) & \langle((1,1),1) & \langle((1,2),1) \\ \langle((1,3),1) & \langle((1,4),1) & \langle((2,0),1) & \langle((2,1),1) \\ \langle((2,2),1) & \langle((2,3),1) & \langle((2,4),1) & \langle((3,0),0) \\ \langle((3,1),0) & \langle((3,2),1) & \langle((3,3),2) & \langle((3,4),1) \end{array}$$

La lista *KBM2L* que construimos tiene 6 ítems:

$$\langle 3,0| \quad \langle 14,1| \quad \langle 16,0| \quad \langle 17,1| \quad \langle 18,2| \quad \langle 19,1|$$

en notación de desplazamiento, o

$$\begin{array}{lll} \langle(0,3),0| & \langle(2,4),1| & \langle(3,1),0| \\ \langle(3,2),1| & \langle(3,3),2| & \langle(3,4),1| \end{array}$$

en notación de índices. Podría comprobarse que la otra *Base*, $[B, A]$, presenta un lista con 4 ítems más. □

Observamos varios hechos:

1. posiblemente varias *Bases* distintas determinen la misma lista;
2. seguramente varias *Bases* distintas determinen listas distintas pero con la misma longitud o número de ítems;
3. por último, el número de ítems puede hacerse menor si consideramos cambios de sistema de referencia más generales que descompongan el esquema de los atributos en sus valores, como ya indicamos en la subsección 2.2.2.

2.3.2. Construcción de la Lista *KBM2L*

El procedimiento para construir una lista *KBM2L*, véase la Figura 2.3, consiste en los pasos siguientes:

1. Obtenemos la tabla. En general, la tabla completa puede no estar disponible. Tal vez, su contenido esté distribuido en diferentes localizaciones o haya que evaluarla por partes secuencialmente o en paralelo. Puede consistir en un conjunto de subtablas o subproblemas que llamaremos **instancias del modelo**. La idea es que una tabla puede ser muy compleja, por ello es que la tratamos de resumir y explicar.

2. Las tablas de cada subproblema son almacenadas en ficheros. Todas comparten el mismo esquema, al ser fragmentos de la tabla completa. La información de una de las instancias es leida y trasladada a la lista *KBM2L*. La organización de esta primera tabla, el orden de los atributos, determina la *Base* inicial, canónica. Inicialmente esta lista está vacía, con un solo ítem que representa la ausencia de conocimento (el valor de respuesta que hemos definido como *unKB*). Así, el estado inicial de la lista *KBM2L* con un solo ítem es $\langle W_0 * \delta_0 - 1, unKB|$, en la notación desplazamiento y $\langle(\delta_0 - 1, \delta_1 - 1, \ldots, \delta_{\delta-1} - 1), unKB|$ en la notación de índice.

3. El resto de las instancias del problema son leidas y trasladadas a la estructura inicializada en el paso 2, a través de un mecanismo de aprendizaje. El aprendizaje se realiza en dos planos. El primero es mediante el incremento de la masa de datos de la lista. Esto se detallará en la sección 2.6. El segundo es mediante la búsqueda simultánea de una *Base* óptima, que permita almacenar los datos en el menor número de ítems posible, que se verá en el capítulo 3. Cuanto más óptima sea la *Base*, más fácil es la construcción de la lista.

4. Finalmente optimizamos la lista. Esto permite la explotación de la BC y proporciona explicaciones de la respuesta de la tabla e interrelaciones entre los atributos. Ambos aspectos los desarrollaremos en detalle en la sección 2.4 y mostraremos ejemplos en los capítulos 4 y 5.

La instanciación de las tablas (paso 1) es una consecuencia natural de la complejidad de los problemas. Así por ejemplo en un DI, por motivos de espacio o tiempo, puede ser necesario evaluar un subproblema (Ezawa, 1998) mediante una instancia parcial de algunos atributos involucrados en las TDO. Por tanto, la estrategia que acabamos de proponer es la construcción

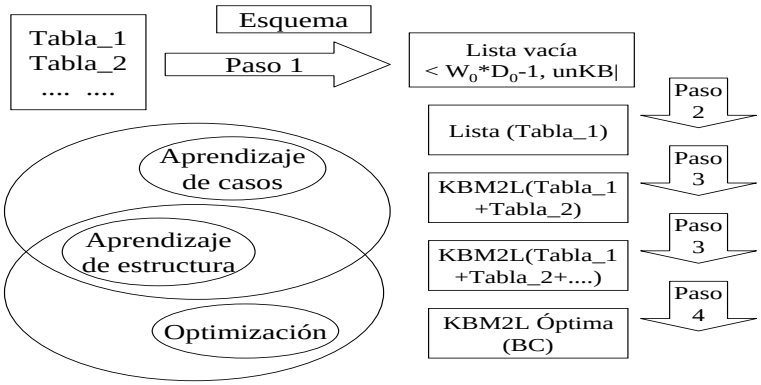

Figura 2.3: Construcción de la lista *KBM2L*

incremental de las BC, mediante una lista *KBM2L*, que recoge todas las tablas parciales tan pronto como estén disponibles.

En la práctica los cuatro pasos están estrechamente conectados en la implementación del sistema y cooperan en el objetivo principal: obtener una BC eficiente para el SAD. Los cuatro pasos son en realidad cuatro tareas que pueden realizarse de forma concurrente (Arnold et al., 1994) y distribuida (Tanembaum, 1990), lo cual dota al sistema de una potencia y escalabilidad muy grande. Pero más aún, las estructuras matriciales subyacentes permiten un grano de paralelismo (Hwang y Briggs, 1985) fino en la gestión de la lista *KBM2L*, véase la Figura 2.2. Tal como muestran las flechas, el movimiento de transposición de los valores del índice puede ser concurrente. Además, la tabla se puede procesar simultáneamente en distintas partes y finalmente construir la lista completa a partir de las listas parciales. Esta característica permite la distribución del SAD mencionada en la sección 1.4 y se podrá observar en detalle al describir la gestión de la lista en la sección 2.6.

2.4. Explicación del Contenido de una Tabla Mediante Listas *KBM2L*

Desarrollamos ahora para las listas la definición de índice propuesta en la sección 2.1 para las tablas. El concepto de índice es esencial en el procesamiento de la información de las tablas. El índice permite identificar la información y su localización en las estructuras (tablas y listas). Los índices de las listas permiten, por consiguiente, realizar las operaciones usuales sobre los datos: consulta y actualización. Además, lo interesante será que, en las listas, el índice permite presentar, en las *Bases* óptimas, explicaciones de las respuestas y caracterizar la importancia

relativa de los atributos.

2.4.1. Índice de Ítems para Obtener Explicaciones

Definimos la **familia** de ítems en una lista: todos los ítems que comparten la misma respuesta. La idea de familia es interesante desde dos puntos de vista. El primero está relacionado con la optimización. Ésta puede verse como un mecanismo de aprendizaje automático de la *Base* óptima, que desarrollaremos en el capítulo 3. En general, es más fácil aprender sobre cada familia por separado y luego generar la lista completa, con todas las familias. Como un estudiante que aprende clases o tipos de problemas en una asignatura, por separado, y finalmente adquiere una comprensión de carácter global acerca de la materia. El segundo está relacionado con la explicación que desarrollaremos en esta sección. Ésta puede ser más sencilla dentro de la familia que en el contexto de la lista completa. Es posible que haya interferencias entre las explicaciones similares de las familias de ítems, es decir, la lista puede configurarse para mostrar buenas explicaciones de unas familias pero otras no se explican con similar sencillez. Así, dependiendo del objeto de la explicación, en principio es mejor centrarse en la respuesta de una familia para obtener una explicación más clara y sencilla.

Denotamos mediante I_{inf} e I_{sup} a los índices correspondientes a los casos primero y último de un ítem. Definimos sobre estos vectores índice asociados a los desplazamientos extremos de un ítem dos partes o grupos de atributos.

La primera es la **parte fija** del índice, los índices que están evaluados en valores concretos y que son comunes a todos los casos del ítem. Es el resultado del operador AND-lógico ($I_{inf} \wedge I_{sup}$). Así, un atributo A_k está en la parte fija si coinciden los valores c_k de I_{inf} y I_{sup}. Además, por la forma de enumerar los casos, coincidirán también los valores c_i de las componentes tales que $i \leq k$. Para establecer la parte fija solamente hay que mirar los índices extremos porque todos los casos de un ítem (de toda la lista) están ordenados. La segunda parte, y complementaria a la primera, es la **parte variable**.

Por otro lado, observamos que el primero (más a la izquierda en el esquema) de los atributos de la parte variable (más a la derecha en el esquema) puede tomar, entre los casos del ítem, todos y cada uno de los valores de su dominio. Pero puede ser que tan sólo tome algunos valores, no siempre el mismo, y algunos no se presenten. Esta consideración no se plantea si el dominio del atributo es binario, en cuyo caso si fuese un atributo de la parte variable toma o recorre necesariamente todo su domino. Tampoco ocurre con el segundo y sucesivos atributos de la parte variable, que toman todos y cada uno de los valores de sus respectivos dominios. Si un ítem tiene un atributo como el descrito, que llamamos **atributo de dominio restringido** (ADR), veremos

que la explicación es más compleja. Si el dominio es binario para todos los atributos, entonces no encontramos ningún ADR en las explicaciones.

Si en la familia encontramos algunos atributos que toman los mismos valores podemos simplificar la explicación estableciendo el contexto correspondiente con la intersección de las partes constante en la familia. Esta simplificación permite explicar una respuesta con el menor número de atributos. En la *Base* óptima encontramos los ítems que generalizan más eficientemente el contenido de las tablas agrupando conjuntamente todos los casos de acuerdo a los valores de la política óptima o respuesta adyacentes. La explicación es mínima si los ítems son de tamaño máximo, véase la sección 2.4.2.

Ejemplo 2.8. (Cont. del ejemplo 2.7) Como una ilustración de las partes del índice, sea el ítem $\langle 3, 0|$. Los índices extremos son $I_{inf} = (i_0 = 0, i_1 = 0)$ e $I_{sup} = (i_0 = 0, i_1 = 3)$. Entonces, el ítem contiene 4 casos, $\{i_0\}$ es la parte fija del índice e $\{i_1\}$ es la parte variable. $i_0 = 0$ explica en cierta medida el hecho de que la respuesta sea 0. Si tomamos el ítem $\langle 19, 1|$, entonces $I_{inf} = (i_0 = 3, i_1 = 4) = I_{sup}$. Así, éste es un ítem con un único caso y tiene todo el índice fijo.

El ítem $\langle 16, 0|$ que representa los casos $((3, 0), 0)$ y $((3, 1), 0)$ tiene el atributo A fijo con valor 3 y el B es un ADR con dominio restringido a los valores 0 y 1. □

Consecuencia de las definiciones anteriores es que estos conjuntos de atributos proporcionan una forma de construcción de explicaciones de las respuestas de los casos circunscritos al ítem:

C1 El análisis de los índices extremos y las partes fija y variable y los ADR permite interpretar la parte fija, con valores restringidos, como la explicación de la respuesta común en el ítem.

C2 Las partes fijas de cada famila constituyen la explicación de la correspondiente respuesta en la BC. El resto de atributos que desarrollan todo el dominio no es información relevante.

C3 Un ítem de un solo caso tiene todo su índice fijo e instanciado en las coordenadas del caso en el sistema de referencia. Los índices I_{inf} e I_{sup} son iguales. La parte fija es todo el índice. La parte variable es vacía.

C4 Un ítem que recoge casi todos o la mayoría de los casos posibles de una lista puede tener parte fija vacía. Por ejemplo, puede ocurrir que ningún atributo sea relevante en la respuesta. La descripción del ítem es más complicada y precisa del análisis del dominio del atributo de mayor peso, i_0, que es un ADR.

C5 Cuando no hay parte fija ni ADR consideramos que no hay explicación y por tanto podemos convenir que el ítem no tiene conocimiento.

Supongamos que no hay ADR, dada la respuesta r en la lista *KBM2L* y las explicaciones de r en los h diferentes ítems de la familia, $r \rightarrow e_0, e_1, \ldots, e_h$. Entonces, la explicación de la tabla es la disyunción de las conjunciones de atributos de la parte fija correspondientes a cada ítem. Las explicaciones e_j de cada ítem con $p+1$ atributos en la parte fija son de la forma:

$$r_k \rightarrow (A_0 = a_{0,i}, A_1 = a_{1,j}, \ldots, A_p = a_{p,m})$$

donde r_k es la modalidad de la respuesta del ítem y los valores de los atributos se denotan con dos subíndices. El primero recorre la *Base* de atributos y el segundo especifica el valor.

La parte fija y el ADR del índice de los ítems es la materia prima de las explicaciones. Así, en un ítem genérico, la respuesta es r y los atributos de la parte fija son $A_0, A_1, \ldots A_p$ y A_q, $p+1 = q$, es el ADR. La explicación es: $r \rightarrow (A_0 = a_{0,i}, A_1 = a_{1,j}, \ldots, A_p = a_{p,m}, A_q = a_{q,x})$ $\forall a_{q,x} \leq a_{q,y}$, donde $a_{q,y}$ es el valor del atributo A_q en el índice I_{sup}.

Unos ejemplos muy sencillos con tablas de operadores de la lógica proposicional nos ilustran las consecuencias mencionadas.

Ejemplo 2.9. (Cont. del ejemplo 2.8) Sea la lista:

$$\langle 3,0|\ \langle 14,1|\ \langle 16,-1|\ \langle 17,1|\ \langle 18,2|\ \langle 19,-1|, \text{ con 6 ítems.}$$

Las explicaciones siguientes se construyen mediante la parte fija y los ADR de los ítems.

ítem		I_{inf}	I_{sup}	Explicaciones
0	$\langle 3,0\|$	$(0,0)$	$(0,3)$	$0 \rightarrow i_0 = 0, i_1 \leq 3$
1	$\langle 14,1\|$	$(0,4)$	$(2,4)$	$\emptyset$,$i_0 \leq 2$
2	$\langle 16,-1\|$	$(3,0)$	$(3,1)$	$\emptyset$,
3	$\langle 17,1\|$	$(3,2)$	$(3,2)$	$1 \rightarrow i_0 = 3, i_1 = 2$
4	$\langle 18,2\|$	$(3,3)$	$(3,3)$	$2 \rightarrow i_0 = 3, i_1 = 3$
5	$\langle 19,1\|$	$(3,4)$	$(3,4)$	$1 \rightarrow i_0 = 3, i_1 = 4$

Por ejemplo, el ítem $\langle 3,0|$ tiene $I_{inf} = (0,0)$ y $I_{sup} = (0,3)$ y la explicación es $r_0 = 0 \rightarrow A_1 = a_{1,0} = 0, A_2 \leq 3$. □

Ejemplo 2.10. Sean dos atributos A, B que toman valores $\{f,t\}$ y la *Base* $[A, B]$. La respuesta de la tabla es la del operador lógico *AND* entre los valores de A y B. La lista *KBM2L* es:

$$\langle (t,f), f|\ \langle (t,t), t|$$

El primer ítem recoge a la mayoría de los casos de la tabla, con respuesta f. Su parte fija es vacía: $I_{inf} = (f,f), I_{sup} = (t,f)$. Por el contrario, el segundo ítem tiene un solo caso de la

tabla. Su parte variable es vacía y su parte fija es igual a $I_{inf} = I_{sup} = (t, t)$. Mientras que la explicación del segundo ítem es $t \rightarrow (A = t \text{ y } B = t)$, la explicación del primero no existe. Podemos considerar la negación de $(A = t \text{ y } B = t)$ como explicación complementaria, en este caso que la respuesta es dicotómica. Pero en definitiva, no explicamos propiamente el primer ítem sino su complementario. También una construcción algo más compleja, es $f \rightarrow A = f$ o $(A = t \text{ y } B = f)$, que hace una partición implícita del ítem y revela un conocimiento aparente. Si cambiamos de *Base* a $[B, A]$ dicha "explicación" es $f \rightarrow B = f$ o $(B = t \text{ y } A = f)$, y creemos que es mejor decir que no hay explicación para evitar tener explicaciones muy complicadas distintas en diferentes *Bases*. Éste es un ítem sin parte fija ni ADR que ilustra la consecuencia C5.

En este ejemplo no se aprecia la ventaja de explicar mediante la parte fija, que es un pequeño subconjunto de atributos instanciados. Pero si el esquema tiene muchos atributos y la respuesta un dominio no binario, un ítem puede tener una explicación eficiente, en la *Base* óptima. □

Ejemplo 2.11. Sean tres atributos A, B, C que toman valores $\{f,t\}$ y la *Base* $[A, B, C]$. La respuesta de la tabla es la del operador lógico: $(a \ AND \ b \ OR \ c)$, donde a, b y c son valores cualesquiera de los atributos. La lista *KBM2L* es:

$$\langle (f, f, f), f| \ \langle (f, f, t), t| \ \langle (f, t, f), f| \ \langle (f, t, t), t| \ \langle (t, f, f), f| \ \langle (t, t, t), t|$$

La lista está muy fragmentada y la mayor parte de los ítems necesitan de la participación de todos los atributos para construir su explicación, lo cual es poco eficiente. En la *Base* $[C, A, B]$ la lista *KBM2L* es:

$$\langle (f, t, f), f| \ \langle (t, t, t), t|$$

Ahora tenemos una *Base* óptima. El primer ítem es explicado mediante $f \rightarrow C = f$. El segundo ítem con parte fija vacía, mediante $t \rightarrow C = t$ o $(C = f \text{ y } A = t \text{ y } B = t)$ que ilustra también la consecuencia C5 apuntada respecto de las explicaciones de ítems grandes. □

2.4.2. Explicación y Relevancia en Sistemas de Ayuda a la Decisión

El tema de la explicación en SAD es un área de investigación muy activa donde conviven muchas propuestas. En esta sección tan sólo nos aproximamos al fundamento de la explicación en las listas *KBM2L*. Los capítulos 4, 5 y 6 muestran ejemplos de explicaciones y análisis de sensibilidad (AS), aspectos de la ingeniería del conocimiento que están estrechamente relacionados.

Relevancia y Explicación A continuación concretamos los conceptos expuestos en la sección 2.4.1 que podemos aplicar a las tablas de un SAD.

La relevancia de atributos es importante en la construcción de un modelo de un problema para un SAD porque permite estudiar el papel que desempeñan los conceptos en el resultado de la inferencia (John et al., 1994). Del mismo modo que la relevancia, la explicación, en un SAD, es importante porque sustenta una argumentación a favor de las inferencias que muestra el sistema. La construcción de explicaciones basadas en atributos relevantes es un caso particular del problema de selección de características. En (Kohavi, 1995b) encontramos una exposición detallada de diferentes aproximaciones a la solución de este problema y la propuesta planteada en dicha tesis. Para nosotros consiste en seleccionar el conjunto más pequeño de atributos suficiente para que la respuesta sea una determinada.

Un conjunto de atributos es relevante en cierto proceso de toma de decisión o de clasificación si la decisión o la clase es sensible a cambios en los valores que tienen los atributos (John et al., 1994). En esta sección trataremos de formalizar la observación anterior y estudiar sobre la lista *KBM2L* de las TDO y TPC procedimientos para caracterizar la revelancia de los atributos del esquema. En principio, todos los atributos del esquema son relevantes, son atributos del nodo de valor cuando se obtiene la TDO correspondiente mientras se evalúa el DI, o bien son atributos padres de la variable aleatoria condicionada en la TPC. Pero un análisis más profundo nos sugiere que fijados ciertos atributos, que definen un subproblema o contexto particular, no todos son igualmente relevantes.

La explicación de las propuestas del SAD (contenidas en las TDO) y las relaciones entre conceptos que sostiene el modelo (representadas en las TPC) es un tema que está presente en todos los sistemas basados en el conocimiento: sistemas expertos probabilistas, sistemas de ayuda a la inferencia y a la decisión, sistemas de clasificación, sistemas de diagnóstico,... Se trata de explicar por qué el sistema responde de cierto modo; cómo llega a cierta conclusión; qué elementos son necesarios para manifestar el comportamiento observado; por qué no responde de cierta manera, tal vez, esperada por el usuario. Hay varias razones que dificultan la explicación. Destacamos tres: el conocimiento y razonamiento involucrado puede ser complejo, el modelo de razonamiento que soporta el problema puede ser muy poco intuitivo, el usuario y receptor de las explicaciones puede ser muy variable respecto de sus expectativas. Esto da lugar a diferentes aproximaciones o paradigmas de explicación según el problema y el usuario y diversas técnicas para los diferentes modelos (Lacave, 2003).

La relevancia de los atributos en las recomendaciones que proponen las tablas y la explicación de las decisiones óptimas son dos aspectos que analizaremos conjuntamente al procesar las

consultas dirigidas a la BC, véase el capítulo 4.

La explicación es muy compleja (Lacave, 2003) porque debe satisfacer al usuario y debe conocer el modelo en profundidad. Es decir, la construcción de sistemas que suministren buenas explicaciones es muy difícil porque (Henrion et al., 1991):

1. La explicación debe presentarse desde todos los puntos de vista posibles de forma estructurada y jerárquica.

2. Los modelos de explicación para usuarios y analistas deben estar a niveles diferentes. Por tanto, deben dejar los aspectos internos del modelo de razonamiento y del motor de inferencia en un segundo nivel.

3. Deberían usar tan sólo conocimiento del dominio del usuario (Druzdzel, 1996).

4. Las explicaciones deberían ser lo más generales posible y hacer énfasis tanto en la presencia de argumentos a favor de la propuesta como en la ausencia de argumentos en contra.

5. La explicación tiene puntos débiles como consecuencia de la incertidumbre y subjetividad inherentes al proceso de toma de decisiones. Esta debilidad debe mostrarse.

Así, se pretende resolver cuestiones como (Henrion et al., 1991):

- detectar conflictos entre hallazgos,
- detectar errores en el modelo,
- convencer al usuario,
- enseñar al usuario y
- mostrar la sensibilidad del resultado frente a los hallazgos.

En el trabajo (Lacave, 2003) encontramos un estudio actual sobre las capacidades de explicación para sistemas basados en modelos de RB. Hay tres aspectos a explicar:

- la BC, mediante la visualización gráfica, verbal o con marcos de la información, llamada explicación estática o del modelo,
- la evidencia introducida (abducción) y
- el proceso de razonamiento seguido (explicación del razonamiento o explicación dinámica).

En el contexto de la lista *KBM2L*, que podría ser un conjunto de casos inferidos por una RB o una TDO, tenemos un planteamiento similar. La visualización la mostramos en la sección 2.5 mediante el gráfico llamado *espectro* de la lista. La gestión de múltiples casos como evidencia es un objetivo esencial de la representación de la tabla como lista *KBM2L*, la cual trata de sintetizar los grupos de casos o ítems más afines y explotar la organización de la lista óptima para explicar los contenidos de la BC. La explicación de un caso se abordaría estudiando el ítem propietario de la lista óptima, y los ítems adyacentes que tienen respuesta diferente debido a algunos (pocos) atributos que explican el cambio de respuesta. La evidencia asociada a ciertos atributos en ciertos contextos de la BC permite establecer los atributos discriminantes respecto de las respuestas. Por último, el proceso de razonamiento no es accesible en la lista *KBM2L* y por tanto, debemos recurrir al modelo gráfico y al algoritmo de propagación para aportar una explicación de carácter dinámico.

Los trabajos mencionados insisten en la importancia de la explicación como requisito para el usuario del SAD. Concretamente (Lacave, 2003) comenta la traza del algoritmo como tentativa de mecanismo de explicación del razonamiento. Trata sobre cuestiones como la multitud de hallazgos relacionados con un caso y los conflictos que presentan. Pretende establecer la influencia de cada factor de evidencia sobre el resultado. Sin embargo, los algoritmos son particularmente difíciles de entender (propagación de evidencia en RB, evaluación de DI,...) precisamente por implementar mecanismos de razonamiento muy generales. La explicación de este aspecto es importante debido a que la asignación de probabilidades, por ejemplo en los modelos que utilizan DI, es una tarea difícil, tanto si las distribuciones se aprenden automáticamente con datos, como si son asignadas subjetivamente por expertos. La cuestión es la estrecha relación existente entre la explicación y el AS del modelo respecto de sus parámetros, véase el capítulo 6.

Por tanto, propone un modelo de explicación que se fundamenta en los elementos mencionados y aspira a resolver las dificultades que no supera la traza de la inferencia. Proporciona una implementación de las capacidades de explicación en (Elvira, 2002), que es un programa de edición y evaluación de MGP (RB y DI).

La explicación en el contexto de esta tesis no trabaja con la traza del algoritmo, que está oculto. Para nosotros la inferencia o evaluación del modelo ha concluido. Hemos generado una lista (o varias) con el resultado (tablas de casos, hechos, reglas, políticas óptimas,...) y la hemos configurado de forma óptima. La lista es la BC del sistema y sobre ella pretendemos generar la explicación. Nuestra propuesta complementa el MGP inicial y otros mecanismos de explicación (probabilísticos). La característica principal es la explicación mediante la evidencia del resultado.

La explicación se clasifica por su *contenido*, medio de *comunicación* y *adaptación* al receptor

(Lacave, 2003). Para la lista *KBM2L*:

1. El contenido es la relación entre la respuesta y los atributos relevantes que obtenemos al identificar la parte fija de los ítems de la BC. El propósito es la descripción de los ítems y la relación entre las descripciones de las respuestas de interés que aportan cierta comprensión sobre la respuesta y la propia descripción. Los niveles de explicación sencillamente se corresponden con la explicación de los ítems (nivel macro) o de los casos individuales (nivel micro).

2. La comunicación, es decir, cómo interactúa el sistema con el usuario, se lleva a cabo mediante texto para las reglas de decisión inversas y directas (Pawlak, 2002) y gráficos como el espectro de la lista. Si el modelo de representación del conocimiento que origina la BC está disponible, la explicación se complementa con la correspondiente información cualitativa y cuantitativa, por ejemplo un DI. El diálogo entre el usuario y el sistema tiene lugar en una sesión de consultas, véase la sección 4.3, donde se plantea la necesidad de producir explicaciones.

3. La adaptación al receptor se realiza mediante el lenguaje proposicional basado en reglas, la longitud mínima de las explicaciones y relevancia de los atributos, que se manifiesta en el *peso* asociado en la *Base* óptima. Salvo que el usuario pida una explicación relativa a los parámetros del modelo (probabilidades, utilidades,...) todo el discurso es fijo, pues se circunscribe al esquema de la tabla que se supone perfectamente conocido. En nuestra propuesta consideramos el nivel de máximo detalle respecto del dominio y de mínimo nivel de detalle respecto del modelo. Creemos que los detalles del modelo le interesan exclusivamente al analista, que dispone de medios para inspeccionarlo, mientras que el experto o usuario toma al modelo como una herramienta. En resumen, el usuario desconoce los fundamentos de la tecnología y le resultan útiles sólo las explicaciones conceptuales independientes del nivel operativo del modelo.

El AS aumenta la confianza en el modelo y los resultados. Se trata de una tarea muy compleja, incluso para modelos sencillos. Encontramos autores que consideran el AS como una herramienta para generar explicaciones (Henrion et al., 1991). Precisamente tal relación nos llevará en el capítulo 6 a estudiar el AS de modelos (DI) con ayuda de las listas *KBM2L*. Por último, la explicación tiene relación con la validación del modelo del problema y con el aprendizaje (construcción y mejora del modelo).

El punto de vista anterior contempla el aspecto operativo de la explicación frente al conceptual. El primero es el más sencillo de concretar una vez que se conoce el objeto a explicar.

Pensemos en una explicación de una proposición genérica, contenida en una BC.

En (Mayfield, 2000) se toma como definición de explicación de un evento el conjunto de condiciones que permiten o causan la ocurrencia del evento. Categorías de las condiciones anteriores son: el estado del problema; relaciones causales; creencias, intenciones e influencias de agentes; otros eventos y/o acciones; leyes físicas;... La caracterización de una buena explicación es, según (Mayfield, 2000), dependiente de tres propiedades o criterios: aplicabilidad, conocimiento básico y completitud. La aplicabilidad recoge el hecho de que la explicación sea la que se necesita en el sistema o usuario. El conocimiento básico es el hecho de que los agentes del diálogo de explicación conozcan la explicación que se ofrece y no *sorprenda*. La completitud es tal si cubre en profundidad todos los aspectos de la proposición que se explica y no deja ningún aspecto oscuro. La calidad de la explicación no depende del algoritmo que la construye.

Entendemos que la BC que se construye con una lista *KBM2L* se ajusta bien a la definición anterior. El objeto de nuestra explicación son proposiciones almacenadas en una tabla. Desde el punto de vista de la calidad de las explicaciones, podemos decir que las tablas son el resultado de evaluar un modelo validado de un problema de decisión, o un modelo de probabilidad de una variable multidimensional, en cierto contexto de condicionantes y/o causas. El esquema recoge toda la información relevante del modelo y es conocido por el usuario del sistema. Las tablas tienen toda la combinatoria que permite garantizar la completitud de las explicaciones. En este sentido, expresado en (Mayfield, 2000), creemos que las explicaciones propuestas tienen una calidad óptima si añadimos un cuarto criterio económico de mínima longitud.

La explicación, tal como establecimos al comienzo de esta sección, es la parte fija de un ítem. Cuanto mayor sea el número de casos del ítem, menor es el conjunto de atributos de su parte fija y más sencilla es la explicación. El peso de un atributo está relacionado con la importancia de los atributos y la capacidad para explicar el conocimiento que recoge la BC. Observamos que tanto la optimización del almacenamiento de las tablas como la búsqueda de explicaciones eficientes (sencillas) son de alguna manera el mismo problema. La primera tarea tiene su objetivo en el plano sintáctico y la segunda en el semántico.

La explicación de un caso, de un ítem, o de la familia de cierta respuesta, es interesante desde el punto de vista de una consulta pero es bastante probable que el usuario del SAD quiera ir aún más lejos. En concreto, es también interesante tener una visión de conjunto en la BC que permita analizar el cambio de política observado respecto de los casos de ítems adyacentes. Identificamos a los atributos de la explicación que cambian su valor en los ítems anterior y/o posterior y por tanto son discriminantes. En la sección 4.2.2 mostramos un ejemplo utilizando la lista *KBM2L* del SAD PGNHL. En dicho ejemplo se muestra también un análisis de la familia de una política

óptima. Vemos la complementariedad del estudio de la BC mediante la minimización de la lista (global) y de una modalidad específica de la respuesta (local).

Se cumplen ciertas propiedades como:

1. Este modelo de explicación depende de la *Base*. En la *Base* óptima la explicación es más concisa y clara.

2. La explicación debe ser mínima en el número de atributos en la cláusula (necesarios y suficientes).

3. El tamaño de un ítem (cardinal del conjunto de casos que representa) y el tamaño de la cláusula de explicación (cardinal del conjunto de atributos que explican) están en relación inversa. Por ejemplo, un ítem de un solo caso necesita ser explicado por todos los atributos del esquema instanciados. Por el contrario, un ítem que represente todos los casos de la matriz tiene explicación vacía, parte fija vacía, no hay conocimiento o hay demasiado para explicarlo con reglas.

4. La unión de ítems, vease la subsección 2.4.3, simplifica y especializa las explicaciones respectivas.

5. La falta de información, desconocimiento o ignorancia ($unKB$), no tiene explicación, aunque tiene parte fija. La respuesta $coKB$, aunque no es parte del dominio de la respuesta sí puede tener explicación. Es por naturaleza una proposición sobre la incompatibilidad de ciertas combinaciones de valores de los atributos que aparecen al desarrollar el esquema completamente.

6. El orden fijo y arbitrario del dominio de los atributos puede dificultar la extracción de explicaciones. Las dos técnicas, permutación de orden en el dominio o modificación del esquema (descomposición de atributos), permiten manejar todos los grados de libertad en las bases posibles y generar, aunque con mayor complejidad computacional, explicaciones completas.

2.4.3. Síntesis del Conocimiento: Uniones de Ítems

Recordemos que una *Base* es mejor que otra si la lista *KBM2L* resultante es más pequeña. Claramente tenemos que reorganizar la lista en la *Base* que permita que los ítems actuales se unan, recolocando los ítems de igual respuesta de forma adyacente.

Si un atributo desarrolla todo su dominio sobre los términos de la explicación se puede sacar de la misma, simplificándola. Si el resto de atributos son factor común se presenta la posibilidad

de una *Base* mejor donde se produce la unión de los ítems. Por ejemplo, $A = a_0, B = b_0, C = c_0$ y $A = a_0, B = b_1, C = c_0$, donde el dominio del atributo *Base* es $\{b_0, b_1\}$. La explicación en notación algebraica es:

$E = a_0b_0c_0 + a_0b_1c_0 = a_0c_0(b_0 + b_1) = a_0c_0$.

La unión que describimos conecta con la simplificación de la explicación expresada de forma clausular, en forma normal conjuntiva (FNC) (Karnaugh, 1953).

Por ejemplo, supongamos que tenemos dos ítems con igual respuesta, y que están separados por otros ítems con otra respuesta. Sean p y q el desplazamiento de cada uno, respectivamente:

$\langle p, x\vert$ con índice $(i_0, i_1, i_2, i_3, i_4)$,
donde i_0, i_1, i_2 son fijos e i_3, i_4 son variables
$[\langle p, x\vert] \leq \vert i_3\vert * \vert i_4\vert$
$\langle q, x\vert$ con índice $(i_0, i_1, i_2, i_3, i_4)$,
donde i_0, i_1, i_2, i_3 son fijos e i_4 es variable
$[\langle q, x\vert] \leq \vert i_4\vert$

donde $[\langle p, x|]$ y $[\langle q, x|]$ denotan el número de casos que incluye cada ítem, e $|i_j|$ denota el cardinal del dominio del atributo i_j. Además, no hay ADR en ningún ítem.

Si queremos que los ítems se unan, debemos transponer algunos atributos en la *Base*, en concreto $\{i_0, i_1, i_2\}$. Algunos perderán peso y otros irán a posiciones de mayor peso. Señalemos que si alguno de los atributos fijos toma todo su dominio en la familia, es irrelevante y podemos simplificar la parte fija de ambos ítems si le asignamos menor peso y lo llevamos a la parte variable. Potencialmente podríamos unir los ítems mediante esta transposición. Así, supongamos que todos los atributos son binarios (0 y 1) y la situación que se muestra a continuación:

$\langle p, x\vert$ con índice $(i_0, i_1, i_2, i_3, i_4)$,
donde $i_0, i_1, i_2 = 0$ son fijos e i_3, i_4 son variables
$[\langle p, x\vert] = \vert i_3\vert * \vert i_4\vert$
$\langle q, x\vert$ con índice $(i_0, i_1, i_2, i_3, i_4)$,
donce $i_0 = 0, i_1 = 1, i_2 = 0, i_3 = 0$ son fijos e i_4 es variable
$[\langle q, x\vert] = \vert i_4\vert$

En este caso, el atributo segundo toma todos sus valores en la familia formada por los dos ítems. Debemos entonces moverlo a la parte variable. Ambos ítems de desplazamientos p y q se unen en un ítem de desplazamiento r. La parte fija queda simplificada:

$\langle r, x\vert$ con nuevo índice $(i_0, i_2, i_1, i_3, i_4)$
donde ahora $i_0, i_2 = 0$ son fijos e i_1, i_3, i_4 son variables
$[\langle r, x\vert] = \vert i_1\vert * \vert i_3\vert * \vert i_4\vert$

Ejemplo 2.12. Sea la siguiente lista *KBM2L*. La *Base* inicial es $[0, 1, 2, 3, 4]$ con atributos de dominio binario. El espacio de representación de la lista (tabla) es de 32 casos. La lista *KBM2L* inicial en notación desplazamiento es:

$$\langle 3, x\vert\langle 7, y\vert\langle 9, x\vert\langle 31, y\vert$$

donde $p = 3$ y $q = 9$.

Y en notación índice es:

$$\langle (0, 0, 0, 1, 1), x\vert\ \langle (0, 0, 1, 1, 1), y\vert\ \langle (0, 1, 0, 0, 1), x\vert\ \langle (1, 1, 1, 1, 1), y\vert$$

La lista final después de la unión de los ítems en la *Base* $[0, 2, 1, 3, 4]$ es, en notación desplazamiento:

$$\langle 5, x\vert\ \langle 31, y\vert,$$

donde $r = 5$.

En notación índice es:

$$\langle (0, 0, 1, 0, 1), x\vert\ \langle (1, 1, 1, 1, 1), y\vert$$

Observamos que la unión de los ítems de respuesta x implica la unión de los ítems de respuesta y. Así, la MM será en notación desplazamiento:

$$(0, x)\ (1, x)\ (2, x)\ (3, x)\ (4, x)\ (5, x)\ (6, y)\ (7, y)\ (8, y) \ldots (30, y)\ (31, y)$$

Y en notación índice:

$$((0, 0, 0, 0, 0), x) \ldots ((0, 0, 1, 0, 1), x)$$
$$((0, 0, 1, 1, 0), y) \ldots ((1, 1, 1, 1, 1), y)$$

Observamos los vectores de atributos I_{inf} y I_{sup}. La parte fija del primer ítem es (i_0, i_2) y la parte variable (i_1, i_3, i_4). Para el segundo ítem, su parte fija es vacía y su parte variable es $(i_0, i_2, i_1, i_3, i_4)$.

El Cuadro 2.3 contiene un resumen de esta operación de unión de dos ítems mostrando sólo los casos de respuesta x. La *Base* inicial es $[0, 1, 2, 3, 4]$ y la nueva *Base* es $[0, 2, 1, 3, 4]$.

Cuadro 2.3: Unión de ítems correspondientes a los desplazamientos p y q

La *Base* inicial es $[0,1,2,3,4]$						
i_0	i_1	i_2	i_3	i_4	desplazamiento	respuesta
0	0	0	0	0	0	x
0	0	0	0	1	1	x
0	0	0	1	0	2	x
0	0	0	1	1	p=3	x
0	1	0	0	0	8	x
0	1	0	0	1	q=9	x
La nueva *Base* es $[0,2,1,3,4]$						
i_0	i_2	i_1	i_3	i_4	desplazamiento	respuesta
0	0	0	0	0	0	x
0	0	0	0	1	1	x
0	0	0	1	0	2	x
0	0	0	1	1	3	x
0	0	1	0	0	4	x
0	0	1	0	1	r=5	x

□

El $XBase$ repercute en la unión/fragmentación de los ítems. Y ésta es la clave. Las *Bases* mejores son las que presentan familias de ítems menos fragmentadas. Tal y como hemos visto, el análisis de los ítems fragmentados aporta conocimiento acerca de las características de dichas *Bases*, por ejemplo indicando qué atributos, al perder peso, favorecen las uniones.

El próximo capítulo describe cómo se optimiza el almacenamiento y se producen las explicaciones mejores en una lista.

2.5. Exploración y Visualización de Listas *KBM2L*

Esta sección aborda, en el contexto de las listas *KBM2L*, la representación gráfica de datos multidimensionales, de dimensión muy grande.

2.5.1. Presentación de la Información de una Tabla

Dado el esquema y una *Base* de la tabla, podemos mostrar de diferente modo la tabla y las listas asociadas.

Ejemplo 2.13. Supongamos la tabla del ejemplo 2.1 (sección 2.1) con algunas respuestas desconocidas (*unKB*) denotadas con -1. El espacio de representación de la tabla es de 20 casos.

$[A, B]$	0	1	2	3	4
0	0	0	0	0	1
1	1	1	1	1	1
2	1	1	1	1	1
3	-1	-1	1	2	-1

Algunas posibles vistas de la tabla presentan la información de las siguientes formas:

1. La tabla completa como MM, con todas las entradas explícitas (imposible en casos reales). No es necesario mostrar el desplazamiento.

   $$(0,0)\ (1,0)\ (2,0)\ (3,0)\ (4,1)\ (5,1)\ (6,1)\ (7,1)\ (8,1)\ (9,1)\ (10,1)\ (11,1)\ (12,1)\ (13,1)$$
   $$(14,1)\ (15,-1)\ (16,-1)\ (17,1)\ (18,2)\ (19,-1);\ \text{con 20 casos.}$$

2. Sólo las entradas conocidas explícitas (imposible en casos reales).

   $$(0,0)\ (1,0)\ (2,0)\ (3,0)\ (4,1)\ (5,1)\ (6,1)\ (7,1)\ (8,1)\ (9,1)\ (10,1)\ (11,1)\ (12,1)\ (13,1)$$
   $$(14,1)\ (17,1)\ (18,2);\ \text{con 17 casos.}$$

3. La tabla explícita sin las entradas que tienen como respuesta la moda de toda la tabla (1 en este ejemplo) (imposible en casos reales).

   $$(0,0)\ (1,0)\ (2,0)\ (3,0)\ (15,-1)\ (16,-1)\ (18,2)\ (19,-1);\ \text{con 8 casos.}$$

4. La lista *KBM2L* en su configuración óptima con todas las entradas implícitas, conocidas y desconocidas.

   $$\langle 3,0|\ \langle 14,1|\ \langle 16,-1|\ \langle 17,1|\ \langle 18,2|\ \langle 19,-1|;\ \text{con 6 ítems.}$$

5. En problemas que manifiesten restricciones muy fuertes, se puede dejar el contenido sometido a restricción junto a las entradas desconocidas (*unKB*). La marca de restricción *coKB* introducida en la sección 2.1, es una extensión del dominio de las respuestas imposibles o infactibles en el espacio de representación del esquema, denotadas con -2. Por ejemplo, si $A = 2$ o $B = 2$ es imposible, salvo que sean ambos iguales a 2:

$$\langle 1,0|\ \langle 2,-2|\ \langle 3,0|\ \langle 6,1|\ \langle 7,-2|\ \langle 9,1|\ \langle 11,-2|\ \langle 12,1|\langle 14,-2|\langle 16,-1|$$
$$\langle 17,-2|\langle 18,2|\langle 19,-1|;\ \text{con 13 ítems.}$$

6. La lista de vectores I_{inf} e I_{sup} de cada ítem en su configuración óptima, con entradas conocidas y desconocidas. Esta vista proporciona explicaciones basadas en instancias parciales y mínimas del esquema, véase el ejemplo 2.9.

$$\langle (0,0)(0,3),0|\ \langle (0,4)(2,4),1|\ \langle (0,3)(3,1),-1|\ \langle (3,2)(3,2),1|\ \langle (3,3)(3,3),2|$$
$$\langle (3,4)(3,4),-1|;\ \text{con 6 ítems.}$$

□

La presentación de la información y la propuesta de explicaciones está estrechamente relacionada. El estudio de diferentes formas de presentación eficientes nos ha conducido a formular la explicación en los términos descritos en la sección 2.4.

Cada vista es interesante para una tarea concreta. Si las tablas son pequeñas podemos utilizar la primera vista, por ejemplo, si se trata de proyecciones que involucran a pocos atributos. Si hay mucha información desconocida o la moda de las respuestas es muy grande en la lista, la segunda y tercera vista son buenas opciones, respectivamente. La cuarta vista es la más adecuada en el caso general de una lista con una granularidad aceptable. La quinta vista es útil cuando tenemos conocimiento en forma de restricciones semánticas sobre el esquema de la tabla y cuando descomponemos el esquema sintácticamente, véase la sección 5.1.

Los procedimientos descritos pueden parecer sencillos. Sin embargo, debemos pensar que las tablas de interés tienen decenas de atributos en el esquema y son imposibles de almacenar. Además, las tablas deben desarrollarse y completarse de forma inteligente para poder extraer todo el conocimiento que poseen.

2.5.2. Espectros y Mapas de Listas *KBM2L*

Las representaciones usuales de datos multidimensionales (Manly, 1994), tales como polígonos estrellados, matrices de diagramas de dispersión, diagramas de caras, curvas de Andrews,... que utilizan diversos recursos (dimensiones espaciales, iconos, color, texturas,...) (SAS, 1990; R Development Core Team, 2005; Swayne et al., 1998; S-PLUS, 2003), no son adecuadas para el tipo de tablas que manejamos con las listas. Con tales técnicas, el hecho de tener dimensiones superiores a 30, nos limita a proyecciones muy sencillas de los datos.

En la representación de los datos se pueden utilizar herramientas lingüísticas (gramáticas simbólicas) y herramientas gráficas (gramáticas geométricas) para modelizar un problema. Proponemos la siguiente clasificación de las herramientas para representar y describir los datos:

1. Herramientas de representación mediante grafos para relaciones cualitativas (lógicas–estructurales, probabilistas–incertidumbre, temporales) entre conceptos. Se trata de redes y diagramas que representan conceptos, relaciones, influencias,... Por ejemplo: DI, RB, redes semánticas, redes de neuronas artificiales (RNA), redes de marcos, diagramas entidad/relación, redes de Petri,...

2. Herramientas de representación mediante funciones para definiciones de conceptos y relaciones cuantitativas entre variables. Por ejemplo: probabilidades, potenciales, utilidades, espectros de señal, posibilidades, combinacionales (lógica),...

3. Herramientas de representación mediante mapas para relaciones topológicas y métricas entre variables y objetos. Se fundamentan en el almacenamiento lógico de la información en el ordenador. Representan relaciones espaciales y temporales. Se construyen mediante proyecciones de MM del conocimiento del dominio. El índice se descompone en subíndices de todas las maneras posibles dando lugar cada descomposición a una cartografía de la MM. El almacenamiento puede ser lineal (*KBM2L*) y no lineal. La proyección de la MM implica establecer la relación o clase de adyacencia o vecindad, que permite estructurar el mapa. El mapa lo determina el esquema, la información, la dimensión, la *Base* y la clase de adyacencia.

Este último tipo es el que pretendemos introducir y desarrollar en esta tesis. En dimensión lineal 1 sólo hay una clase de adyacencia y es el caso de las listas *KBM2L*. La dimensión no lineal es mucho más complicada y constituye tema abierto para futuras investigaciones. Por consiguiente, nos centramos en los espectros de las listas.

Espectro de listas. La representación gráfica que proponemos está directamente inspirada en el almacenamiento lógico de la información de una tabla, el vector lineal de datos. Consideramos la lista como la proyección unidimensional de los casos que asumimos que tienen entidad multidimensional, donde se agrupan los casos adyacentes con igual respuesta.

Primero asignamos un color a cada modalidad de la respuesta. Segundo, establecemos una coordenada horizontal homóloga al desplazamiento de cada dato, aunque igualmente podría ser vertical. Asignamos un píxel a cada desplazamiento, con su coordenada relativa al gráfico y el

color de la respuesta. En último lugar, la dimensión vertical, no utilizada, es ajustada para que la representación sea clara.

El resultado es una representación de la tabla, y de la lista, en forma de banda multicolor. Normalmente asignamos el color blanco a *unKB* y el negro a *coKB*. Cada rectángulo coloreado representa un ítem de la lista, con área (o anchura) proporcional al número de casos o tamaño del ítem.

El gráfico obtenido lo denominamos **espectro de la lista**. Vemos algunos ejemplos en las Figuras 2.4 y 2.5.

Figura 2.4: Ejemplo de espectro de dos listas *KBM2L*

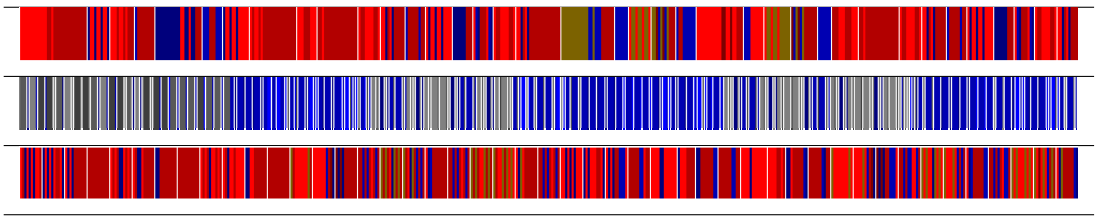

Figura 2.5: Ejemplo de espectros de listas *KBM2L*: lista de 341 casos en tres secciones de desplazamiento: 0-195 (arriba), 196-293 (medio) y 294-340 (abajo)

Como alternativa y mejora al gráfico anterior podemos considerar una representación que asocia la coordenada x desplazamientos, véase el segundo espectro de ítems en la Figura 2.6. Cuando una lista tiene ítems de tamaños muy diferentes, los más pequeños no se pueden observar junto a los grandes. Así, es útil en esta situación hacer equivalente el largo de los rectángulos al logaritmo del tamaño de los ítems. Este último gráfico es similar a un diagrama de barras de los ítems, no de la respuesta. Vemos algunos ejemplos en la Figura 2.6.

Podemos además trocear el espectro y mostrarlo en varias líneas, aunque esto puede ser más confuso, como en la Figura 2.5. Sin embargo, es esta forma la que sugiere que tal vez en ciertas tablas sea interesante utilizar un desplazamiento múltiple, donde *dos* índices se reparten los atributos del esquema, véase la sección 2.2.2. Considerar más de dos índices no es práctico de cara a la representación gráfica de una lista, aunque puede serlo para optimizar la tabla.

El espectro de una lista en dos *Bases* diferentes, con muy diferente grado de coalescencia, puede verse en la Figura 2.8. Ambos espectros describen la lista. El superior da idea del patrón

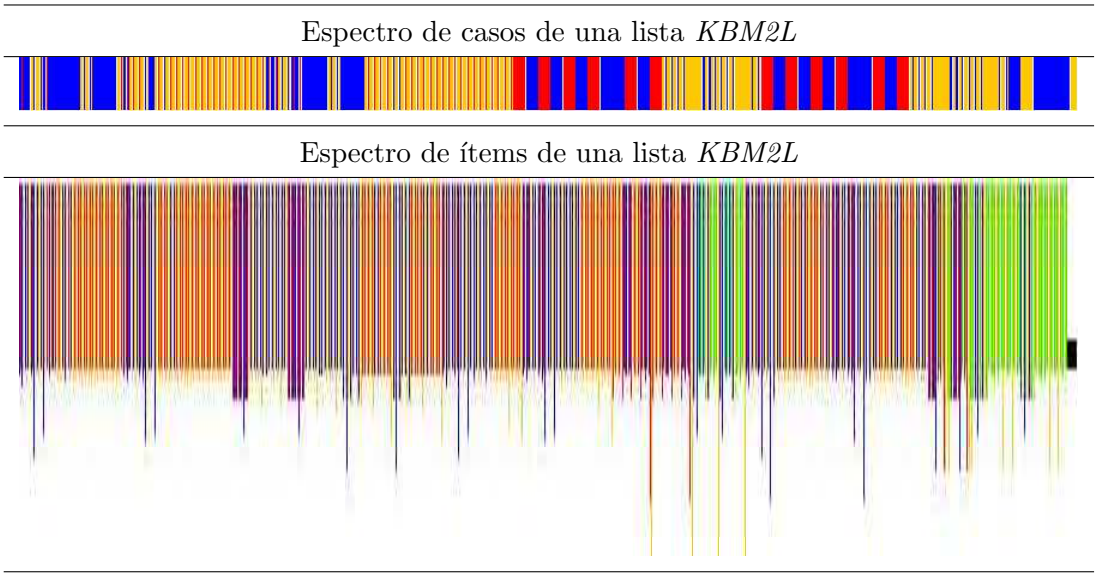

Figura 2.6: Ejemplo de espectros de listas *KBM2L*

tan fragmentado que sigue la respuesta. El segundo muestra la configuración de mínimo coste y máxima organización del conocimiento.

La aplicación implementa la lista *KBM2L* y diversos procedimientos para minimizar su tamaño, visualizar su contenido, extraer explicaciones, realizar consultas y análisis de sensibilidad. La solución software propuesta la denominamos *kbm32* y está realizada en Java 1.5.0_06. La documentación técnica del código y ejemplos se pueden consultar en: http://www.dia.fi.upm.es/-~jafernan/kbm32docs/index.html. La Figura 2.7 muestra las ventanas principal, de control, del browser y del espectro de una lista *KBM2L*.

La ventana *System* muestra las tareas en curso y posibles errores. La ventana *Browser* muestra el contenido de la lista *KBM2L* de trabajo. La ventana *Control* muestra, en diferentes secciones, parámetros de los distintos procedimientos (optimización, análisis, consultas,...) de modo interactivo. La ventana *Time* muestra tiempos de ejecución y eventos significativos del proceso de optimización, análisis, construcción, consulta y verificación de las listas. La ventana *Spectrum* muestra el espectro de la lista de acuerdo con las preferencias de la ventana de *Control* y permite consultar las propiedades de un ítem (I_{inf}, I_{sup}, desplazamiento, tamaño,...). Ciertas opciones de control, el gráfico del espectro y el browser constituyen un sistema de navegación sobre los ítems del la lista en un *Base* cualquiera. Al hacer clik con el botón izquierdo mostramos las propiedades internas del ítem (desplazamiento menor y mayor, respuesta, longitud, coordenadas en el gráfico,... y con el bóton derecho traducimos la información anterior a los literales del esquema)

La aplicacion *kbm32* está organizada en paquetes (Java packages) con las clases que implementan la lista *KBM2L*, las diferentes funcionalidades de interés y la interfaz de usuario. Todas las funcionalidades mencionadas se introducen, especifican y analizan en el resto de la tesis.

En primer lugar son: **kernel**, **target**, **tables** y **cpt**. El paquete **kernel** define las listas *KBM2L* y las operaciones para construir la lista de una TDO a partir de un fichero de texto de casos (capítulo 2). El paquete **target** permite reconfigurar las listas, es decir, crea la lista compatible con la inicial en la *Base* nueva requerida (capítulo 3). El paquete **tables** define la interfaz de ficheros de entrada (TDO) y salida (*KBM2L*) y permite procesar la evaluación del DI (un conjunto de TDO) para obtener la secuencia de listas del protocolo de decisiones del problema (capítulo 4). El paquete **cpt** implementa una extensión de **kernel** para trabajar sobre las TPC del modelo (capítulo 5).

En segundo lugar son: **analysis**, **learning**, **spectrum**, **synthesis**, **tools**, **query**, **schema** y **genetic**. El paquete **analysis** implementa el análisis sintáctico, semántico (explicaciones) y de análisis de sensibilidad extendiendo la *KBM2L* del modelo (capítulo 6). El paquete **learning** implementa la construcción incremental de las listas a partir de TDO incompletas resultado de evaluar subproblemas o instancias particulares del DI (capítulo 4). El paquete **spectrum** facilita la visualización gráfica del contenido de las listas (capítulo 2). Los paquetes **synthesis** y **genetic** implementan una batería de estrategias de optimización combinatoria para resumir las listas (capítulo 3). El paquete **query** aporta la interfaz y lógica necesarias para realizar un diálogo de consultas a la lista *KBM2L* donde el usuario puede especificar parcialmente los valores de las variables (capítulo 4). Los paquetes **schema** y **tools** suministran procedimientos varios para manipular la lista y los esquemas que definen su configuración y tamaño (capítulo 2).

En tercer lugar, los paquetes **kbm2l**, **menu** y **control** constituyen la interfaz con ventanas. El paquete **control** permite definir un amplio conjunto de parámetros que configuran desde la interfaz de usuario los procedimientos anteriores. El paquete **menu** recoge las opciones de la aplicación. Finalmente, el paquete **kbm2l** gestiona las ventanas, las el entorno de trabajo, los errores de la aplicación y ficheros log.

La aplicación define algo más de 120 clases que codifican la lista *KBM2L*, sus componentes y los procedimientos de análisis. Destacamos de entre ellas **listManager30** que implementa las reglas de gestión del contenido de las lista con operaciones como insertar un caso o un conjunto de casos, borrar casos,... (capítulo 2).

Mapas de listas. El espectro puede ser insuficiente al usar una única dimensión para describir la lista. Proponemos una extensión del espectro para listas cuyo esquema es muy complejo. Los **mapas** son espectros multidimensionales y, aunque son más generales, son complejos de analizar.

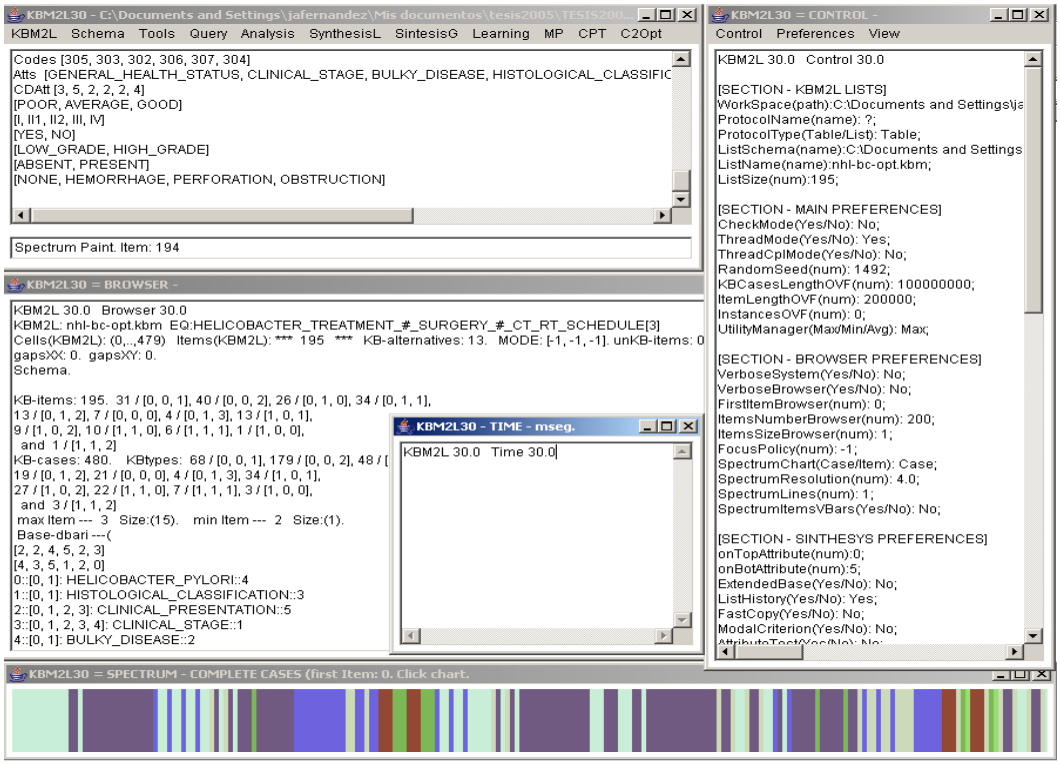

Figura 2.7: *KBM2L* Shell

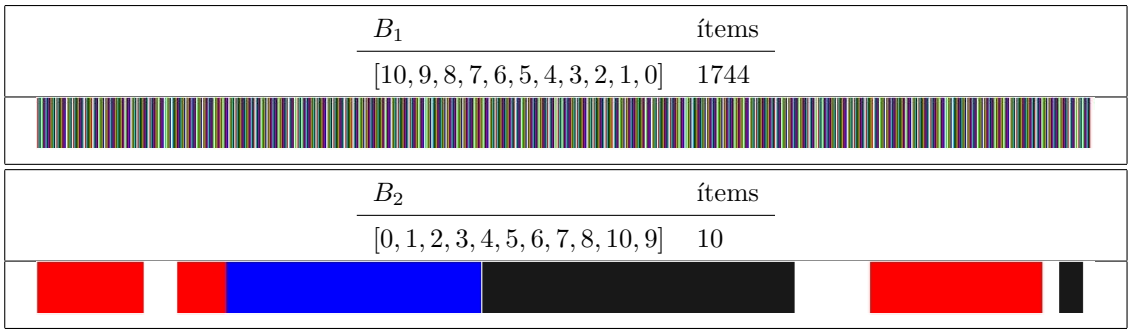

Figura 2.8: Ejemplo de espectros de listas *KBM2L*

Salvo en casos particulares, véase el capítulo 6,no es sencillo establecer los índices del mapa, es decir, qué atributos forman parte de cada índice. Si no hay información relativa a esto tendremos una doble dificultad: primero, la combinatoria asociada al considerar múltiples índices (éste es un problema similar a los que surgen en selección de características (John et al., 1994)), y segundo, la interpretación de los propios índices (éste es un problema similar a los que surgen en el análisis de componentes principales (Manly, 1994)).

Ejemplo 2.14. Supongamos un esquema de cuatro atributos {A,B,C,D}. La lista que ob-

tenemos no es satisfactoria e intentamos construir dos índices: I_1 e I_2. Los posibles mapas bidimensionales son:

	I_1	I_2
(1)	$\{A\}$	$\{B, C, D\}$
(2)	$\{B\}$	$\{A, C, D\}$
(3)	$\{C\}$	$\{A, B, D\}$
(4)	$\{D\}$	$\{A, B, C\}$
(5)	$\{A, B\}$	$\{C, D\}$
(6)	$\{A, C\}$	$\{B, D\}$
(7)	$\{A, D\}$	$\{B, C\}$

Para cada mapa podemos calcular el número de *Bases*. (1) tiene $1! * 3! = 6$, (2), (3) y (4) tienen 6 también. (5), (6) y (7) tienen $2! * 2! = 4$. Observamos que hay $4! = 24$ *Bases* del mapa unidimensional y hay $4 * 6 + 3 * 4 = 24 + 12 = 36$ *Bases* de los mapas bidimensionales.

Vemos que el proceso de búsqueda es más complejo si no se sabe cómo asignar los atributos a los índices. Si se conoce una buena asignación de índices, entonces es más sencillo. Supongamos que es la (3), entonces hay 6 *Bases*. □

En el caso bidimensional, el espacio de búsqueda pasa de ser $\delta!$ a $\delta_1! * \delta_2!$ con $\delta_1 + \delta_2 = \delta$, lo cual simplifica mucho, pero a costa de manejar conocimiento del dominio del problema o tabla original en la asignación de los atributos a cada dimensión. Por ejemplo, sabemos que ciertos atributos tienen una semántica temporal y los asignamos a una componente, mientras que el resto es estático y los asignamos a la otra componente.

2.6. Implementación de la Lista *KBM2L*: Gestión de Casos e Ítems

En esta sección abordamos la estructura y manipulación de la lista *KBM2L*. La implementación puede apoyarse tanto en el desplazamiento como en el índice, y en la práctica lo hace en ambos.

Introducimos algo de notación y procedemos a describir la implementación del algoritmo que permite mantener la lista consistente cuando se añaden casos o ítems. La lista inicial es: $\langle W_0 * \delta_0 - 1, unKB|$. Es decir, la consideramos vacía, aunque cualquier lista con un solo ítem y respuesta arbitraria está igualmente vacía de conocimiento. Por lista vacía entendemos cualquier lista que sólo tiene un ítem, con independencia de la respuesta, ya que si todos los casos presentan la misma respuesta, no hay conocimiento.

El software que hemos desarrollado para implementar BC mediante listas *KBM2L* emplea 27 reglas de gestión de los ítems de la lista y trabaja usando el *índice* (Clase Java Vector) y el *desplazamiento pregunta* (Clase Java BigInteger) (Sun Microsistems J.D.K., 2006).

Las reglas procesan las tablas y matrices sintetizando la BC mediante una lista *KBM2L*. Presentamos las reglas en tres grupos según se caracteriza el estado de la lista en el que se deben aplicar para introducir un caso nuevo en la BC. Las reglas distinguen dos estados fundamentales de la lista: vacía y no vacía. Si no está vacía, discriminan si el nuevo caso corresponde al primer ítem de la lista o no.

Una regla se ejecuta cuando sus precondiciones, que recogen los elementos del estado de la lista que son relevantes para la adquisición del caso, son verdaderas. En la implementación, primero se comprueban las precondiciones de las reglas del bloque inicial (reglas R0 a R5), si la lista es vacía. Después se comprueba si el nuevo caso corresponde al ítem primero de la lista, y es por tanto una regla del bloque segundo (reglas R6 a R14). Por último, se comprueba qué regla del bloque tercero (reglas R15 a R26), es la que cumple sus precondiciones. Siempre se ejecuta una regla y solamente una.

Sea d la respuesta que será asignada a la celda $\overrightarrow{c}$ de la MM que corresponde a un caso de la tabla. Sea $\overrightarrow{c} = (c_0, c_1, \ldots, c_{\delta-1})$ y $f(c_0, c_1, \ldots, c_{\delta-1}) = x$ su respectivo desplazamiento, véase la ecuación (2.1).

Mostramos en detalle, con la notación *ad hoc*, cómo es la lista *KBM2L* de ítems antes y después de la ejecución de cada regla. La aplicación de una regla supone un proceso elemental de aprendizaje sobre la lista que incorpora un caso $\langle x, d|$. La función *set* con argumentos x y d (desplazamiento y respuesta del caso, respectivamente) junto a la notación propia de la lista, muestran la precondición de cada regla y las diferencias entre el estado previo y posterior de la lista. El caso **válido** es aquel que se ajusta a la definición del esquema. Dicho caso pertenece a un único ítem de la lista considerada, su ítem propietario, que incluye el desplazamiento del caso nuevo.

Simbólicamente y algebraicamente las reglas son la operacionalización de la *suma* de dos listas, una genérica y otra con un solo caso: $\langle q_0, d_0|\ \langle q_1, d_1| \ldots \langle W_0 * \delta_0 - 1, d_{W_0*\delta_0-1}|$ y $\langle x-1, -1|\ \langle x, d|\ \langle W_0 * \delta_0 - 1, -1|$, que produce otra lista. Queremos indicar que el conjunto de reglas no es único y tiene interés práctico disponer de otras que sumen listas cualesquiera, ítems, bloques de ítems,...

La lista siempre está ordenada (es consistente conforme a los axiomas, sección 2.3.1) y la búsqueda de un caso se realiza en esta estructura en tiempo logarítmico por el hecho de estar ordenada, frente a lineal en la tabla. La agrupación de casos en ítems además aumenta mucho

más la velocidad de acceso y gestión. Nuestra implementación puede mejorarse al considerar la posibilidad de realizar operaciones en paralelo.

Sólo consideraremos la operación de adición de casos o ítems. No hay operación de borrado propiamente dicha, aunque la respuesta *unKB* cuando se inserta puede implementar el borrado, es decir, borrar es añadir casos con respuesta *unKB*. En cierto sentido se trata de considerar que la BC olvida el caso borrado. Para borrar celdas deberían considerarse cambios en el esquema tales como eliminar valores del dominio de los atributos o suprimir atributos.

Por último, la asignación es destructiva: se reemplaza el valor anterior, si es conocido, véase el ejemplo 3.12 que trabaja con un conjunto de datos y la necesidad de introducir la **dimensión de observación** (DdO). Esta característica dota a la lista de un comportamiento no monótono respecto de la representación del conocimiento que permite revisar los casos. Por una parte, si el caso que se introduce es desconocido produce la captura de la información y cambia el estado de la lista. Si el caso es conocido y la nueva respuesta es diferente, se actualiza, produciéndose una revisión del conocimiento. Ahora puede ocurrir que la nueva respuesta sea desconocida, lo cual produce una pérdida de memoria u olvido. Este posible comportamiento de la lista *KBM2L* como BC permite implementar políticas para gestionar recursos de memoria limitados. Así, la BC recuerda ciertos casos y olvida otros que no necesita, para realizar las tareas que tenga programadas.

Veamos ahora en detalle los tres grupos de reglas mencionados:

1. Las seis primeras reglas consideran el estado inicial de la lista, la lista vacía $\langle W_0 * \delta_0 - 1, -1|$, o una respuesta genérica d_0. La lista de ítems consiste en un solo ítem y capturamos un nuevo caso $\langle x, d|$ que es insertado. Las seis reglas se deducen de considerar la posición del caso a introducir. El desplazamiento x puede ser el menor posible (0), el mayor posible $(W_0 * \delta_0 - 1)$ u otro comprendido entre ambos. Por otra parte la respuesta d puede ser igual a la respuesta d_0 o distinta.

   R0 $set(x = 0, d = d_0)$ { $\langle W_0 * \delta_0 - 1, d_0| \rightarrow$
   $\langle W_0 * \delta_0 - 1, d_0|$ }
   La lista no cambia, el caso está contemplado en la lista inicial en el extremo inferior, de desplazamiento igual a 0. La respuesta es la misma que la del ítem propietario del caso. La regla es invariante.

   R1 $set(x = 0, d \neq d_0)$ { $\langle W_0 * \delta_0 - 1, d_0| \rightarrow$
   $\langle 0, d|\langle W_0 * \delta_0 - 1, d_0|$ }
   La lista cambia, el nuevo caso sobreescribe la información con su respuesta, d. El

gestor de la lista crea un nuevo ítem, en el extremo inferior del ítem propietario. El ítem nuevo tiene un solo caso y está al comienzo de la lista. El resto de casos están en el ítem segundo. La longitud se incrementa en un ítem.

R2 $set(x > 0, x < W_0 * \delta_0 - 1, d = d_0)$ { $\langle W_0 * \delta_0 - 1, d_0|$ → $\langle W_0 * \delta_0 - 1, d_0|$ }

La lista no cambia porque $d = d_0$. La regla es invariante.

R3 $set((x > 0), (x < W_0 * \delta_0 - 1),\ d \neq d_0)$ { $\langle W_0 * \delta_0 - 1, d_0|$ → $\langle x - 1, d_0|\langle x, d|\langle W_0 * \delta_0 - 1, d_0|$ }

La lista cambia. Se introduce un ítem de longitud un caso con la nueva información, $\langle x - 1, d_0|\langle x, d|$. Se introduce un primer ítem con al menos un caso que termina en la posición de desplazamiento $x - 1$. La respuesta de este primer ítem es la inicial. La longitud se incrementa en dos ítems.

R4 $set(x = W_0 * \delta_0 - 1, d = d_0)$ { $\langle W_0 * \delta_0 - 1, d_0|$ → $\langle W_0 * \delta_0 - 1, d_0|$ }

La lista no cambia porque $d = d_0$. A diferencia de R0 y R2, el desplazamiento del caso es el mayor posible. La regla es invariante.

R5 $set(x = W_0 * \delta_0 - 1, d \neq d_0)$ { $\langle W_0 * \delta_0 - 1, d_0|$ → $\langle W_0 * \delta_0 - 2, d_0|\langle W_0 * \delta_0 - 1, d|$ }

La lista cambia. Se introduce un ítem de un caso al final de la lista con la nueva respuesta y el desplazamiento máximo. Un ítem anterior recoge el resto de casos de la lista inicial desde la posición de desplazamiento 0 hasta la posición $W_0 * \delta_0 - 2$. La longitud se incrementa en un ítem.

Si en el estado inicial no se ejecuta ninguna de las reglas al capturar un caso en la lista, necesariamente o bien el índice del caso no es válido o el estado inicial no es consistente, véase la sección 2.3.

2. Las reglas R6 a R14 son inserciones de casos en el primer ítem de una lista cualquiera no vacía. El desplazamiento x es menor o igual a q_0, el desplazamiento del primer ítem.

R6 $set(x = 0, d = d_0)$ { $\langle q_0, d_0|\langle q_1, d_1| \ldots \langle W_0 * \delta_0 - 1, d_\sharp|$ → $\langle q_0, d_0|\langle q_1, d_1| \ldots \langle W_0 * \delta_0 - 1, d_\sharp|$ }

La lista no cambia porque $d = d_0$. La regla es invariante.

R7 $set(x = 0, q_0 = 0, d \neq d_1)$ { $\langle q_0, d_0|\langle q_1, d_1| \ldots \langle W_0 * \delta_0 - 1, d_\sharp|$ → $\langle q_0, d|\langle q_1, d_1| \ldots \langle W_0 * \delta_0 - 1, d_\sharp|$ }

La lista cambia si $d \neq d_0$. El primer ítem tiene un solo caso cuya respuesta es sobreescrita por la del caso nuevo, d, que como es distinto de d_1, mantiene el ítem en la lista. La regla es invariante.

R8 $set(x = 0, q_0 = 0, d = d_1)$ { $\langle q_0, d_0|\langle q_1, d_1| \ldots \langle W_0 * \delta_0 - 1, d_\sharp| \rightarrow$
$\langle q_1, d_1| \ldots \langle W_0 * \delta_0 - 1, d_\sharp|$ }

La lista cambia porque el primer ítem desaparece debido a que $d = d_1$. La regla es coalescente.

R9 $set(x = 0, q_0 \neq 0, d \neq d_0)$ { $\langle q_0, d_0|\langle q_1, d_1| \ldots \langle W_0 * \delta_0 - 1, d_\sharp| \rightarrow$
$\langle 0, d|\langle q_0, d_0|\langle q_1, d_1| \ldots \langle W_0 * \delta_0 - 1, d_\sharp|$ }

Si $q_0 \neq 0$ entonces el primer ítem tiene más de un caso. Se introduce un ítem en el extremo inferior del primer ítem, que alberga al caso nuevo pues la respuesta $d \neq d_0$. La longitud se incrementa en un ítem.

R10 $set(x = 0, q_0 \neq 0, d = d_0)$ { $\langle q_0, d_0|\langle q_1, d_1| \ldots \langle W_0 * \delta_0 - 1, d_\sharp| \rightarrow$
$\langle q_0, d_0|\langle q_1, d_1| \ldots \langle W_0 * \delta_0 - 1, d_\sharp|$ }

La lista no cambia. Las respuestas del caso nuevo y el ítem propietario, el primero, son iguales. La regla es invariante.

R11 $set(0 < x < q_0, d \neq d_0)$ { $\langle q_0, d_0|\langle q_1, d_1| \ldots \langle W_0 * \delta_0 - 1, d_\sharp| \rightarrow$
$\langle x - 1, d_0|\langle x, d|\langle q_0, d_0|\langle q_1, d_1| \ldots \langle W_0 * \delta_0 - 1, d_\sharp|$ }

El caso nuevo afecta al primer ítem y su desplazamiento x es interior al de dicho ítem. Si la respuesta d es distinta de la respuesta del primer ítem se introducen dos nuevos ítems. El primero alberga los casos de respuesta d_0 desde el desplazamiento 0 hasta el desplazamiento $x - 1$. El segundo ítem tiene un solo caso, el nuevo. La longitud se incrementa en dos ítems.

R12 $set(0 < x < q_0, d = d_0)$ { $\langle q_0, d_0|\langle q_1, d_1| \ldots \langle W_0 * \delta_0 - 1, d_\sharp| \rightarrow$
$\langle q_0, d_0|\langle q_1, d_1| \ldots \langle W_0 * \delta_0 - 1, d_\sharp|$ }

El caso nuevo afecta al primer ítem y su desplazamiento x es interior al de dicho ítem. Si la respuesta d es igual a la respuesta del primer ítem, no se introducen ítems. La longitud es invariante.

R13 $set(x = q_0, d \neq d_0, d \neq d_1)$ { $\langle q_0, d_0|\langle q_1, d_1| \ldots \langle W_0 * \delta_0 - 1, d_\sharp| \rightarrow$
$\langle x - 1, d_0|\langle x, d|\langle q_1, d_1| \ldots \langle W_0 * \delta_0 - 1, d_\sharp|$ }

El caso nuevo afecta al primer ítem y su desplazamiento x es igual al de dicho ítem. Si la respuesta d es distinta de la respuesta del segundo ítem se introducen dos nuevos ítems. El primero alberga los casos de respuesta d_0 desde el desplazamiento 0 hasta

el desplazamiento $x-1$. El segundo ítem tiene un solo caso, el nuevo. La longitud se incrementa en un ítem.

R14 $set(x = q_0, d = d_1)$ { $\langle q_0, d_0 | \langle q_1, d_1 | \ldots \langle W_0 * \delta_0 - 1, d_\sharp |$ $\rightarrow$
$\langle x-1, d_0 | \langle q_1, d_1 | \ldots \langle W_0 * \delta_0 - 1, d_\sharp |$ }

El caso nuevo afecta al primer ítem y su desplazamiento x es igual al de dicho ítem. Si la respuesta d es distinta de la respuesta del segundo ítem se introducen dos nuevos ítems. El primero alberga los casos de respuesta d_0 desde el desplazamiento 0 hasta el desplazamiento $x-1$. El resultado es como si el último caso del ítem primero se trasladase al ítem adyacente. La longitud es invariante.

3. Las reglas R15 a R26 son inserciones de casos en ítems o contextos de la lista interiores o al final de una lista no vacía. Las reglas R15 a R21 consideran el caso cuyo desplazamiento está en el extremo superior del ítem propietario. Las restantes reglas consideran desplazamientos en posiciones intermedias y el extremo inferior. Para localizar el (los) ítem(s) afectado(s) por el nuevo caso realizamos una búsqueda binaria en la lista. La lista tiene dos ítems al menos, de otro modo se trataría de una regla del primer bloque, R0 a R5. El caso a insertar pertenece al ítem de desplazamiento genérico q_{i+2} o q_{i+3}, que en cualquier caso no será el primer ítem de la lista, pues entonces se trataría de una regla del segundo bloque, R6 a R14. Los ítems de desplazamiento genérico q_{i+1} y q_{i+4} pueden no existir si exceden los límites de la lista y en esos casos la regla no cumple las precondiciones. Dependiendo del desplazamiento, la posición relativa (extremos o interior) de dicho ítem, y de la respuesta insertada y las adyacentes, obtenemos las siguientes reglas.

R15 $set(x = q_{i+3}, d = d_{i+3})$ { $\ldots \langle q_{i+2}, d_{i+2} |$ $\langle q_{i+3}, d_{i+3} | \ldots \rightarrow$
$\ldots \langle q_{i+2}, d_{i+2} | \langle q_{i+3}, d_{i+3} | \ldots$ }

La regla es invariante pues las respuestas son iguales.

R16 $set(x = q_{i+3}, q_{i+3} - q_{i+2} = 1, d \neq d_{i+2}, d \neq d_{i+3}, d \neq d_{i+4})$ {
$\ldots \langle q_{i+2}, d_{i+2} | \langle q_{i+3}, d_{i+3} | \langle q_{i+4}, d_{i+4} | \ldots \rightarrow$
$\ldots \langle q_{i+2}, d_{i+2} | \langle q_{i+3}, d | \langle q_{i+4}, d_{i+4} | \ldots$ }

La lista cambia la respuesta del caso de desplazamiento q_{i+3}. La regla es invariante, pues las respuestas son distintas pero el ítem propietario contiene un solo caso, $q_{i+3} - q_{i+2} = 1$.

R17 $set(x = q_{i+3}, q_{i+3} - q_{i+2} = 1, d = d_{i+2}, d \neq d_{i+4})$ {
$\ldots \langle q_{i+2}, d_{i+2} | \langle q_{i+3}, d_{i+3} | \langle q_{i+4}, d_{i+4} | \ldots \rightarrow$
$\ldots \langle q_{i+3}, d_{i+2} | \langle q_{i+4}, d_{i+4} | \ldots$ }

La lista cambia la respuesta del caso de desplazamiento q_{i+3} que al ser igual a la del ítem anterior es capturado por éste. La regla es coalescente.

R18 $set(x = q_{i+3}, q_{i+3} - q_{i+2} = 1, d = d_{i+2}, d = d_{i+4})$ {

$\ldots \langle q_{i+2}, d_{i+2}| \langle q_{i+3}, d_{i+3}| \langle q_{i+4}, d_{i+4}| \ldots \rightarrow$

$\ldots \langle q_{i+4}, d_{i+4}| \ldots$ }

La lista cambia la respuesta del caso de desplazamiento q_{i+3} que al ser igual a la de los ítems adyacentes y dicho caso constituir un ítem de longitud unidad produce la coalescencia de los tres ítems en el desplazamiento q_{i+4}. La regla es coalescente.

R19 $set(x = q_{i+3}, q_{i+3} - q_{i+2} = 1, d \neq d_{i+2}, d = d_{i+4})$ {

$\ldots \langle q_{i+2}, d_{i+2}| \langle q_{i+3}, d_{i+3}| \langle q_{i+4}, d_{i+4}| \ldots \rightarrow$

$\ldots \langle q_{i+2}, d_{i+2}| \langle q_{i+4}, d_{i+4}| \ldots$ }

La lista cambia la respuesta del caso de desplazamiento q_{i+3} que al ser igual a la respuesta del ítem siguiente es capturado por aquél. La regla es coalescente.

R20 $set(x = q_{i+3}, q_{i+3} - q_{i+2} > 1, d \neq d_{i+3}, d \neq d_{i+4})$ {

$\ldots \langle q_{i+2}, d_{i+2}| \langle q_{i+3}, d_{i+3}| \langle q_{i+4}, d_{i+4}| \ldots \rightarrow$

$\ldots \langle q_{i+2}, d_{i+2}| \langle q_{i+3} - 1, d_{i+3}| \langle q_{i+3}, d| \langle q_{i+4}, d_{i+4}| \ldots$ }

Se introduce un nuevo ítem con el caso de desplazamiento x y el ítem de desplazamiento q_{i+3} pierde un caso. La longitud se incrementa en un ítem.

R21 $set(x = q_{i+3}, q_{i+3} - q_{i+2} > 1, d = d_{i+4})$ {

$\ldots \langle q_{i+2}, d_{i+2}| \ \langle q_{i+3}, d_{i+3}| \langle q_{i+4}, d_{i+4}| \ldots \rightarrow$

$\ldots \langle q_{i+2}, d_{i+2}| \langle q_{i+3} - 1, d_{i+3}| \langle q_{i+4}, d_{i+4}| \ldots$ }

El caso de desplazamiento x cambia de ítem; antes pertenecía al $\langle q_{i+3}, d_{i+3}|$ y ahora pertenece al $\langle q_{i+4}, d_{i+4}|$. La longitud es invariante.

R22 $set(x = q_{i+2}, d = d_{i+2})$ {

$\ldots \langle q_{i+2}, d_{i+2}| \langle q_{i+3}, d_{i+3}| \ldots \rightarrow$

$\ldots \langle q_{i+2}, d_{i+2}| \langle q_{i+3}, d_{i+3}| \ldots$ }

La regla es invariante pues las respuestas son iguales.

R23 $set(q_{i+2} < x < q_{i+3}, d = d_{i+3})$ {

$\ldots \langle q_{i+2}, d_{i+2}| \ \langle q_{i+3}, d_{i+3}| \ldots \rightarrow$

$\ldots \langle q_{i+2}, d_{i+2}| \ \langle q_{i+3}, d_{i+3}| \ldots$ }

La regla es invariante pues las respuestas del nuevo ítem y del ítem de desplazamiento q_{i+3} son iguales.

R24 $set(x < q_{i+3}, q_{i+2} + 1 = x, d = d_{i+2})$ {

$\ldots \langle q_{i+2}, d_{i+2}| \ \langle q_{i+3}, d_{i+3}| \ldots \rightarrow$

$\ldots \langle x, d_{i+2}|\ \langle q_{i+3}, d_{i+3}| \ldots\ \}$

La regla es invariante pues las respuestas del nuevo ítem y del ítem de desplazamiento q_{i+2} son iguales.

R25 $set(x < q_{i+3}, q_{i+2} + 1 = x, d \neq d_{i+2}, d \neq d_{i+3})$ {

$\ldots \langle q_{i+2}, d_{i+2}|\langle q_{i+3}, d_{i+3}| \ldots \rightarrow$

$\ldots \langle q_{i+2}, d_{i+2}|\langle x, d|\langle q_{i+3}, d_{i+3}| \ldots\ \}$

La longitud de la lista se incrementa en un ítem.

R26 $set(q_{i+2} < x < q_{i+3}, x - 1 > q_{i+2}, d \neq d_{i+3})$ {

$\ldots \langle q_{i+2}, d_{i+2}|\langle q_{i+3}, d_{i+3}| \ldots \rightarrow$

$\ldots \langle q_{i+2}, d_{i+2}|\langle x - 1, d_{i+3}|\langle x, d|\langle q_{i+3}, d_{i+3}| \ldots\ \}$

La longitud de la lista se incrementa en dos ítems.

Estas reglas de gestión de los ítems están siendo revisadas actualmente para mejorar su definición y eficiencia. En el programa que implementa la lista, las reglas están organizadas por bloques y por similitud de las precondiciones, para hacer más eficiente el algoritmo.

Consideramos además una regla especial que inserta el caso nulo o vacío o no válido, y no altera el estado de la lista. Esta regla facilita la gestión al centralizar las comprobaciones en procedimientos de alto nivel, como la agregación de TDO, véase el capítulo 4.

Podemos clasificar las reglas de acuerdo al tamaño de la lista *KBM2L* antes y después de su aplicación.

1. Reglas **Invariantes**: {R0, R2, R4, R6, R7, R10, R12, R14, R15, R16, R21, R22, R23, R24}. El número de ítems es el mismo antes y después de ejecutar estas reglas.

   Además hacemos notar que las reglas {R0, R2, R4, R6, R10, R12, R15, R22, R23}, aquellas que tienen la misma respuesta en el caso nuevo y en la lista inicial, no cambian la lista, ni la longitud ni el contenido. Las consideramos para que el conjunto de reglas sea completo y así no necesitemos de una regla por defecto o nula. La implementación es más robusta de este modo frente a cambios en la gestión.

2. Reglas **Inflacionarias**: {R1, R3, R5, R9, R11, R13, R20, R25, R26}. El número de ítems se incrementa después de ejecutar estas reglas.

3. Reglas **Coalescentes**: {R8, R17, R18, R19}. El número de ítems disminuye después de ejecutar estas reglas.

Hemos decidido exponer todas las reglas, a pesar de ser muchas, porque creemos que nos permite conocer en profundidad la dinámica de la lista. De este modo queda ahora más clara

la sección 2.4 y es más fácil comprender las cuestiones que allí se muestran y en el capítulo 3 quedará más clara la justificación de las heurísticas propuestas para obtener mejores *Bases* para las listas.

Veamos un ejemplo completo con una sencilla tabla almacenada en dos instancias del modelo. También ilustramos el procedimiento de construcción de la lista *KBM2L* descrito en la sección 2.3.2.

Ejemplo 2.15. El esquema tiene cuatro atributos $\{X_0,\ X_1,\ X_2,\ X_3\}$, la *Base* es $[0, 1, 2, 3]$ y todos los dominios son binarios.

Denotamos los valores del dominio como 0 y 1 para cada atributo. El conjunto de modalidades de respuesta del problema es denotado por {A,B,C,D,E}. La primera tabla, véase el Cuadro 2.4, es la instancia de $X_0 = 0$ y la segunda, véase el Cuadro 2.5, la de $X_0 = 1$.

Cuadro 2.4: Instancia de la tabla de $X_0 = 0$

X_0	X_1	X_2	X_3	desplazamiento	respuesta
0	0	0	0	0	A
0	1	0	0	4	A
0	0	1	0	2	A
0	1	1	0	6	A
0	0	0	1	1	B
0	1	0	1	5	B
0	0	1	1	3	C
0	1	1	1	7	C

Ahora ilustramos el procedimiento de construcción incremental de la *KBM2L* (sección 2.3.2). Los Cuadros 2.4 y 2.5 corresponden al primer paso del procedimiento.

El segundo paso, véase el Cuadro 2.6, muestra la traducción de la primera tabla a la lista *KBM2L*. La lista vacía inicial es $\langle 15, -1|$. Luego se lee la instancia de la tabla con $X_0 = 0$ y se introducen una a una las entradas o casos de la tabla en la lista.

No hay respuestas iguales adyacentes en la lista con respecto a la *Base* inicial, véase el Cuadro 2.6, excepto los ocho casos desconocidos desde el desplazamiento 8 al 15. Esta *KBM2L* es, después del segundo paso:

$$\langle 0, A|\ \langle 1, B|\ \langle 2, A|\ \langle 3, C|\ \langle 4, A|\ \ \langle 5, B|\ \langle 6, A|\ \langle 7, C|\ \langle 15, -1|_{[0,1,2,3]}$$

El paso tercero, en el Cuadro 2.7, indica cómo el resto de los resultados parciales es añadido

Cuadro 2.5: Instancia de la tabla de $X_0 = 1$

X_0	X_1	X_2	X_3	desplazamiento	respuesta
1	0	0	0	8	C
1	0	1	0	10	C
1	1	0	0	12	C
1	0	1	1	11	D
1	0	0	1	9	D
1	1	0	1	13	D
1	1	1	0	14	D
1	1	1	1	15	E

Cuadro 2.6: Construcción de la *KBM2L*: segundo paso

caso en el desplazamiento q por la regla $R \rightarrow$ KBM2L
$((0,0,0,0),A)$ en desplazamiento 0 por la regla 1 $\rightarrow$ $\langle 0,A\vert\langle 15,-1\vert$
$((0,1,0,0),A)$ en 4 por R26 $\rightarrow$ $\langle 0,A\vert\langle 3,-1\vert\langle 4,A\vert\langle 15,-1\vert$
$((0,0,1,0),A)$ en 2 por R20 $\rightarrow$ $\langle 0,A\vert\langle 1,-1\vert\langle 2,A\vert\langle 3,-1\vert\langle 4,A\vert\ \langle 15,-1\vert$
$((0,1,1,0),A)$ en 6 por R20 $\rightarrow$ $\langle 0,A\vert\langle 1,-1\vert\langle 2,A\vert\langle 3,-1\vert\langle 4,A\vert\ \langle 5,-1\vert\langle 6,A\vert\langle 15,-1\vert$
$((0,0,0,1),B)$ en 1 por R16 $\rightarrow$ $\langle 0,A\vert\langle 1,B\vert\langle 2,A\vert\langle 3,-1\vert\langle 4,A\vert\ \langle 5,-1\vert\langle 6,A\vert\langle 15,-1\vert$
$((0,1,0,1),B)$ en 5 por R16 $\rightarrow$ $\langle 0,A\vert\langle 1,B\vert\langle 2,A\vert\langle 3,-1\vert\langle 4,A\vert\ \langle 5,B\vert\langle 6,A\vert\langle 15,-1\vert$
$((0,0,1,1),C)$ en 3 por R16 $\rightarrow$ $\langle 0,A\vert\langle 1,B\vert\langle 2,A\vert\langle 3,C\vert\langle 4,A\vert\ \langle 5,B\vert\langle 6,A\vert\langle 15,-1\vert$
$((0,1,1,1),C)$ en 7 por R24 $\rightarrow$ $\langle 0,A\vert\langle 1,B\vert\langle 2,A\vert\langle 3,C\vert\langle 4,A\vert\ \langle 5,B\vert\langle 6,A\vert\langle 7,C\vert\langle 15,-1\vert$

a la lista anterior. En este contexto se pone en funcionamiento el mecanismo de aprendizaje. Éste puede ser *simple*, si no intentamos optimizar la lista antes de procesar la siguiente instan-

cia, u óptimo, si optimizamos la lista mientras el nuevo conocimiento es introducido en la lista. Aquí realizamos aprendizaje óptimo. La permutación de los atributos que muestra el mínimo número de ítems en la lista conduce a la *Base* $[0, 3, 2, 1]$. En este pequeño ejemplo podemos examinar *todas* las posibles soluciones, $(4! = 24)$. La lista óptima es: $\langle 3, A|\langle 5, B|\langle 7, C|\langle 15, -1|_{[0,3,2,1]}$. Observamos el espacio ahorrado en esta configuración inducida por la nueva *Base*: de 9 ítems iniciales a solamente 4. Indicamos como subíndice la *Base* de la lista porque ahora estamos trabajando en dos *Bases* diferentes simultáneamente, la de las instancias y la de la lista parcial optimizada. Ahora accedemos a la siguiente tabla y continuamos, véase el Cuadro 2.7, insertando casos sobre la lista optimizada de 4 ítems.

Cuadro 2.7: Construcción de la *KBM2L*: tercer paso

caso en el desplazamiento q por la regla $R \rightarrow$ KBM2L
$((1,0,0,0), C)$ en desplazamientos $8_{[0,1,2,3]}$ y $8_{[0,3,2,1]}$ por la regla 20 $\rightarrow$ $\langle 3, A|\langle 5, B|\langle 8, C|\langle 15, -1|_{[0,3,2,1]}$
$((1,0,1,0), C)$ en $10_{[0,1,2,3]}$ y $10_{[0,3,2,1]}$ por R26 $\rightarrow$ $\langle 3, A|\langle 5, B|\langle 8, C|\langle 9, -1|\langle 10, C|\langle 15, -1|_{[0,3,2,1]}$
$((1,1,0,0), C)$ en $12_{[0,1,2,3]}$ y $9_{[0,3,2,1]}$ por R18 $\rightarrow$ $\langle 3, A|\langle 5, B|\langle 10, C|\langle 15, -1|_{[0,3,2,1]}$
$((1,0,1,1), D)$ en $11_{[0,1,2,3]}$ y $14_{[0,3,2,1]}$ por R20 $\rightarrow$ $\langle 3, A|\langle 5, B|\langle 10, C|\langle 13, -1|\langle 14, D|\langle 15, -1|_{[0,3,2,1]}$
$((1,0,1,1), D)$ en $9_{[0,1,2,3]}$ y $12_{[0,3,2,1]}$ por R20 $\rightarrow$ $\langle 3, A|\langle 5, B|\langle 10, C|\langle 11, -1|\langle 12, D|\langle 13, -1|\langle 14, D|\langle 15, -1|_{[0,3,2,1]}$
$((1,1,0,1), D)$ en $13_{[0,1,2,3]}$ y $13_{[0,3,2,1]}$ por R18 $\rightarrow$ $\langle 3, A|\langle 5, B|\langle 10, C|\langle 11, -1|\langle 14, D|\langle 15, -1|_{[0,3,2,1]}$
$((1,1,1,0), D)$ en $14_{[0,1,2,3]}$ y $11_{[0,3,2,1]}$ por R19 $\rightarrow$ $\langle 3, A|\langle 5, B|\langle 10, C|\langle 14, D|\langle 15, -1|_{[0,3,2,1]}$
$((1,1,1,1), E)$ en $15_{[0,1,2,3]}$ y $15_{[0,3,2,1]}$ por R16 $\rightarrow$ $\langle 3, A|\langle 5, B|\langle 10, C|\langle 14, D|\langle 15, E|_{[0,3,2,1]}$

El último paso es la optimización de la lista final. En este ejemplo podemos ver que hay cinco alternativas agrupadas en cinco ítems y no es posible mejorar la configuración. En general, no es tan fácil ver y caracterizar la *Base* óptima.

Observamos las diferencias entre la configuración óptima y la *Base* inicial:

Base Inicial
$\langle 0, A\vert\ \langle 1, B\vert\ \langle 2, A\vert\ \langle 3, C\vert\ \langle 4, A\vert$
$\langle 5, B\vert\ \langle 6, A\vert\ \langle 8, C\vert\ \langle 9, D\vert\ \langle 10, C\vert$
$\langle 11, D\vert\ \langle 12, C\vert\ \langle 14, D\vert\ \langle 15, E\vert_{[0,1,2,3]}$
Base Óptima
$\langle 3, A\vert\langle 5, B\vert\langle 10, C\vert\langle 14, D\vert\langle 15, E\vert_{[0,3,2,1]}$

□

Capítulo 3

Optimización de la Lista *KBM2L*

Definimos, en la sección 3.1, el problema de optimización como la búsqueda en el conjunto de configuraciones de la *KBM2L*. La búsqueda consiste en transformar la *KBM2L* mediante transposiciones de las dimensiones de la MM asociada. En este capítulo planteamos además los problemas que aparecen al tratar de configurar una lista con longitud mínima.

Para ello, a partir de la tabla, o parte de una tabla, se construye una *KBM2L* y se proponen otras en nuevas *Bases* a las que se copia el contenido de la original. Si la nueva estructura es de menor tamaño, se guarda y continúa el procedimiento de optimización. La necesidad de un procedimiento de copia de contenidos entre estructuras *KBM2L* en diferentes *Bases* surge al optimizar una lista y es computacionalmente muy complejo.

La sección 3.2 muestra el problema del $XBase$ de complejidad exponencial y proponemos diferentes estrategias para manipular la estructura del espacio de representación de las tablas. Las propuestas van encaminadas a aprovechar la estructura de la lista y al desarrollo de algoritmos de aprendizaje que permitan acelerar el $XBase$, es decir, el procedimiento de copia.

Introducimos la operación de unión de listas que permite implementar el aprendizaje de contenidos a partir de fragmentos de las tablas, por ejemplo, instancias parciales de la evaluación de un modelo.

En la sección 3.3 profundizamos en el problema de la búsqueda de la *Base* óptima y en los elementos de información para fundamentar las heurísticas propuestas. Dichos elementos son características de una lista que se pueden calcular con complejidad aceptable y dan información sobre su longitud y el nivel de coalescencia. Proponemos algunas soluciones al problema de búsqueda del óptimo y al problema de cambio de configuración de una lista.

Proponemos también una estrategia combinada de búsqueda local y global mediante algoritmos de entorno variable (sección 3.4) y genéticos (sección 3.5).

En la sección 3.6 realizamos un conjunto de experimentos que ilustran las posibilidades, limitaciones y principales características de los algoritmos propuestos.

El capítulo termina con la sección 3.7 donde mostramos algunas estructuras conocidas en ciencias de la computación para manejar tablas y matrices de gran tamaño. Discutimos las semejanzas y diferencias entre las listas *KBM2L* y estas estructuras respecto de la capacidad de representación e inferencia del conocimiento.

3.1. Definición de Lista *KBM2L* Óptima

Una *KBM2L* de una tabla tiene asociado un cierto espacio de almacenamiento y organización de la información. Cuando tenemos la misma información almacenada pero en el menor espacio y con la mayor organización decimos que la *KBM2L* está optimizada. Como el problema debe ser semánticamente el mismo, es decir, el mismo conjunto de atributos original, no podemos buscar cualquier posible forma de almacenamiento de la información de nuestra tabla. Como ya indicamos en el capítulo 2, nuestra búsqueda se va a restringir a las posibles permutaciones de los atributos. En tablas con δ atributos, debemos considerar $\delta!$ posibles soluciones. A causa de la discretización de los dominios de los atributos, el orden de los dominios puede también ser permutado, incrementando el tamaño del espacio de búsqueda de soluciones a $\delta! \prod_{i=0}^{\delta-1} \delta_i!$. Actualmente hemos implementado esta posibilidad con caracter experimental pero nos restringiremos aquí a las $\delta!$ *Bases* que conservan el orden del dominio de cada atributo. Así, los *XBase* son respecto de un esquema de referencia fijo. Pensamos que el orden del dominio es un aspecto sintáctico, pero que tiene un cierto componente semántico en el esquema de atributos de la tabla, especialmente si los valores definen una escala ordinal o métrica.

Alterar el esquema supone un cambio en la representación semántica, lo cual es más complejo que el *XBase*, que repercute tan sólo en la sintaxis de la tabla. Teoricamente podríamos considerar *XBases* más generales que alteran el esquema. Ejemplos de *XBase* más generales se detallan en (Fernández del Pozo y Bielza, 2002b), donde a partir de un esquema inicial se deducen descomposiciones y composiciones de atributos que pueden en principio mejorar la representación del conocimiento. Si alteramos el esquema, la permutación en el dominio de atributos no es significativamente diferente de la permutación en la *Base*, ya que algunos de los atributos del nuevo esquema son valores y/o combinaciones de valores del esquema inicial. La cuestión es si la dificultad de alterar la semántica del esquema y la complejidad de la búsqueda se compensa con la representación interna más eficiente de la BC.

El problema que nos planteamos es encontrar una *Base* de la MM cuya lista *KBM2L* tenga

el menor número de ítems. La *Base* canónica tiene el orden $[0, 1, 2, \ldots, \delta - 1]$, y así, tal como hemos dicho anteriormente hay $\delta!$ *Bases*. Es un conjunto finito, pero muy grande y no es posible enumerarlo en la práctica. Podemos generar *Bases* aleatoriamente (usando una estrategia multicomienzo) e intentar mejoras por medio de $XBase$.

Por consiguiente, el problema de encontrar la lista *KBM2L* con mínimo número de ítems es de optimización combinatoria. La estrategia general de búsqueda de una buena *Base* trata de mejorar la *Base* actual generando otra de acuerdo con alguna heurística. Si la nueva lista es mejor que la primera (esto es, requiere un espacio menor) se guarda, y el proceso itera hacia una nueva mejora.

Hay aún un problema: la *mejora* de la solución no se puede verificar en tiempo polinómico si se trata de una lista completa (donde todas las entradas de la tabla asociada, o la mayoría, son casos conocidos), pues en cada paso debe ser resuelto un problema de complejidad exponencial.

Por tanto, por una parte, el tamaño del espacio de búsqueda es factorial, y por otra, el $XBase$ tiene la complejidad exponencial de la tabla. La búsqueda propiamente dicha se aborda siguiendo dos estrategias: global, mediante un algoritmo genético que describiremos en la sección 3.5 y local mediante un algoritmo de entorno variable que describiremos en la sección 3.4. En ambos casos hay que caracterizar las *Bases* o las configuraciones de las listas, es decir, calcular la longitud de la lista o al menos acotarla.

Recordamos del capítulo anterior el método para construir la lista *KBM2L*. Consiste en leer las entradas de la tabla con un orden inicial original o *Base* canónica representada por una *Base* de permutaciones de los atributos identidad en una estructura sin información:

- $\langle W_0 * \delta_0 - 1, unKB|$ en notación de desplazamiento, o
- $\langle (\delta_0 - 1, \delta_1 - 1, \ldots, \delta_{\delta-1} - 1), unKB|$ en notación de índice.

La *Base* determina en gran medida la coalescencia y tamaño de la *KBM2L*. Más tarde se copia la información a otra estructura compatible con una permutación de la *Base*. En general no conocemos toda la tabla, si es muy grande, pero ignoremos esta cuestión por el momento.

Una vez que hemos construido la lista y sabemos cómo gestionar sus casos mediante ítems conforme a las reglas, véase la sección 2.6, la lista debe ser optimizada, minimizando el número de ítems. La lista mejor (más eficaz como BC) es la que tiene menos ítems para representar el contenido de la tabla. A la *Base* que determina una lista con mínimo número de ítems la llamaremos óptima.

Veremos que no es sólo importante disminuir los requisitos de memoria (más casos de la tabla por ítem de la lista) sino también mostrar aspectos tales como afinidad entre atributos

y relevancia/irrelevancia con respecto a la respuesta (explicación). Y ambas características las proporcionan de modo eficiente las *Bases* óptimas.

La lista óptima, así definida, no es única, pues diferentes *Bases* pueden dar la misma longitud de la lista. El hecho de que existan múltiples *Bases* óptimas está relacionado con la falta de ajuste entre el esquema y el contenido de la tabla. Dicho de otra forma, el esquema de la tabla no es la mejor representación del conocimiento. La idea es que atributos redundantes o valores de atributos irrelevantes implican un espacio de representación mayor (exponencialmente) y facilitan la aparición de configuraciones equivalentes. Esto es lo que tratamos de detectar.

Como casos especiales, para la lista vacía y para la lista completamente densa, todas las *Bases* son óptimas. El primer caso se corresponde con ausencia de conocimiento, con longitud 1, la menor. El segundo caso es la propia función del desplazamiento, con longitud máxima.

Ejemplo 3.1. Sea el esquema $\{A_0, A_1, A_2\}$ con dominio $\{0, 1\}$, $\forall A_i$. Las listas $\langle 7,\ unKB|$, y $\langle 0, r_0|\ \langle 1, r_1|\ \langle 2, r_2|\ \langle 3, r_3|\ \langle 4, r_4|\ \langle 5, r_5|\ \langle 6, r_6|\ \langle 7, r_7|$, son vacía y completamente densa, respectivamente. □

El problema de búsqueda (encontrar la *Base* óptima que minimiza la lista) es NP-completo al igual que encontrar el árbol de clasificación mínimo equivalente a un conjunto de fórmulas lógicas proposicionales en FND (Zantema y Bodlaender, 2000), una de las representaciones del conocimiento semejantes a la lista *KBM2L*. La tabla puede verse como un conjunto de fórmulas lógicas proposicionales en FND.

La idea de minimizar el número de ítems puede ser planteada de manera global o parcial enfocando sobre algunas familias de ítems, es decir, sobre un subconjunto de las modalidades del contenido o respuesta de la tabla. En particular, sobre una de las modalidades, implica optimizar el almacenamiento de una familia. Entonces, tal vez globalmente la lista no sea mínima, pero la modalidad objeto es almacenada con mayor eficiencia. La utilidad de la minimización selectiva de una modalidad fue mencionada en el capítulo anterior en la sección 2.4 y se trató mediante ejemplos de la relevancia y explicación en tablas de decisión (sección 2.4.2).

Aquellos $XBase$ que determinan reordenaciones de los casos de los ítems de la misma familia no cambian la longitud de la lista. En concreto, decimos que las *Bases* son **equivalentes** si las listas son de igual longitud. Decidir la equivalencia entre *Bases* tiene una complejidad muy alta, pero refutar la equivalencia, en media, es mucho menos complejo. En la sección 3.2 y siguientes justificaremos ambas afirmaciones. En (Zantema, 1998) se estudia la equivalencia entre árboles de clasificación y se determina que la complejidad es exponencial (veremos al final del capítulo que es una representación semejante a la lista *KBM2L*). Dos árboles de clasificación son equivalentes si clasifican en las mismas clases los casos posibles. Decimos que las listas son

equivalentes simplemente si almacenan idénticos casos de la misma tabla. Análogamente, decidir la equivalencia de las listas tiene la complejidad de la tabla, exponencial, y refutar la equivalencia en el caso peor también.

Ejemplo 3.2. Este ejemplo corresponde a una tabla con un espacio de representación de 2^8 casos. Veamos el espectro de la lista que muestra cómo los casos se agrupan mediante un $XBase$. En la Figura 3.1, la respuesta (color), con cuatro posibles modalidades (1, 2, 3 y 4), es representada frente al desplazamiento o índice en su respectiva $Base$ (eje X). La idea de fragmentación de ítems con respecto de la $Base$ es evidente cuando observamos el espectro. Seguro que si se encuentra un patrón que se repite como el del primer espectro podemos asegurar que hay una $Base$ mejor. El problema es que en tablas de dimensionalidad muy alta pueden superponerse muchos patrones dando lugar a un perfil desordenado.

El $XBase$ afecta a la unión/fragmentación de los ítems. Las listas asociadas a cada espectro tienen la misma información distribuida en 256 casos, pero la lista óptima (la última) requiere menos espacio de memoria para almacenarlo, véanse los Cuadros 3.1 y 3.2.

Cuadro 3.1: Lista *KBM2L* no óptima de la Figura 3.1

Base Inicial: [0,1,2,3,4,5,6,7] <((índice), respuesta)|	Desplazamientos de índices (I_{inf}, I_{sup})	64 ítems nº casos
⟨(0,0,0,0,1,0,0,1), 1|	(0, 9)	10
⟨(0,0,0,0,1,0,1,1), 2|	(10, 11)	2
⟨(0,0,0,0,1,1,0,1), 1|	(12, 130)	2
⟨(0,0,0,1,0,1,0,1), 2|	(14, 21)	8
⟨(0,0,0,1,1,0,0,1), 3|	(22, 25)	4
⟨(0,0,0,1,1,0,1,1), 4|	(26, 27)	2
...	...	...
⟨(1,1,1,1,1,1,1,1), 4|	(254, 255)	2

A continuación mostramos cómo construir una estructura *KBM2L* óptima.

El resto del capítulo trata, a lo largo de cuatro secciones, el $XBase$ y las heurísticas que permiten optimizar la lista y responde a: ¿cúal es el orden del esquema de atributos o $Base$ que minimiza el número de ítems almacenados para una matriz dada?

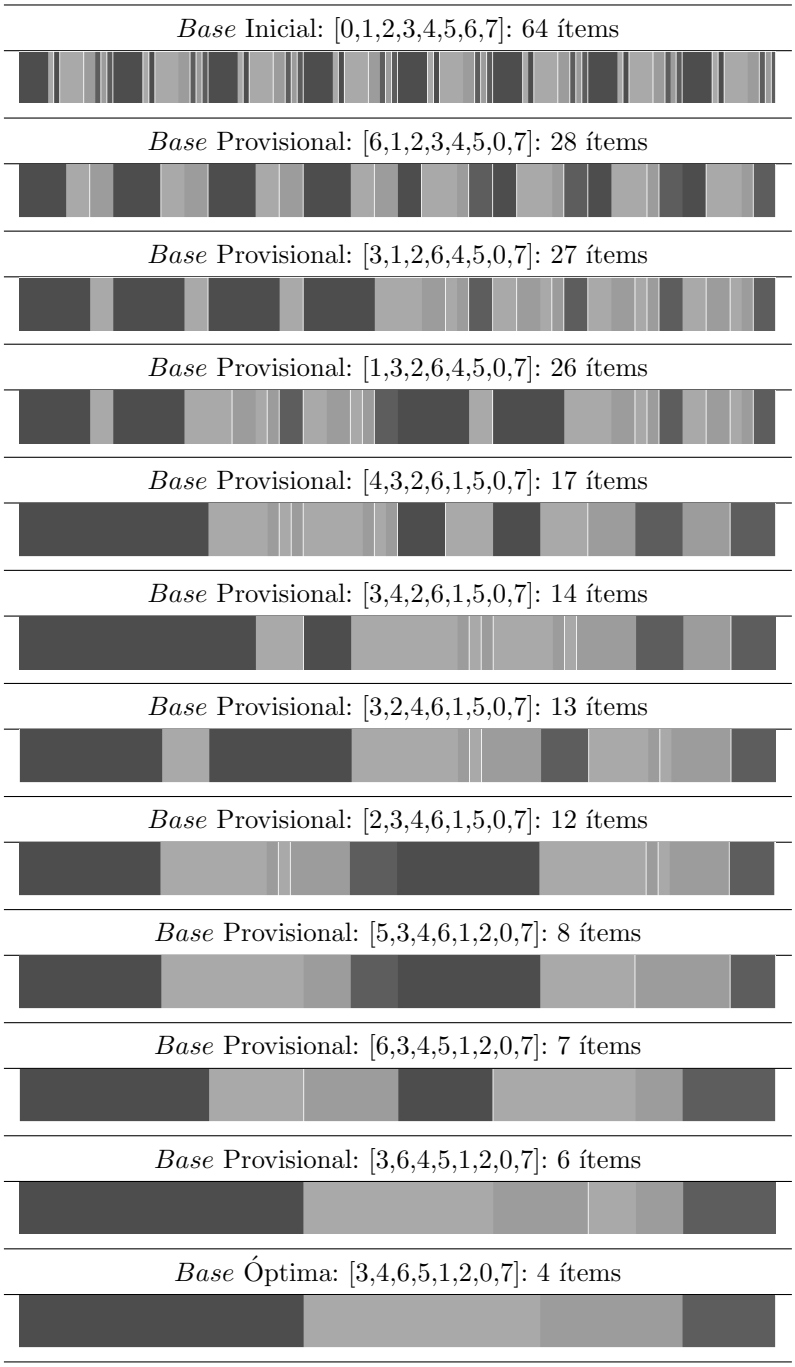

Figura 3.1: Espectro de lista *KBM2L*

3.2. Cambio de Base de Listas *KBM2L*: Copias

Llamamos lista inicial a la lista objeto del $XBase$, y lista nueva al resultado en la nueva $Base$. La lista puede estar en una $Base$ arbitraria y la podemos transponer modificando los índices del contenido de la tabla en la $Base$ inicial a otra $Base$.

Cuadro 3.2: Lista *KBM2L* óptima de la Figura 3.1

Base Óptima:	Desplazamientos de	4 ítems
[3,4,6,5,1,2,0,7]	índices (I_{inf}, I_{sup})	nº casos
<((índice), respuesta)|		
⟨(0,1,0,1,1,1,1,1), 1|	(0, 95)	96
⟨(1,0,1,0,1,1,1,1), 2|	(96, 175)	80
⟨(1,1,0,1,1,1,1,1), 3|	(176, 223)	48
⟨(1,1,1,1,1,1,1,1), 4|	(224, 255)	32

□

Realizar un $XBase$ a una lista es cambiar de *Base* todos los casos de la lista inicial a la nueva *Base*. Para ello debemos transponer el índice de *cada* caso: reescribir el orden de las componentes conforme a la *Base* nueva. Consideramos el algoritmo *B1BEB2* que cambia la *Base* de la lista *KBM2L* entre dos *Bases* cualesquiera *B1* y *B2* utilizando la *Base* del esquema o canónica *BE*. Los algoritmos *B1BE* y *BEB2* implementan el $XBase$ canónica desde una *Base* cualquiera y viceversa. El código que mostramos es una simplificación del código real donde se ha eliminado el manejo de excepciones y se supone que no hay errores en los parámetros de entrada. Las referencias de los objetos *Integer* y *Vector* están en (Sun Microsistems J.D.K., 2006) y del objeto *Base30* en http://www.dia.fi.upm.es/ jafernan/kbm32docs/index.html.

Algoritmo de $XBase$**.** B1BEB2: transposición general de índices para *Bases* naturales simples y extendidas: $B1 \rightarrow BE \rightarrow B2$. Parámetros de entrada: $B1$ de clase base30, I índice en la *Base* de $B1$ de clase Vector, $B2$ de clase base30. Salida: V Vector índice en la *Base* de $B2$.

```
B1BEB2( base30 B1, Vector I, base30 B2) {
//si las Bases son la misma no hay que hacer nada
if ( eqB( B1, B2)) return I;
//el número de atributos de la base B1
int b1sz = B1.size();
//reserva de espacio para el índice transpuesto
Vector U = new Vector( b1sz);
//transposición a la BE desde la Base inicial
for( int k = 0; k < b1sz; k++) {
```

```
      //el k-ésimo atributo de la Base B1 es el kB1-ésimo atributo de la BE
      int kB1 = B1.bas( k);
      //el k-ésimo atributo del I toma el valor iv
      int iv = ((Integer)I.get( k)).intValue();
      //B1BE, iv pasa de la posición k de I a la kB1 de U
      U.set( kB1, new Integer( iv));
      }//for
 //el número de atributos de la Base B2
 int b2sz = B2.size();
 //reserva de espacio para el índice transpuesto
 Vector V = new Vector( b2sz);
 //transposición a la Base nueva desde BE
 for( int k = 0; k < b2sz; k++) {
      //el k-ésimo atributo de la Base B2 es el kB2-ésimo atributo de la BE
      int kB2 = B2.bas( k);
      //el k-ésimo atributo del U toma el valor iv
      int iv = ((Integer)U.get( k)).intValue();
      //BEB2, iv pasa de la posición k de I a la kB1 de U
      V.set( k, new Integer( iv));
      }//for
 return V;//el índice ahora tiene sus valores en correspondencia con la Base B2
 }//B1BEB2
```

Un ejemplo de transposición de índices se mostró en la Figura 2.2.

Nótese que los índices transpuestos, dependiendo de las *Bases* inicial y nueva y de los dominios de los atributos del esquema, son variables o son invariantes. Así, el $XBase$ simple mantiene al índice $(0, 0, \ldots, 0)$ (CERO) invariante, no cambia nunca, y al índice $(\delta_0 - 1, \delta_1 - 1, \ldots, \delta_{\delta-1} - 1)$ (ÚLTIMO) también, siempre que el cardinal del dominio sea igual para todos los atributos.

Finalmente afirmamos que dos matrices P y Q almacenan la misma información si existe una permutación de índices $t(i)$ que lleva una a la otra: $P_{t(i)} = Q$.

3.2.1. Cambio de Base e Invarianza

En general, un índice es constante o invariante en un $XBase\ B1 \rightarrow B2$ si todos los valores de las posiciones del índice que permutan en dicho $XBase$ son iguales. Es decir, el índice es formalmente el mismo en ambas permutaciones $B1 \rightarrow B2$ y $B2 \rightarrow B1$. El $XBase$ es genérico y consideramos que permuta cualquier número de atributos.

Mostramos un algoritmo que comprueba la invarianza de casos e ítems:

Test de invarianza de un índice. El test lo denominamos *fix*. Parámetros de entrada: $B1$ $Base$ inicial de clase base30, I índice de un caso o parte fija de un ítem de clase Vector, $B2$ $Base$ nueva de clase base30. Salida: invariante de tipo *boolean* (true/false).

```
fix( base30 B1, Vector I, base30 B2) {
boolean invariante = true;
int i = 0, j = 0;
for( i = 0; ( i <I.size() AND invariante); i++)
     if ( B1.att ( i).equals( B2.att( i))) {
          if ( B1.bas( i) != B2.bas( i)) {//i permuta
              j = B2.indexOfbas( B1.bas( i));
              invariante = invariante AND
              ( ((Integer)I.get( i)).intValue() == ((Integer)I.get( j)).intValue());
              }//es coord movil
          }//if-
     else { return false; }
return invariante;
}//fix
```

Si los valores de las posiciones que permutan son iguales no es sintácticamente posible distinguir si se ha aplicado la transposición o no. Un punto fijo en la transposición de atributos requiere que los valores de las posiciones que se intercambian sean idénticos.

En la Figura 3.2 podemos ver en las flechas horizontales los índices invariantes de un ejemplo sencillo. Observando el ejemplo de la mencionada figura podemos pensar que los índices invariantes tienen desplazamiento invariante pero, en general, es falso salvo que todos los atributos tengan igual cardinal del dominio.

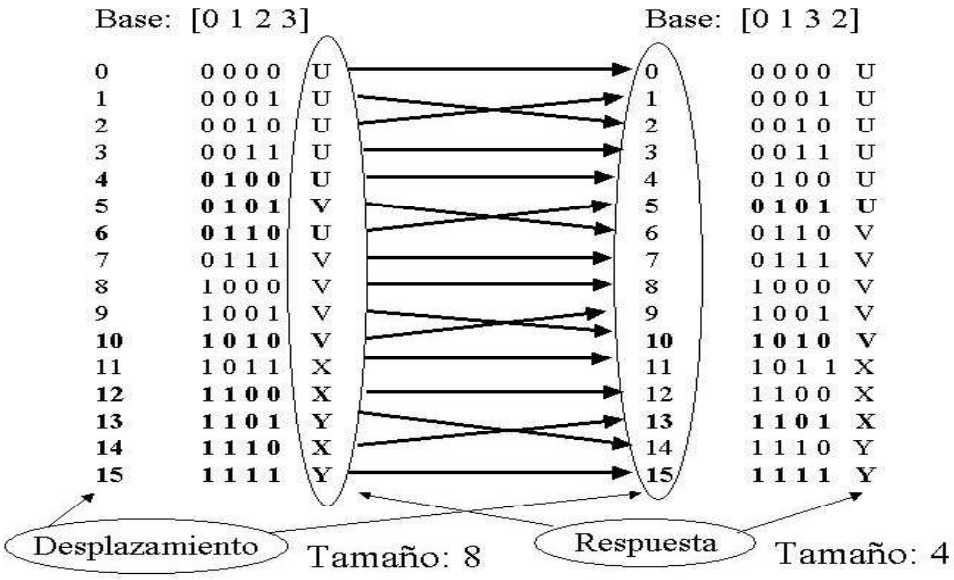

Figura 3.2: *XBase* de listas *KBM2L*

Ejemplo 3.3. Sean $[0, 4, 2, 3, 1]$ y $[0, 4, 3, 2, 1]$ dos *Bases*. Sean $(u_0^0, u_4^1, u_2^2, u_3^3, u_1^4)$ y $(v_0^0, v_4^1, v_3^2, v_2^3, v_1^4)$ los respectivos índices. La invarianza es $u^k = v^k$, componente a componente. En este ejemplo deben ser $u_2^2 = v_3^2$ y $u_3^3 = v_2^3$. Así, supuesto un dominio binario para todos los atributos, $(1, 0, 0, 0, 0)$ es invariante y $(1, 0, 1, 0, 0)$ no es invariante pues se transforma en $(1, 0, 0, 1, 0)$.

El vector de pesos es $\overrightarrow{w} = (16, 8, 4, 2, 1)$ en todas las *Bases* y el desplazamiento de un índice invariante sí es igual en cualquier *Base*, pues los atributos que permutan toman el mismo valor del dominio respectivo.

En otros casos con dominio no homogéneo no tiene por qué ocurrir esto. Por ejemplo, con dominios de 2, 3, 4, 5 y 6 valores respectivamente, para los atributos de la primera *Base*, tenemos el vector de pesos $\overrightarrow{w}$ = (360, 120, 30, 6, 1). La otra *Base* tiene el siguiente vector de pesos $\overrightarrow{w}$ = (360, 120, 24, 6, 1). Ahora vemos que puede ocurrir que algunos índices invariantes tengan desplazamiento invariante, como el $(1, 0, 0, 0, 0)$ cuyo desplazamiento es 360 en ambas *Bases*, pero $(0, 0, 2, 2, 0)$ tiene desplazamiento 72 en la primera y 60 en la segunda. □

Una última consideración sobre la invarianza de los índices bajo transposiciones es la caracterización de **invarianza de un ítem**. Si es fácil caracterizar la invarianza para un ítem dado el *XBase*, entonces el ítem puede transponerse, con todos sus casos, en una sola operación, véase la sección 3.2.2.

Es muy interesante relacionar la invarianza de un caso (un índice) con la de un ítem. De este modo podemos explotar la organización de los casos en la lista para implementar procedimientos

de $XBase$ eficientes. La cuestión importante es que sabemos que no se conserva en general el desplazamiento y éste determina la proximidad y adyacencia de los casos. En definitiva, determina, dada la respuesta, si los casos están agrupados en un ítem. Nos preguntamos si podemos determinar la invarianza de un conjunto de casos que involucren a ítems, en sentido fuerte o débil, a partir de la invarianza de sus límites o de la parte fija. Veamos un ejemplo.

Ejemplo 3.4. Consideramos el esquema de atributos {A,B,C} y los respectivos dominios con 2, 3, y 4 valores. Indicamos el desplazamiento q y denotamos los valores de la respuesta R con X e Y. Sea la $Base$ $[A, B, C]$ con índice (i_0, i_1, i_2) y la $Base$ $[B, A, C]$ con índice (i_1, i_0, i_2). En la tabla siguiente señalamos en negrita los índices y desplazamientos invariantes en el $XBase$. Puede observarse que hay cuatro clases de casos:

1. invariantes respecto del índice y del desplazamiento,
2. invariantes respecto del índice,
3. invariantes respecto del desplazamiento y
4. no invariantes.

q	A	B	C	R	q	A	B	C	R	q	B	A	C	R	q	B	A	C	R
0	**0**	**0**	**0**	X	12	1	0	0	Y	**0**	**0**	**0**	**0**	X	12	**1**	**1**	**0**	Y
1	**0**	**0**	**1**	X	13	1	0	1	Y	**1**	**0**	**0**	**1**	X	13	**1**	**1**	**1**	Y
2	**0**	**0**	**2**	X	14	1	0	2	Y	**2**	**0**	**0**	**2**	X	14	**1**	**1**	**2**	X
3	**0**	**0**	**3**	X	15	1	0	3	Y	**3**	**0**	**0**	**3**	X	15	**1**	**1**	**3**	X
4	0	1	0	X	16	**1**	**1**	**0**	Y	4	0	1	0	Y	16	2	0	0	X
5	0	1	1	X	17	**1**	**1**	**1**	Y	5	0	1	1	Y	17	2	0	1	X
6	0	1	2	Y	18	**1**	**1**	**2**	X	6	0	1	2	Y	18	2	0	2	X
7	0	1	3	Y	19	**1**	**1**	**3**	X	7	0	1	3	Y	19	2	0	3	X
8	0	2	0	X	**20**	1	2	0	X	8	1	0	0	X	**20**	2	1	0	X
9	0	2	1	X	**21**	1	2	1	X	9	1	0	1	X	**21**	2	1	1	X
10	0	2	2	X	**22**	1	2	2	Y	10	1	0	2	Y	**22**	2	1	2	Y
11	0	2	3	X	**23**	1	2	3	Y	11	1	0	3	Y	**23**	2	1	3	Y

En cuanto a los ítems, éstos dependen de la adyacencia de la respuesta.

Mediante la $Base$ $[A, B, C]$ almacenamos la lista *KBM2L* con 6 ítems:

$$\langle 5, X| \ \langle 7, Y| \ \langle 11, X| \ \langle 17, Y| \ \langle 21, X| \ \langle 23, Y|.$$

Mediante la $Base$ $[B, A, C]$ almacenamos la lista *KBM2L* con 6 ítems:

$$\langle 3, X| \ \langle 7, Y| \ \langle 9, X| \ \langle 13, Y| \ \langle 21, X| \ \langle 23, Y|.$$

En el ejemplo vemos que los índices invariantes adyacentes son adyacentes en la $Base$ nueva. Es decir, los desplazamientos correspondientes, aunque no todos sean invariantes, son consecutivos en ambas $Bases$. Por tanto, el bloque de casos con desplazamientos {0, 1, 2, 3} en la $Base$ $[A, B, C]$ que constituye parte del ítem $\langle 5, X|$ se mueve de forma solidaria con la transposición y constituye el ítem $\langle 3, X|$ en la $Base$ $[B, A, C]$. Los desplazamientos y los índices son invariantes. De igual modo el bloque de casos con desplazamientos {16, 17, 18, 19} en $[A, B, C]$ se mueve a los desplazamientos {12, 13, 14, 15} en $[B, A, C]$. Los desplazamientos no son invariantes pero sí los índices correspondientes.

El ítem $\langle 23, Y|_{[A,B,C]}$ es invariante pues los índices de sus casos no son invariantes pero sí sus desplazamientos límite. El ítem $\langle 3, X|_{[B,A,C]}$ es invariante, la parte fija del índice lo es, en el $XBase$ $[B, A, C] \rightarrow [A, B, C]$. Si consideramos el $XBase$ de los casos, la invarianza del índice es en ambos sentidos, como se muestra en el test de invarianza de un índice, pero si consideramos el $XBase$ de los ítems no tiene por qué verificarse en ambos sentidos. De hecho, al cambiar de $Base$, ítems de igual respuesta pueden ser adyacentes y constituir un ítem con la unión de todos los casos. Esto ocurre con $\langle 9, X|_{[B,A,C]}$ (que no es invariante) y $\langle 3, X|_{[B,A,C]}$ que se transforman en $\langle 5, X|_{[A,B,C]}$.

En el $XBase$ $[A, B, C] \rightarrow [B, A, C]$, el ítem $\langle 7, Y|$ se desplaza en bloque pero no conserva el desplazamiento, ni los índices son invariantes. □

Como conclusión vemos que el $XBase$ define rangos del desplazamiento que clasifican los casos en alguna de las cuatro clases mencionadas al comienzo del ejemplo 3.4. Dichos rangos constituyen bloques de casos que son puntos fijos de la aplicación que asocia los casos iguales en ambas listas cuando el desplazamiento es invariante, pero no si lo es sólo el índice. Vemos claramente que la invarianza es interesante, respecto del desplazamiento o del índice, pues conserva la adyacencia que define los ítems.

1. Si la parte fija de un ítem es invariante, entonces todos los casos son invariantes y podemos hacer el $XBase$ en bloque.

2. Si la invarianza es de desplazamiento, el bloque no se mueve en la proyección unidimensional de la simetría que define la permutación de dimensiones del $XBase$.

3. Si la invarianza es sólo del índice, el bloque se mueve, pero estos casos invariantes, si son inicialmente adyacentes, lo son en la nueva lista.

3.2.2. Copia de Listas *KBM2L*

El *XBase* es una operación necesaria para optimizar las listas. Al menos hay que realizarlo una vez, supuesto que estemos en una *Base* y encontremos una mejor, con menos ítems. El *XBase* supone copiar la información de la lista en la *Base* inicial a una lista vacía configurada en la *Base* nueva, véase la Figura 3.2. El problema es que la información que almacenamos en la lista, con independencia del número de ítems (unos cientos o miles), puede ser enorme. El número de casos que almacena una lista completa, todos los ítems con conocimiento, es el producto del cardinal de los dominios de los atributos del esquema: el cardinal de la tabla.

Para realizar la copia asociada a un *XBase* se deben seguir estrategias heurísticas. Consideramos a la lista inicial que contiene la información de la BC y a la lista vacía en la nueva base como agentes de la copia.

Adoptamos una perspectiva de listas–agentes: una o varias fuentes de conocimiento (tablas) y varias listas que aprenden, de oferta o a demanda, de dichas fuentes mediante ciclos de aprendizaje y examen. Bajo esta perspectiva proponemos tres estrategias de copia, según qué agente tome la iniciativa en el proceso (oferta/demanda), que el proceso tenga en cuenta la invarianza (estándar/rápida) y que el resultado sea exacto o aproximado (total/parcial). Cada estrategia tiene dos modos de operación excluyentes pero podemos combinarlos dando lugar a ocho escenarios posibles.

Copia de Oferta y a Demanda

Según el agente que tome la iniciativa en la copia distinguimos dos tipos de copia de casos entre listas: copia de oferta y copia a demanda.

La copia de oferta consiste en recorrer la lista inicial (total o parcialmente) expandiendo o desarrollando los ítems e ir ofertando los casos a la lista nueva con índice transpuesto. Véase en la Figura 3.2 el sentido de las flechas.

La copia a demanda consiste en solicitar a la lista inicial un caso de la lista nueva, transponiendo el índice a la *Base* inicial para identificar la respuesta correspondiente.

La copia de oferta surge de forma más natural como forma de completar la estructura de la lista nueva, que es inicializada sin conocimiento. La copia a demanda es interesante para completar regiones de la lista nueva, con el fin de aprender la tabla sin procesarla vorazmente.

Copia Estándar y Rápida

Según el desarrollo del índice que implementa el motor de la copia, distinguimos dos tipos de copia de casos entre listas: copia estándar y rápida.

La copia estándar considera el recorrido de todas las entradas de la tabla y la *Base* para establecer el desarrollo del índice. Copiamos caso a caso toda la información de la lista inicial en la lista nueva desde el primer desplazamiento hasta el último.

Observamos que al realizar la copia de contenidos de una estructura en otra se copian solamente los contenidos que tienen información, no es necesario copiar las entradas marcadas como desconocidas *unKB*. En general, se puede ignorar una de las modalidades en la copia, que se puede considerar el complemento del resto, y la marca *unKB* es tomada como conocimiento en esta operación. Lo más eficiente es fijarnos en la *moda* de la respuesta que presenta la lista e inicializar la lista nueva con dicha modalidad de la respuesta. La moda de la lista es muy fácil de obtener y el procedimiento tiene la complejidad de la lista, es decir, el número de ítems.

Si el esquema tiene más de 20 atributos y la tabla no está muy vacía la copia estándar no es útil, tiene la complejidad exponencial de la tabla. El procedimiento de copia de información entre listas *KBM2L* con diferente configuración o *Base* requiere que sean copiados del orden de $W_0 * \delta_0$ elementos o casos. Se trata de una estrategia muy poco eficiente aunque es la única forma exacta de realizar un $XBase$ para ciertas permutaciones. Éstas son áquellas en las que los atributos que permutan tienen poco peso en la *Base* inicial. En un $XBase$ es la permutación de los atributos de menor peso la causa de que pierdan la adyacencia inicial muchos casos, y por tanto, si había ítems grandes, en la nueva *Base* se fragmentan y viceversa. La inicialización de la lista nueva con la moda de la respuesta no facilita la copia de listas, salvo que dicha moda sea prácticamente toda la tabla.

La copia estándar requiere unos recursos computacionales muy grandes y es preciso evitarla. El problema es que la copia estándar no tiene en cuenta el almacenamiento de la lista en ítems y trata las listas como tablas, caso a caso.

Para definir la copia rápida, observamos que si las *Bases* tienen atributos en común en las posiciones de menor peso (aquellas más próximas a la derecha), los ítems pueden copiarse siguiendo el almacenamiento de los casos en las listas mediante ítems. Estos $XBase$ permiten poner en práctica un procedimiento de copia rápido, que copia bloques en lugar de casos individuales. En dicho procedimiento la inicialización de la lista destino es una clonación de la lista origen, en lugar de la lista vacía, y es necesario copiar todos los casos, incluidos los de ítems de respuesta *unKB*. La copia rápida es factible en ciertos $XBase$ donde la mayoría de los índices son invariantes en el $XBase$ y además están agrupados en grandes bloques invariantes, pues esto permite copiar bloques de información (véase el ejemplo 3.5, donde podemos considerar varias dimensiones adicionales que no permutan).

En la sección 3.4 se propone una clasificación de los $XBase$ con el fin de caracterizar la

relación entre la transposicion de atributos y las listas que se obtienen. Los $XBase$ de tipo 1 y 3 son los que sacarán partido a la copia rápida.

El algoritmo de invarianza de índices es clave en el procedimiento de la copia rápida. Si el índice I es la parte fija de un ítem, el algoritmo prueba que dicho ítem es fijo/móvil en la transformación. Si todo el ítem no es fijo puede que lo sea un bloque grande de sus casos y procedemos a la copia de los casos fijos en bloque.

Ejemplo 3.5. Sea una lista con el esquema de atributos de dominio binario $\{A_0, A_1, A_2, \ldots, A_{\delta-1}\}$ y sea la $Base$ inicial $[A_0, A_1, A_2, \ldots, A_{\delta-1}]$ y la $Base$ final $[A_1, A_0, A_2, \ldots, A_{\delta-1}]$. Sean los bloques de casos (que pueden ser o no ítems): U, V, X e Y. El $XBase$ rápido consiste en la transposición de los bloques tal y como mostramos a continuación:

$[A_0, A_1, \ldots]$	Respuesta	$[A_1, A_0, \ldots]$	Respuesta
0 0 ...	U	0 0 ...	U
0 1 ...	V	0 1 ...	X
1 0 ...	X	1 0 ...	V
1 1 ...	Y	1 1 ...	Y

donde los bloques U e Y son fijos y los bloques X y V intercambian sus posiciones. □

Si interpretamos la optimización de la lista como un proceso de aprendizaje, estos dos últimos tipos de copia con los que implementamos el $XBase$ manifiestan cómo en configuraciones malas (listas fragmentadas en muchos ítems), la copia rápida no es útil pues no podemos mover bloques grandes de casos. Por otra parte, la copia estándar no está restringida a ningún $XBase$ y permite aprender aunque lentamente. Sin embargo, una vez que disponemos de una configuración o $Base$ buena, la copia rápida permite copiar enormes bloques de casos incrementando la velocidad notablemente. En resumen, se trata de un procedimiento sobre la estructura de la lista *KBM2L* con cierta analogía con la conocida curva de aprendizaje (de las personas), cuya pendiente es casi horizontal en los comienzos y se torna más vertical conforme se aprende.

Ejemplo 3.6. El Cuadro 3.3 muestra en detalle un $XBase$ igual al mostrado en el ejemplo 3.4, $[A, B, C, D] \rightarrow [B, A, C, D]$, donde los atributos son binarios. Observamos que éste es un $XBase$ de tipo 1, véase la sección 3.4. El $XBase$ produce una rejilla que compartimenta la tabla, con una resolución de $|A| \times |B| = 4$ (véanse las líneas horizontales).

Una vez que hemos clonado la lista origen, comenzamos la copia. Recordamos que es necesario seguir los pasos:

1. leer un caso;

2. permutar los valores de los atributos en el índice para calcular el nuevo desplazamiento;

Cuadro 3.3: Procedimiento de copia rápida en listas *KBM2L*

[ABCD] Inicial	desplazamiento [ABCD]	respuesta		[BACD] Nueva	desplazamiento [ABCD]	respuesta
0000	0	X	$*$	0000	0	X
0001	1	X		0001	1	X
0010	2	X		0010	2	X
0011	3	X		0011	3	X
0100	4	X		0100	8	X
0101	5	Y		0101	9	X
0110	6	Y		0110	10	X
0111	7	Z		0111	11	X
1000	8	X		1000	4	X
1001	9	X		1001	5	Y
1010	10	X		1010	6	Y
1011	11	X		1011	7	Z
1100	12	Z	$*$	1100	12	Z
1101	13	Z		1101	13	Z
1110	14	Z		1110	14	Z
1111	15	Z		1111	15	Z

3. comparar su respuesta respecto de la asignada en la clonación;

4. actualizar, si son diferentes; y

5. aplicar las reglas de gestión de ítems para organizar la tabla como una *KBM2L*.

La idea aquí es copiar sólo el conocimiento que se mueve con respecto a la tabla origen, copiar en bloque los casos adyacentes con índice invariante y desplazamiento no invariante y no tocar los casos con desplazamiento invariante. Para los índices de los atributos involucrados en la permutación (A y B) que tienen la misma secuencia de valores antes y después de la permutación, el valor del desplazamiento no cambia, véase el algoritmo *fix* en la sección 3.2.1. En nuestro ejemplo son los índices con $AB = 00$ y $AB = 11$. El símbolo $*$ en la tabla significa que estamos al comienzo de un segmento de desplazamientos invariantes de la rejilla. Así, hemos evitado copiar (y otras tareas enumeradas anteriormente) 8 casos.

Más aún, el bloque de casos con desplazamiento 4 a 7 en la tabla origen se mueve a la posición del bloque de desplazamientos 8 a 11 en la tabla destino, y viceversa. Entonces, cuando detectamos un bloque, se lanza la copia en bloque. Esto significa que las cuatro respuestas de

un bloque son copiadas directamente, en el orden original, evitando la tarea de $XBase$ (paso 2). La copia de un bloque consiste en copiar el caso de desplazamiento máximo a la lista nueva y eliminar los ítems comprendidos entre los desplazamientos mínimo y máximo, que son parte del bloque. Al final del proceso hay que revisar la consistencia de la lista *KBM2L* para que no haya ítems adyacentes de igual respuesta.

Las filas 6, 7, 8, 10, 11, 12 en la segunda tabla son los únicos casos que han tenido que actualizarse. Así, la copia estándar copia 16 casos, mientras que la copia rápida sólo necesita copiar 6 casos y en bloque. El ahorro en un esquema grande es mucho mayor. □

En el capítulo 2 y en esta sección hemos comentado que cuando el esquema tiene un cardinal homogéneo aparecen algunas propiedades, como la equivalencia entre la invarianza del desplazamiento y del índice al realizar un $XBase$. Esto tiene relación con las técnicas de codificación propias de las herramientas de minería de datos ((Mannila, 2000; Holsheimer y Siebes, 1994) y (Witten y Frank, 2005)), donde es habitual codificar los atributos de forma binaria antes de construir los modelos (redes de neuronas artificiales, regresión lineal, árboles de clasificación,...). La razón que hay tras esta práctica es que los modelos son más flexibles, independientes del dominio y los algoritmos son más eficientes. También son capaces de generalizar mejor y representar el conocimiento con mayor transparencia, a costa de alterar el modelo de los datos, lo cual puede dificultar la interpretación de los resultados. Vemos que hay áreas de la inteligencia artificial que utilizan en sus respectivos problemas la codificación, que llamamos cambio de base no natural, que consiste en revisar el modelo de los datos, la conceptualización del modelo de las variables del problema.

En el ejemplo 3.6 la invarianza de índice y de desplazamiento son, a diferencia del ejemplo 3.4, equivalentes y más fácil de procesar. Por ello puede interesar alterar el esquema y procesar los $XBase$ en dominios binarios y deshacer el cambio de esquema al final. Esto es una ventaja, pues la clonación de las listas permite acelerar el $XBase$ mucho más que en el ejemplo 3.4. Permite por ejemplo analizar muchas configuraciones y reordenaciones de los casos en adyacencias (y de ítems que representan asociaciones, reglas,...) que no son posibles en el esquema inicial. Claro que hay dos pagos asociados: por una parte, semánticamente desconectamos del modelo inicial del problema y dificultamos la interpretación. Por otra parte, necesitamos más memoria para representar el nuevo esquema, que tiene más atributos y codifica restricciones sintácticas. El estudio de las transformaciones del esquema, no solamente a dominio binario, como descomposiciones y composiciones, añadir atributos artificiales (análogas a las variables artificiales del algoritmo del símplex de la programación lineal (Taha, 2002)),..., es muy amplio y no tratado aquí, salvo como recurso de mejora de la representación del conocimiento de las tablas.

Copia Total y Parcial

Según el alcance de la transferencia de información entre las listas, distinguimos dos tipos de copia de casos entre listas: copia total y parcial.

La copia parcial es una forma aproximada de realizar el $XBase$, eludiendo la copia total. La idea es llevar algunos casos de la lista original a la nueva. Después se completa la lista nueva, generalizando los huecos de respuesta desconocida, por un procedimiento de copia a demanda. A continuación se examina la lista nueva comparándola con la lista inicial. Se trata de *comprobar* las condiciones de contorno (debe de conservarse la información de todas las modalidades de respuesta) y *examinar* algunos casos para ver si son iguales en ambas $Bases$. La cantidad de información transpuesta en la nueva lista y el resultado del examen da una medida de la probabilidad de que la copia generalizada sea igual a la copia total. Nos parece que el enfoque más adecuado es utilizar la teoría de muestreo para extraer contenidos de la lista original y llevarlos sobre la lista propuesta, inicialmente vacía. Para seleccionar los casos que participan en la transposición, que entrenan la lista nueva, se propone hacer un muestreo probabilista (Lohr, 2000), conforme a la distribución de las modalidades de respuesta y añadir otros tantos casos seleccionados uniformemente en el rango del desplazamiento. Debido a que generalmente la copia total es imposible de realizar, se trata de hacer una copia parcial que permita comparar la calidad de las $Bases$, respecto de la optimización del almacenamiento mediante el examen de muestras de casos generalizados en dicha copia parcial.

Ejemplo 3.7. Sea el ejemplo 3.4 donde ilustrábamos la invarianza de índices que es relevante en la copia rápida. Ahora mostramos la copia parcial en aquella tabla y un posible algoritmo.

La lista inicial es: $\langle 5, X|\ \langle 7, Y|\ \langle 11, X|\ \langle 17, Y|\ \langle 21, X|\ \langle 23, Y|$.

En primer lugar llevamos a la lista vacía en la $Base$ $[B, A, C]$, $\langle 23, -1|$, los casos de los ítems de la lista inicial con desplazamiento máximo. Son 6 casos: $(5, X)$, $(7, Y)$, $(11, X)$, $(17, Y)$, $(21, X)$ y $(23, Y)$. En una lista muy larga solamente copiaremos proporcionalmente unos pocos, por ejemplo, los que el sistema pueda copiar en 2 segundos, elegidos al azar. La lista nueva es: $\langle 8, -1|\ \langle 9, X|\ \langle 10, -1|\ \langle 11, Y|\ \langle 12, -1|\ \langle 13, Y|\ \langle 18, -1|\ \langle 19, X|\ \langle 20, -1|\ \langle 21, X|\ \langle 22, -1|\ \langle 23, Y|$.

En segundo lugar realizamos copia a demanda de casos desconocidos en la lista nueva. Por ejemplo, el caso central de cada ítem desconocido o de algunos. Se trata de los casos de la lista nueva: $(4, Y)$, $(10, Y)$, $(12, Y)$, $(16, X)$, $(20, X)$ y $(22, Y)$; cuyas respuestas recuperamos de los casos correspondientes: $(12, Y)$, $(6, Y)$, $(16, Y)$, $(8, X)$, $(20, X)$ y $(22, Y)$; de la lista inicial. Así, la copia a demanda nos permite aprender la lista: $\langle 3, -1|\ \langle 4, Y|\ \langle 8, -1|\ \langle 9, X|\ \langle 13, Y|\ \langle 15, -1|\ \langle 16, X|\ \langle 18, -1|\ \langle 21, X|\ \langle 23, Y|$.

En tercer lugar generalizamos la respuesta desconocida, por ejemplo mediante el vecino más

próximo (Mitchell, 1997) (en la métrica del desplazamiento (Fernández del Pozo et al., 2005)). Esto es una heurística que tiene el fundamento siguiente: si la nueva *Base* fuese mejor, los ítems serán menos y más grandes. Por tanto, en las proximidades de un caso, es más probable que haya casos de igual respuesta. Así, generalizando los ítems desconocidos flanqueados por ítems de igual respuesta obtenemos:

$\langle 4, Y|\ \langle 8, -1|\ \langle 9, X|\ \langle 13, Y|\ \langle 15, -1|\ \langle 21, X|\ \langle 23, Y|$.

Al resto de ítems desconocidos les asignamos la respuesta del ítem más próximo, y si es equidistante lo copiamos a demanda:

$\langle 6, Y|\ \langle 9, X|\ \langle 14, Y|\ \langle 21, X|\ \langle 23, Y|$.

Ahora falta comprobar las condiciones de contorno de listas. Debe haber menos de 14 respuestas de X y menos de 10 respuestas de Y. Calculamos el tamaño de los ítems y realizamos el balance:

$\langle 6, Y|_7\ \langle 9, X|_3\ \langle 14, Y|_5\ \langle 21, X|_7\ \langle 23, Y|_2$.

Hay $7 + 5 + 2 = 14$ respuestas de Y y $3 + 7 = 10$ de X. La lista generalizada no coincide con la inicial que tiene 14 respuestas X y 10 respuestas Y. Debemos ir al paso segundo para realizar otra copia a demanda de los casos centrales de los ítems (si es número par se copia el central derecho). Copiamos los casos $(2, X)$, $(7, Y)$, $(15, X)$ y $(18, X)$; cuyas respuestas recuperamos de los casos correspondientes: $(2, X)$, $(15, Y)$, $(19, X)$ y $(10, X)$ de la lista inicial. Obtenemos:

$\langle 1, -1|\ \langle 2, X|\ \langle 3, -1|\ \langle 4, Y|\ \langle 6, -1|\ \langle 7, Y|\ \langle 8, -1|\ \langle 9, X|\ \langle 13, Y|\ \langle 14, -1|\ \langle 16, X|\ \langle 17, -1|\ \langle 21, X|\ \langle 23, Y|$.

Generalizamos los ítems desconocidos y obtenemos:

$\langle 3, X|_4\ \langle 7, Y|_4\ \langle 9, X|_2\ \langle 13, Y|_4\ \langle 21, X|_8\ \langle 23, Y|_2$,

donde los casos $(3, -1)$, $(8, -1)$ y $(14, -1)$ son equidistantes y les asignamos las respuestas X, X y X, respectivamente, tal y como tienen en la lista inicial. A los casos $(0, -1)$, $(1, -1)$, $(6, -1)$ y $(17, -1)$ se les asignan las respuestas generalizadas X, X, Y y X, respectivamente, pues están flanqueados por casos de igual respuesta o en los extremos de la lista.

Comprobamos las condiciones de contorno de listas y vemos que se cumplen: hay $4+2+8 = 14$ respuestas de X y $4 + 4 + 2 = 10$ respuestas de Y. Ahora corresponde examinar la lista nueva mediante la lista inicial:

Lista inicial	Lista nueva
$\langle 5, X|\ \langle 7, Y|\ \langle 11, X|\ \langle 17, Y|\ \langle 21, X|\ \langle 23, Y|$	$\langle 3, X|\ \langle 7, Y|\ \langle 9, X|\ \langle 13, Y|\ \langle 21, X|\ \langle 23, Y|$

En el ejemplo, el último paso de generalización ha consistido en copias a demanda salvo para cuatro casos. Es preciso realizar un examen, pues hemos generalizado algunas respuestas. En general, puede ocurrir que se verifiquen las condiciones de contorno pero las listas no sean equivalentes. Para asegurarnos de que la copia parcial es correcta con cierta probabilidad, rea-

lizamos pruebas aleatorias de igualdad de respuesta entre casos homólogos de ambas listas, la inicial y la nueva. Terminamos la copia cuando la lista nueva supera las pruebas con un número suficiente de casos. □

Por último, consideramos el aspecto computacional en nuestro problema de complejidad exponencial y las características de una copia concreta. Si disponemos de recursos computacionales puede realizarse una copia paralela, pues la estructura de la lista permite fácilmente realizar copias en paralelo. Las transposiciones no se interfieren. La copia paralela intenta acelerar el procedimiento de copia explotando la estructura matricial subyacente a la lista y la estructura vectorial del índice.

Tanto la copia rápida (estructura de la lista) como la copia parcial (generalización iterativa) están relacionadas con el aprendizaje. En la práctica consideramos diferentes combinaciones de modos de copia: la copia total estándar de oferta y paralela, la copia total rápida de oferta y secuencial, la copia parcial rápida a demanda, ..., según el $XBase$, la lista objeto y los recursos computacionales.

En el resto del capítulo exponemos el procedimiento de búsqueda de la *Base* óptima que implica la copia de listas, así como los problemas relacionados, analizando las soluciones parciales que aportan las heurísticas expuestas anteriormente.

3.3. Caracterización de la Búsqueda

En esta sección describimos algunas ideas útiles para aproximarnos a las heurísticas que permitirán acelerar el proceso de búsqueda de la *Base* óptima.

El tamaño del espacio de búsqueda de *Bases* es tan grande que se precisan heurísticas para poder alcanzar una solución satisfactoria en un tiempo razonable.

Así, debemos generar información relativa al espacio de búsqueda, mientras se realiza dicha tarea (Dorigo et al., 1999) para que el proceso de generación aprenda cómo generar *Bases* mejores, con probabilidad alta de unión de contextos o ítems. En nuestro problema esta información se centra en tres aspectos:

1. la estimación CMRS (véase el test ítem-CMRS en esta misma sección),
2. la relevancia de los atributos (véase la sección 2.4) y
3. la similitud entre *Bases* (véase la medida de proximidad entre *Bases* en la sección 3.4).

Los tres aspectos se concretan, respectivamente, en disponer de:

1. una acotación inferior del objetivo a optimizar,

2. una caracterización de las soluciones y

3. una métrica entre soluciones.

En nuestro problema particular realizamos el siguiente análisis que será útil para diseñar los algoritmos de optimización que mostramos en este capítulo.

En principio, la probabilidad de obtener en un ensayo una *Base* óptima, siendo única, es muy baja, $\frac{1}{\delta!} = P_\delta$. La probabilidad de obtener una *Base* óptima en un test aleatorio es $\frac{\beta}{\delta!}$ si hay β soluciones óptimas equivalentes y si todos los atributos son relevantes, respecto de la determinación conjuntiva de la respuesta. En el capítulo anterior presentamos el fundamento del análisis de los atributos en una *Base*. El índice y su parte fija en un ítem permiten hacer una declaración ordinal de la importancia relativa de los atributos y sus valores en la tabla. Y además facilita un mecanismo automático de elaboración de explicaciones. En el capítulo 4 veremos algunos ejemplos aplicados a TDO. De acuerdo con este planteamiento general, si conocemos a priori la relevancia de algunos, v, atributos del esquema, la anterior probabilidad P_δ es $\frac{v!}{\delta!} = P_{\delta-v}$. Observamos que $P_{\delta-v} >> P_\delta$ si $v > 0$, es decir, si tenemos algún conocimiento del dominio del problema.

La condición de parada de la búsqueda la determina el número de ítems de la lista. Este número siempre es mayor o igual al cardinal del dominio de la respuesta. Cuando es igual sabemos que hemos alcanzado el óptimo. Si no es igual, no hay una condición general que evite explorar todo el espacio.

3.3.1. Caracterización del Espacio de la Búsqueda

La información que podemos recabar del proceso de búsqueda está relacionada con la estructura local y global de las soluciones y con las características de las soluciones provisionales que obtenemos mediante diversos "test".

Estructura local de las soluciones. La idea es plantear una relación de proximidad entre las soluciones. De este modo, podemos buscar *lejos* de una *Base* cuya lista es muy larga y viceversa. La clave es definir una medida de proximidad que tenga en cuenta la relación entre las *Bases* y el tamaño de las listas asociadas.

Como veremos posteriormente, para generar información útil en el proceso de búsqueda se propone moverse por el espacio de permutaciones a elementos de distancia *Hamming* $< H$ y utilizar estadísticas que describan los ítems de la lista *KBM2L*. Con esta información imponemos

restricciones a las permutaciones que generamos en el algoritmo de búsqueda de forma que evitamos fragmentar los ítems.

Las restricciones consisten, por una parte, en no considerar permutaciones arbitrarias sino en dar más oportunidad o probabilidad a la permutación de índices restringida a la parte fija correspondiente de los ítems pequeños hacia la zona de la parte variable. La razón es que si deseamos que dos ítems de una familia, de igual contenido, se conviertan en uno, deberíamos considerar permutaciones restringidas a la parte fija hacia la variable pues la parte fija se reduce al efectuarse la unión. La unión de varios ítems de la lista se produce en la *Base* adecuada. Esta *Base* es mejor que otra donde se produce fragmentación de ítems.

Estas permutaciones hacen que algunos atributos fijos de ítems pequeños pierdan peso. Es decir, nos centramos en las familias que presentan mayor fragmentación. Los ítems pequeños tienen una parte fija grande y en el conjunto de *Bases* posibles seleccionamos permutaciones con cambio de peso grande. Un peso inadecuado para un atributo provoca fragmentación de las familias de ítems. El peso óptimo de un atributo puede ser distinto según la familia, y el que buscamos en la lista es el que proporciona el tamaño global mínimo.

Por otra parte, conforme avanza el proceso de búsqueda y las soluciones son mejores, es conveniente restringir algo la magnitud de los cambios de peso en los atributos al cambiar de *Base*. La medida de proximidad está inspirada en la caracterización de la discrepancia entre *Bases* respecto del peso de los atributos, como indicaremos posteriomente.

Estructura global de las soluciones. Deseamos generar información del espacio de búsqueda. En las secciones siguientes proponemos dos técnicas para minimizar el número de ítems de las listas. La primera usa una metodología inspirada en los algoritmos de entorno variable (Hansen y Mladenović, 2001) y la segunda es un algoritmo genético (Holland, 1975; Michalewizc, 1992; Mitchell, 1998), ambos adaptados a nuestro problema específico.

Así, en la sección 3.4 implementamos un algoritmo de entorno variable como estrategia local. En la sección 3.5 establecemos un juego de funciones de *evaluación* de soluciones en el algoritmo genético ad hoc, propuesto para resolver la búsqueda mediante una estrategia global. Introducimos información del dominio en el algoritmo, lo cual multiplica su eficiencia notablemente al limitar la población de individuos a un subconjunto que contiene el óptimo. En la sección 3.6 realizamos experimentos con todos estos algoritmos.

En cada iteración los algoritmos mantienen solamente *una* lista *KBM2L* y un conjunto de *Bases*. El proceso de optimización trabaja en la *Base* candidata usando el *valor* de la función de evaluación de cada *Base* o la proximidad entre *Bases*. Lógicamente, no almacenamos una población de listas, o de entornos de listas, con todos sus casos en todas las *Bases*.

Nótese que la compactación de los ítems no es independiente: si varios ítems de respuestas iguales dan lugar a uno, es posible que otros se unan también. Si la respuesta es binaria, lo anterior es seguro, véase el ejemplo 2.10 de la sección 2.4 donde la optimización de las familias es dual.

Las soluciones ensayadas se memorizan para no repetir cálculos, se evita separar mucho los atributos que manifiestan cierta afinidad (heurística) y la copia exhaustiva del contenido de las matrices en la comparación de listas en *Bases* diferentes.

3.3.2. Caracterización del Proceso de Búsqueda

Terminamos esta sección con una exposición detallada del filtro de pruebas a las que se somete una *Base* en el proceso de búsqueda. La idea es evitar una copia si se puede descartar a priori, por no ser óptima.

El **test de** *Base* **(traza)** supone almacenar un conjunto razonablemente grande de *Base* cuyo valor (tamaño de la lista) es conocido. No almacenamos toda la traza pues puede ser muy grande. El caso más sencillo es cuando se almacena la última que se ha obtenido, pero así es bastante probable que se repitan pasos en la búsqueda. La traza completa garantiza una búsqueda monótona creciente estricta del óptimo, pero es muy costosa en términos de memoria y complica la búsqueda considerablemente.

El **test de atributos (importancia)** implementa de forma general las condiciones de relevancia/irrelevancia de atributos en una familia de ítems. Mediante los ejemplos mostrados en la sección 2.4 ilustramos cómo detectar una permutación prometedora para la unión de ítems de la familia y viceversa. La importancia de atributos se calcula al principio de la búsqueda. De este modo, caracterizamos cada atributo con un indicador de importancia en las diferentes modalidades de la respuesta. Recordamos que optimizar una familia no tiene por qué optimizar la lista, que es la representación conjunta de todas las familias.

En problemas donde el tamaño del conjunto de atributos es muy grande, puede ser necesario fijar el peso o posición de algunos atributos a priori en la *Base* y proceder mediante aprendizaje del subproblema. Esto es equivalente a fijar el papel de un atributo como relevante (peso alto) o irrelevante (peso bajo). Ésta es una estrategia razonable para comenzar. Afortunadamente, el proceso de generación de bases aprende cómo generar listas mejores, con alta probabilidad de unión de contextos. Con esta información podemos imponer restricciones al orden de los atributos y prevenir la fragmentación de los ítems mientras promovemos las uniones.

La regla general para guiar la búsqueda de *Bases* óptimas, desarrolla la idea propuesta de transposición de los índices de atributos que cubren todo su dominio en la familia de ítems hacia

posiciones de menor peso, véase la sección 2.4. Se trata de un test de relevancia de atributos, local a la familia. El atributo asociado es irrelevante y no fragmentará la familia. En principio, el $XBase$ que disminuye el peso de estos atributos podrá unir los ítems. La mejora depende del atributo con el que permute su posición y de la repercusión sobre las otras familias. La información de importancia se materializa en un conjunto de directrices que permiten conocer a priori un orden parcial entre las $Bases$, pues la importancia es siempre relativa a una familia y sin embargo, la lista óptima contempla todos los ítems. Es decir, descartamos las $Bases$ que no están de acuerdo con el orden parcial que suministra el análisis de importancia.

Por último, con el **test de ítem-CMRS (acotación de la coalescencia) global y local** podemos conseguir un gran ahorro computacional evitando tentativas fallidas de copias. La idea consiste en tomar una muestra de los casos a copiar y decidir si merece la pena completar el $XBase$ de toda la lista o si rechazamos la $Base$. No debemos hacer la copia completa de forma ciega, debido a su alto coste, y es mejor hacer cierta prospección del resultado. Copiamos unos pocos casos de la lista actual en la $Base$ inicial a la lista en la $Base$ nueva. En concreto, nuestro programa copia los casos I_{sup}, I_{inf} y un caso intermedio (elegido aleatoriamente) de cada ítem, siempre con respuesta conocida, y cuenta los Cambios de Modalidad de la Respuesta en la Secuencia (CMRS) de ítems. Así, exploramos la lista y registramos un CMRS si vemos

$$\ldots\langle p, A|\langle r, -1|\langle q, B|\ldots \quad \text{o}$$
$$\ldots\langle p, A|\langle q, B|\ldots,$$

donde la respuesta A es diferente de la respuesta B. Observamos que la realización de este test de CMRS tiene la complejidad de la lista que es mucho menor que la complejidad de la tabla, $W_0 * \delta_0$. Podemos rechazar las $Bases$ que tienen más CMRS que la $KBM2L$ mínima que almacenamos como solución provisional. La razón es que el número de CMRS es una cota inferior de la longitud de la lista nueva. Por supuesto, interrumpimos la prueba si el recuento supera la cota CMRS actual del óptimo.

El criterio general de mejora es que ambas estructuras almacenan la misma información pero es mejor aquella que lo hace en una lista más corta. Si la tabla original es muy dispersa, el criterio alternativo a la longitud de la lista es el propio registro CMRS, tal como vimos en el ejemplo 3.12. Por ello, planteamos evitar la copia si se detecta que la permutación propuesta no es mejor, es decir, fragmenta los ítems o contextos, y la lista $KBM2L$ es de mayor tamaño o mayor CMRS.

Recordamos que la forma de inicialización de la lista en la $Base$ nueva está relacionada con el tipo de copia: se inicializa con $unKB$ o la moda de la respuesta en la tabla si la copia es estándar, o con la clonación de la lista actual si la copia es rápida. El primer caso utiliza un valor por

defecto, mientras que el segundo explota la abundancia de índices de casos fijos consecutivos en determinados $XBase$. Si realizamos copias rápidas, el test CMRS sobre la lista nueva supone cierto coste computacional adicional, debido a que no utilizamos la lista resultante del test para el resto del procedimiento de copia, que en este caso inicializa la lista nueva con la clonación o réplica exacta de la lista original.

Este test permite realizar el filtrado de $Bases$ para la heurística de entorno variable, que tiene complejidad cuadrática en el tamaño de la $Base$, véase la sección 3.4. Además, es el núcleo de la función de evaluación del algoritmo genético, véase la sección 3.5. Los resultados experimentales muestran que es muy eficiente: bajo coste computacional, alto ratio de rechazo de $Base$ no óptimas y nunca rechaza la $Base$ óptima.

3.4. Heurística de Entorno Variable

El cambio sistemático de vecindad dentro de una búsqueda posiblemente aleatoria es una metaheurística simple y efectiva para optimización global y combinatoria. Esta técnica se denomina búsqueda de entorno variable (Hansen y Mladenović, 2001).

Para nuestro problema de optimización proponemos movimientos a través del espacio de permutaciones hacia elementos a distancia $Hamming > H$, que definen H-entornos, y calculamos indicadores estadísticos (mínimo, máximo, media del tamaño de los ítems de las familias, ...) que describen los ítems de la lista *KBM2L*. La distancia de Hamming es lógicamente 0 cuando las dos $Bases$ son la misma, 2 si permuta una pareja de atributos exclusivamente y como máximo es δ si todos los atributos ocupan posiciones distintas en ambas bases.

En primer lugar, intentamos una combinación de búsqueda local y global. La estrategia es mezclar búsqueda local mediante enumeración de las $Bases$ en 2-entornos (todas las $Bases$) con algunos saltos aleatorios en los δ-entornos (lejos de la solución actual). Entendemos que *local* significa cambiar pocos atributos de una $Base$ a otra. El uso de la búsqueda local nos dio resultados suficientemente buenos por sí sola con frecuencia. Así, si la $Base$ es $[0, 1, 2, 3]$ y cambiamos solamente dos atributos, nos movemos a través de su 2-entorno: $\{[1, 0, 2, 3], [0, 2, 1, 3], [0, 1, 3, 2], [2, 1, 0, 3], [0, 3, 2, 1], [3, 1, 2, 0]\}$. Esto supone moverse hacia todos los elementos de distancia Hamming H igual a 2. En general, un 2-entorno contiene $\delta\,(\delta - 1)/2$ vecinos, en tanto que hay $(\delta - 1)!$ vecinos a distancia H igual a δ. Para la $Base$ anterior son: $\{[1, 2, 3, 0], [2, 3, 0, 1], [3, 0, 1, 2], [3, 2, 1, 0], [2, 0, 3, 1], [3, 2, 1, 0]\}$.

Posteriormente descubrimos que nuestra idea de *localidad* o *vecindad* debía ser revisada. De modo directo podemos establecer una distancia entre las listas en diferentes $Bases$ mediante la longitud de la lista en ítems. Había que desarrollar una propuesta de distancia que tuviese en

cuenta no solamente la diferencia entre los códigos de la permutación de atributos (distancia de Hamming), sino la semejanza entre las listas y las diferencias entre los patrones de respuestas de los ítems. Una distancia entre *Bases* sería muy útil para organizar la búsqueda si recogiese la semejanza de las listas a las listas que alcanzan el objetivo de mínima longitud. La distancia de Hamming no aporta información sobre la proximidad o semejanza entre las correspondientes listas. Una observación clave fue que el efecto en la lista del cambio de posición de dos atributos es completamente diferente dependiendo de dónde están situados en la *Base*, aunque se trate de distancias Hamming iguales a 2 en todos los casos que involucran a dos atributos.

Así, sean B y B' la *Base* incial y la nueva, respectivamente. Sea $B \rightarrow B'$ la expresión del *XBase*. Sugerimos una clasificación de los *XBase* naturales simples:

1. *XBase* que permutan pocos atributos, todos ellos con peso alto en ambas *Bases*. El efecto en la lista inicial es su transformación por medio de movimientos globales de grandes bloques de información, de casos.

2. *XBase* entre pocos atributos, todos ellos con peso bajo en ambas *Bases*. El efecto es una transposición de la información, de los casos, por medio de movimientos locales de bloques de tamaño muy reducido, unos pocos casos.

3. *XBase* entre pocos atributos, todos ellos con un peso intermedio en ambas *Bases*. La lista inicial es transformada por medio de movimientos globales, de bloques pequeños.

4. Mezcla de los tipos de *XBase* 1, 2 y 3.

5. *XBase* generales, donde muchos atributos cambian de posición (de peso), con cualquier valor de peso posible. La lista se transforma de modo aparentemente desordenado y caótico.

Obsérvese que el tamaño de los bloques de información es del orden del peso mayor involucrado en la permutación de atributos.

La Figura 3.3 resume los tipos de *XBase*. Obviamente, no hay una relación directa entre los *XBase* y la longitud de la lista *KBM2L*. Si no, podríamos elegir el *XBase* mejor y obtener la *KBM2L* óptima fácilmente, y esto no es posible. Depende del contenido de la tabla y una tabla concreta no da pistas respecto del *XBase* más eficaz. Por otra parte, como vimos en la sección 3.2.1, cualquier *XBase* tiene un conjunto de casos que no se mueven en la transformación, es decir, el desplazamiento es el mismo en ambas *Bases*, o hay grandes secuencias de casos que se mueven solidariamente conservando la adyacencia, es decir, sus índices son invariantes. Esto es importante, porque si no se mueven muchos casos, las listas tendrán longitud similar. Observamos que lo contrario no es cierto.

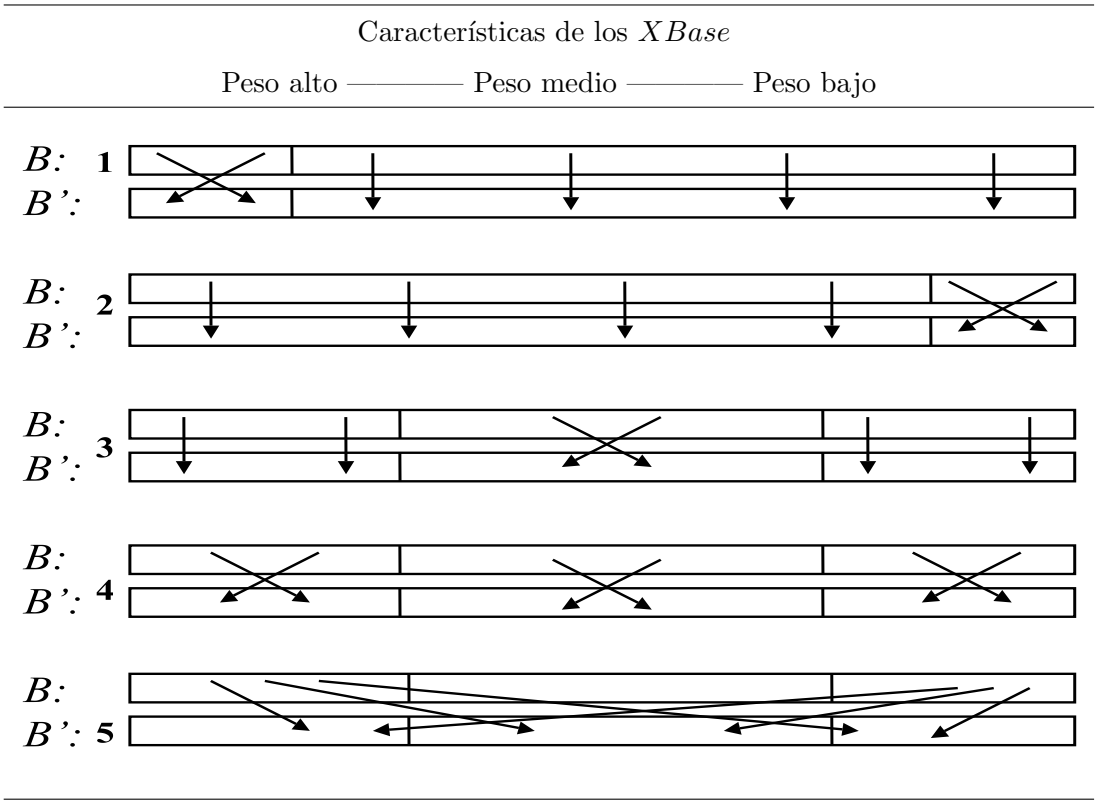

Figura 3.3: Clasificación de los $XBase$

A pesar de todo, esta importante descripción cualitativa de los tipos de $XBase$ nos permite proponer un conjunto de heurísticas para guiar la búsqueda. Se basarán en una idea alternativa de vecindad más precisa y cercana a nuestro problema de optimización. Dada una lista concreta pretendemos establecer en el espacio de búsqueda (permutaciones de atributos del esquema), cierta métrica de proximidad entre las $Bases$ que tiene en cuenta la importancia de los atributos, el peso.

Por consiguiente, vamos a introducir una nueva medida de proximidad entre las $Bases$ que tiene en cuenta el efecto del $XBase$ inducido en la nueva lista. Para un $XBase$ de tipo 1, la nueva lista *KBM2L* será similar a la antigua lista, y el valor de la medida será bajo. Para un $XBase$ de tipo 5, la nueva lista será bastante diferente, es decir, nos habremos movido lejos de la $Base$ inicial, según esta medida.

Sea G una función definida del modo siguiente. Para todas las bases, $G(B, B) = 0$ y para B, B' tal que $B \neq B'$

$$G(B, B') = r_{izq} + \sum_{i=0}^{\delta-1} r_i + H(B, B') - 1$$

donde $H(B, B')$ es la distancia de Hamming entre $Bases$ y B', r_i es el número de atributos

entre la posición inicial y final de cada atributo permutado i, y r_{izq} es el número de atributos fijos adyacentes a la izquierda del atributo permutado de mayor peso.

G está subordinada a la distancia de Hamming. G cuenta, vía H, cuántos atributos han permutado en el $XBase$. Pero H es solamente una distancia entre códigos, y el papel que juega la $Base$ en el problema no es reflejado en su valor. Sin embargo, G contempla los atributos que han sido permutados teniendo presente el cambio de sus pesos, reflejando entonces hasta dónde se han movido en la transformación. Si un atributo no salta o salta a una posición adyacente, entonces no contribuye a r_i ($r_i = 0$). De este modo identificamos los $XBase$ conforme a la clasificación mostrada anteriormente.

Además, para mejorar la clasificación, G recoge la idea de que los $XBase$ entre unos pocos atributos son diferentes dependiendo de dónde están los atributos en las $Bases$ inicial y final. Por ejemplo, un intercambio del primer y segundo atributo (de mayor peso, por tanto se trata de un $XBase$ de tipo 1) conduce a pocos cambios en la *KBM2L* en términos de longitud, pues conserva la mayoría de las adyacencias entre casos que delimitan los ítems. Sin embargo, un intercambio entre el último y el penúltimo atributo (de menor peso , por tanto se trata de un $XBase$ de tipo 2) conduce a muchos cambios de adyacencia y por tanto puede fragmentar o concentrar los ítems de la lista. El término r_{izq} en G recoge esta idea: el último cambio mencionado refleja mayor valor de G que el primero.

El Cuadro 3.4 ilustra algunos ejemplos de la función G. El primer y tercer caso tienen valores de G pequeños porque los pesos de los atributos que permutan (X, Y y Z) cambian poco. Serían $XBase$ de tipo 3. El segundo y cuarto ejemplos tienen valores de G mayores porque los pesos de los atributos que permutan son bastante diferentes. Estos últimos son $XBase$ de tipo 5.

Aunque sería deseable, no es fácil definir una *distancia* en el espacio total de búsqueda que capture el efecto de los movimientos de los atributos. De hecho, G no es una distancia puesto que no cumple la desigualdad triangular, $G(B', B'') \leq G(B', B''') + G(B''', B'')\ \forall B', B'', B'''$.

Contraejemplo. Si B' es $[0, 1, 2, 3, 4, 5]$, B'' es $[0, 1, 2, 3, 5, 4]$ y B''' es $[1, 0, 2, 3, 4, 5]$, entonces $G(B', B''') = 1, G(B'', B''') = 3$, pero $G(B', B'') = 4 + 2 - 1 = 5$. Es decir, no se cumple la desigualdad triangular. □

No obstante, vamos a usar G restringido a un subconjunto de $Bases$ que describimos seguidamente (aunque tampoco es una distancia). Sea $N_{H2}(B_0)$ el conjunto de $Bases$ B tal que $H(B, B_0) = 2$ para una $Base$ B_0 dada. Para toda $B \in N_{H2}(B_0)$, denotamos G como

$$G_{H2}(B_0, B) = r_{izq} + \sum_{i=0}^{\delta-1} r_i + 1$$

Cuadro 3.4: Ejemplos de H y G

$XBase$	H	G
$--^{(q}--XY-----$ $-----YX-----$	2	$q+1$
$--^{(q}--X---^{(r}--Y-----$ $-----Y---^{(r}--X-----$	2	$q+2r+1$
$--^{(q}--XYZ-----$ $-----ZXY-----$	3	$q+3$
$-^{(q}--X---^{(r}--Y---^{(s}--Z---$ $----Z---^{(r}--X---^{(s}--Y---$ $----\mid--------^{(t}-----\mid---$	3	$q+r+s+t+2$

y $G_{H2}(B_0, B_0) = 0$.

Examinemos las diferencias entre la distancia de Hamming y la función G (y G_{H2}) más en profundidad. Dada una *Base* B_0, su vecindad o entorno $N_{H2}(B_0)$ es fácil de calcular. Para $B_0 = [0, 1, 2, 3, 4]$, $N_{H2}(B_0)$ incluye las siguientes 10 *Bases*:

$$
\begin{aligned}
B_0 &= [0,1,2,3,4] \\
B_1 &= [1,0,2,3,4] \quad B_6 = [0,3,2,1,4] \\
B_2 &= [2,1,0,3,4] \quad B_7 = [0,4,2,3,1] \\
B_3 &= [3,1,2,0,4] \quad B_8 = [0,1,3,2,4] \\
B_4 &= [4,1,2,3,0] \quad B_9 = [0,1,4,3,2] \\
B_5 &= [0,2,1,3,4] \quad B_{10} = [0,1,2,4,3]
\end{aligned}
$$

En el Cuadro 3.5 calculamos los valores G para cada pareja de *Bases* de $N_{H2}(B_0)$. La primera fila corresponde a $G_{H2}(B_0, B)$, $\forall B \in N_{H2}(B_0)$.

Obsérvese los diferentes valores que G_{H2} toma en la primera fila, desde 0 hasta 7. Así, para 5 atributos, la distancia H está en el rango discreto de {0,2,3,4,5}, mientras que G_{H2} está en $\{0, 1, 2, 3, 4, 5, 6, 7\}$. La Figura 3.4 dibuja esta idea gráficamente.

Observamos que los valores de G entre *Bases* (las otras filas del Cuadro 3.5) son también

Cuadro 3.5: Valores de G para cada par en $N_{H2}(B_0)$ ($\delta = 5$)

G	B_0	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8	B_9	B_{10}
$[01234] = B_0$	0	1	2	6	7	2	7	6	3	5	4
$[10234] = B_1$	1	0	3	5	7	3	5	7	3	5	3
$[21034] = B_2$	2	3	0	5	7	3	7	9	5	7	5
$[31204] = B_3$	6	5	5	0	7	7	5	11	5	9	7
$[41230] = B_4$	7	7	7	7	0	9	11	8	9	7	7
$[02134] = B_5$	2	3	3	7	9	0	4	6	4	6	4
$[03214] = B_6$	7	5	7	5	11	4	0	6	4	8	6
$[04231] = B_7$	6	7	9	11	8	6	6	0	8	6	6
$[01324] = B_8$	3	3	5	5	9	4	4	8	0	5	5
$[01432] = B_9$	5	5	7	9	7	6	8	6	5	0	5
$[01243] = B_{10}$	4	3	5	7	7	4	6	6	5	5	0

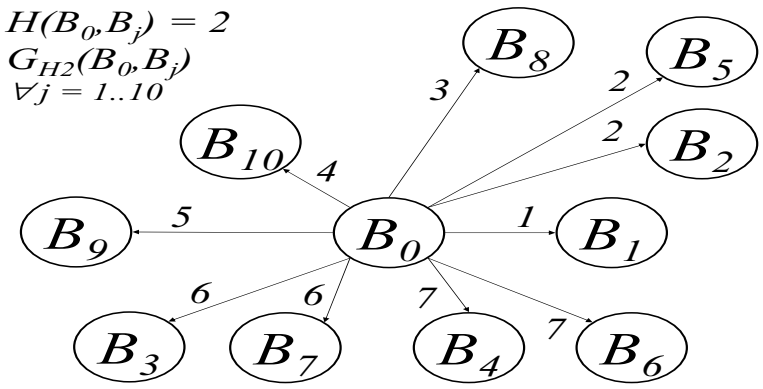

Figura 3.4: Valores de $G_{H2}(B_0, B_j), j = 1, ..., 10$

bastante diferentes, igual que sus distancias H, que no muestra el cuadro. Por ejemplo, el conjunto de *Bases* en $N_{H2(B_0)}$ tal que $G_{H2}(B_0, B_j) \leq 3$ es $\{B_0, B_1, B_2, B_5, B_8\}$. Es un entorno con radio 3 relativo a la medida G_{H2}. Entre ellas, las cantidades $G(B_i, B_j)$ pueden ser mayores que 3, véase la Figura 3.5. La distancia de Hamming entre ellas es siempre 3 para cada par (B_i, B_j) $i, j \neq 0$, excepto para (B_1, B_8), que es 4. Éste es un ejemplo de la G-proximidad entre *Bases* dentro del entorno $N_{H2}(B_0)$.

Por el contrario, el conjunto de *Bases* $N_{H2}(B_0)$ tal que $5 < G_{H2}(B_0, B_j) \leq 7$, es decir, con valores altos para G_{H2}, es $\{B_3, B_4, B_6, B_7\}$. Los valores $G(B_i, B_j)$ son también altos, véase la

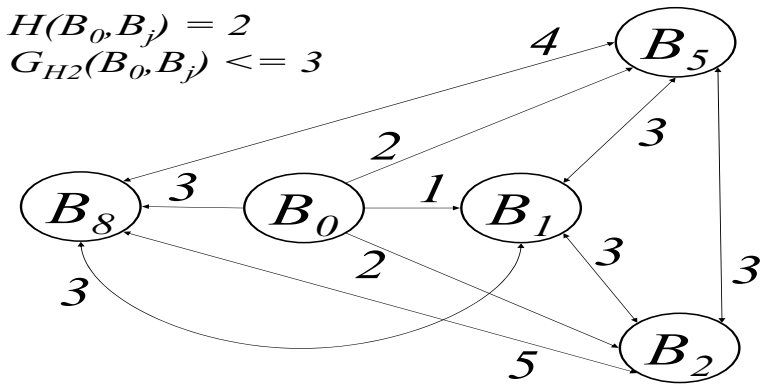

Figura 3.5: Valores de G para $Bases$ tal que $G_{H2}(B_0, B_j) \leq 3$

Figura 3.6, como un ejemplo de G-alejamiento dentro del entorno $N_{H2}(B_0)$. La distancia de Hamming entre ellas es siempre 3 para cada par, excepto para (B_4, B_6) y (B_3, B_7), que es 4. Observamos nuevamente que H discrimina muy poco comparado con G.

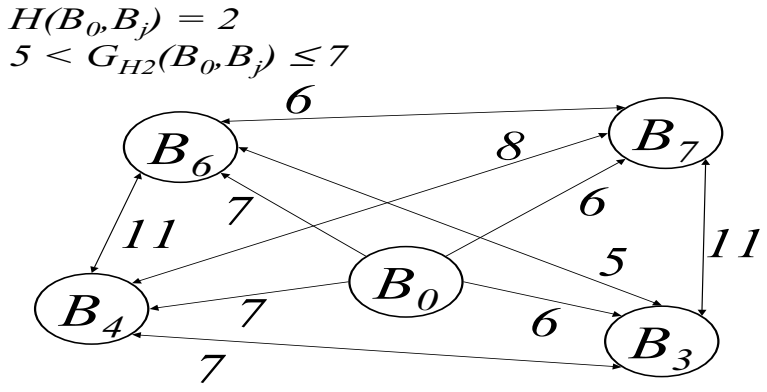

Figura 3.6: Valores de G para $Bases$ tal que $5 < G_{H2}(B_0, B_j) \leq 7$

Ahora se explican en cierta medida nuestros buenos resultados iniciales ya comentados mediante búsqueda local a través de entornos de distancia Hamming igual a 2: la búsqueda era de hecho a través de entornos variables respecto a G_{H2}.

La medida G_{H2} inducida en $N_{H2}(B_0)$ es, al igual que G, muy rica, pues distingue entre $XBase$. Esta nueva medida de proximidad será la guía para controlar la búsqueda. Asumiendo que estamos en la $Base$ B_0, intentamos movernos a una $Base$ B' en $N_{H2}(B_0)$. Si B_0 es juzgada como mala, el siguiente movimiento debería ser hacia una $Base$ alejada en términos de la escala G_{H2} (por ejemplo, $B_0 \rightarrow B_4$ en el Cuadro 3.5). Cuando se sospecha que la $Base$ actual B_0 está cerca de la $Base$ óptima, o al menos el tamaño de la lista es pequeño, los movimientos

serán hacia *Bases* próximas con respecto a G_{H2} (por ejemplo, $B_0 \rightarrow B_1$). La nueva *Base* B' será el centro del entorno $N_{H2}(B)$ donde comienza la siguiente búsqueda. Afortunadamente, esta estrategia cubre el espacio de búsqueda completo, aún con permutaciones sencillas, pero en ocasiones es muy lenta. La búsqueda da muchos pasos con pequeñas mejoras.

Este esquema presenta cierta similitud con el recocido simulado (Kirkpatrick et al., 1983), el cual podría ajustarse a nuestro marco de trabajo.

Ejemplo 3.8. Consideremos el siguiente ejemplo con 4 atributos binarios $\{X, Y, U, V\}$ y un conjunto de 4 posibles respuestas $\{0, 1, 2, 3\}$. Tomemos $B_0 = [V, U, X, Y]$. Las 6 *Bases* en $N_{H2}(B_0)$ son:

$$B_1 = [U, V, X, Y] \quad B_2 = [X, U, V, Y]$$
$$B_3 = [Y, U, X, V] \quad B_4 = [V, X, U, Y]$$
$$B_5 = [V, Y, X, U] \quad B_6 = [V, U, Y, X]$$

Cuando nos movemos de B_0 a B_i, los tipos de *XBase* son: 1 hacia B_1, 2 hacia B_6, 3 hacia B_4, 5 hacia B_3, una mezcla de 1 y 3 hacia B_2, y una mezcla de 2 y 3 hacia B_5.

La Figura 3.7 muestra el vector respuesta de los 16 casos según todas las diferentes *Bases* en $N_{H2}(B_0)$ y los espectros correspondientes. Como veremos, las *Bases* óptimas no están dentro de $N_{H2}(B_0)$.

Con B_1, B_2, B_4 y B_6, el número de ítems de la lista *KBM2L* es 16. Por tanto, B_0 debería mejorar mediante un movimiento hacia B' tal que $G_{H2}(B_0, B')$ es grande (posiblemente un *XBase* de tipo 5). Los valores de G para este primer conjunto de *Bases* se muestran en el Cuadro 3.6. De este cuadro, elegimos B_3 como B' ya que G_{H2} es máxima.

Cuadro 3.6: Valores de G para cada par en $N_{H2}(B_0)$ ($\delta = 4$)

G	B_0	B_1	B_2	B_3	B_4	B_5	B_6
$[VUXY] = B_0$	0	1	3	5	2	4	3
$[UVXY] = B_1$	1	0	4	4	3	5	3
$[XUVY] = B_2$	3	4	0	5	3	7	5
$[YUXV] = B_3$	5	4	5	0	7	5	5
$[VXUY] = B_4$	2	3	3	7	0	4	4
$[VYXU] = B_5$	4	5	7	5	4	0	4
$[VUYX] = B_6$	3	3	5	5	4	4	0

Ahora con B_3, la *KBM2L* tiene 8 ítems e intentamos otros movimientos. El Cuadro 3.6 no es

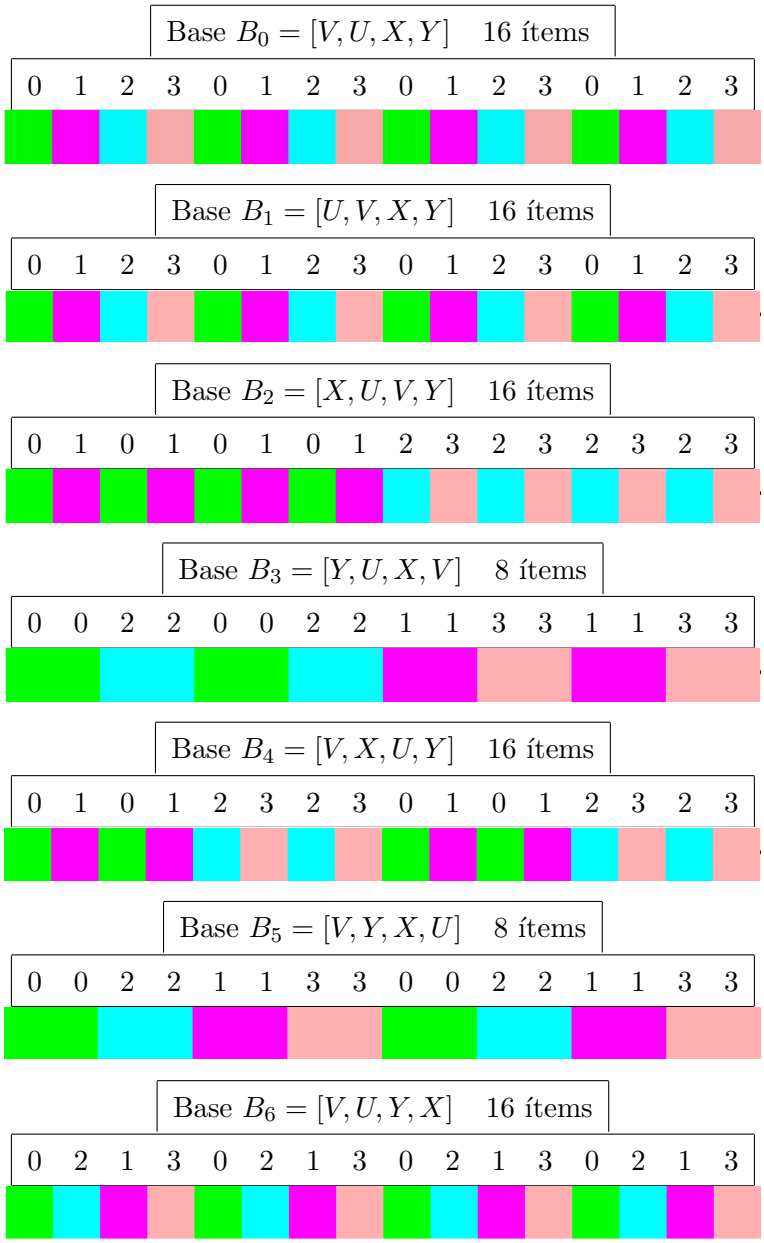

Figura 3.7: Vector de respuestas en las diferentes *Bases* del entorno $N_{H2}(B_0)$

necesario calcularlo otra vez porque aunque B_3 ahora se convierte en B_0, los valores de G para $N_{H2}(B_0 = [Y, U, X, V])$ pueden obtenerse re-etiquetando los atributos. Las 6 *Bases* del nuevo entorno $N_{H2}(B_0)$ son:

$$B_1 = [U, Y, X, V] \quad B_2 = [X, U, Y, V]$$
$$B_3 = [V, U, X, Y] \quad B_4 = [Y, X, U, V]$$
$$B_5 = [Y, V, X, U] \quad B_6 = [Y, U, V, X]$$

El siguiente movimiento debería ser hacia una *Base* B' tal que $G_{H2}(B_0, B')$ sea pequeño, como B_4 o B_1. Después de explorar ambas, B_4 es la mejor. Ésta es una *Base* óptima, dado que la lista alcanza una longitud de 4 ítems con 4 modalidades de respuesta, tal y como vemos en la Figura 3.8.

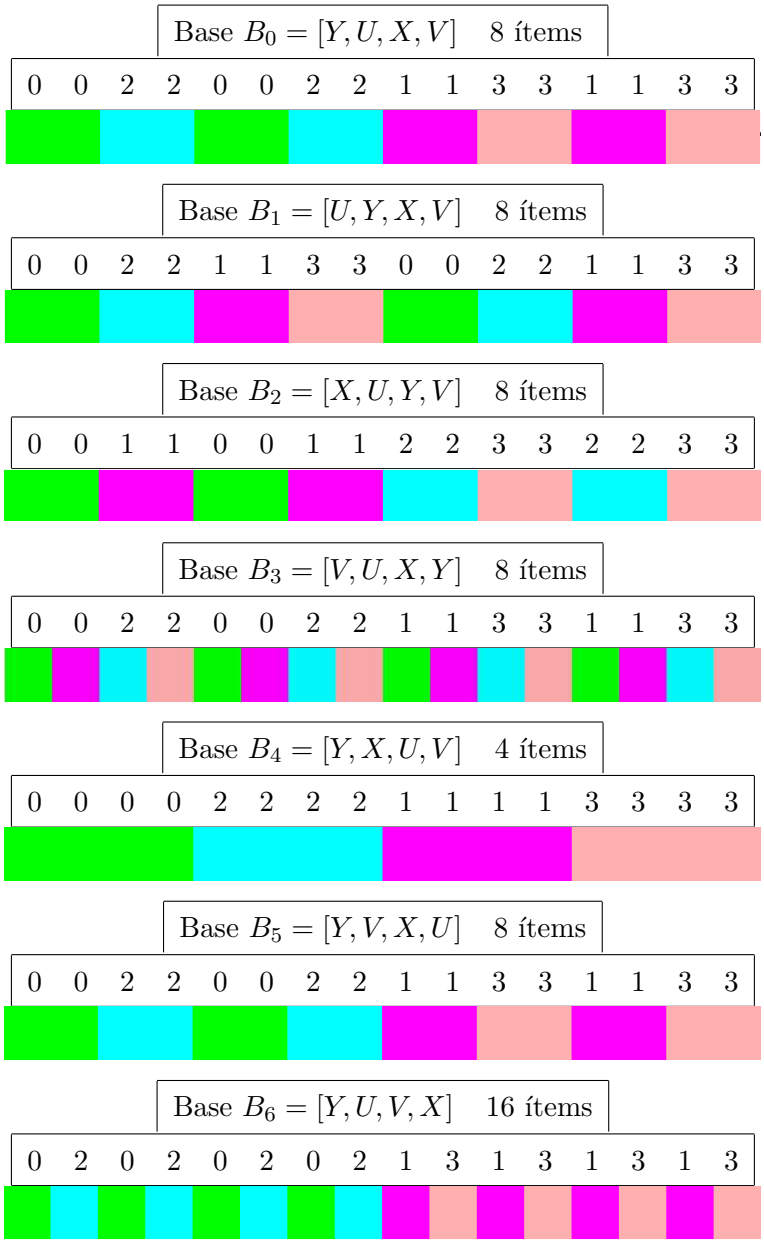

Figura 3.8: Vector de respuestas en las diferentes bases del entorno $N_{H2}(B_0)$

□

El siguiente ejemplo es bastante realista en el sentido de que tiene muchos atributos y muchos

pasos en la búsqueda hasta alcanzar el óptimo.

Ejemplo 3.9. En este caso tenemos 11 atributos binarios, denotados con números desde 0 a 10. El conjunto de alternativas de respuesta es $\{0, 1, 2\}$. Hay 256 casos desconocidos de un total de 2048 casos (es decir, conocemos 1792 casos). Las Figuras 3.9, 3.10 y 3.11 muestran todos los pasos para alcanzar la *Base* óptima B_{20}, que comentamos a continuación.

Base	Items	CMRS	$G_{H2}(B_{i-1}, B_i)$
B_0:[10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0]	1744	1503	-
Base	Items	CMRS	$G_{H2}(B_{i-1}, B_i)$
B_1:[3, 9, 8, 7, 6, 5, 4, 10, 2, 1, 0]	1680	1423	13
Base	Items	CMRS	$G_{H2}(B_{i-1}, B_i)$
B_2:[0, 9, 8, 7, 6, 5, 4, 10, 2, 1, 3]	1359	1086	19
Base	Items	CMRS	$G_{H2}(B_{i-1}, B_i)$
B_3:[4, 9, 8, 7, 6, 5, 0, 10, 2, 1, 3]	1304	1008	11
Base	Items	CMRS	$G_{H2}(B_{i-1}, B_i)$
B_4:[1, 9, 8, 7, 6, 5, 0, 10, 2, 4, 3]	936	551	17
Base	Items	CMRS	$G_{H2}(B_{i-1}, B_i)$
B_5:[0, 9, 8, 7, 6, 5, 1, 10, 2, 4, 3]	847	494	11
Base	Items	CMRS	$G_{H2}(B_{i-1}, B_i)$
B_6:[0, 3, 8, 7, 6, 5, 1, 10, 2, 4, 9]	419	262	18

Figura 3.9: Espectros de la *KBM2L* (I)

Todos los movimientos son hacia *Bases* en $N_{H2}(B)$, donde *Base* es la *Base* actual en cada paso. En este ejemplo, $1 \leq G \leq 35$, y $1 \leq G_{H2} \leq 19$. Como hay muchos ítems en los tres primeros pasos, nuestro software sólo representa en el espectro el cambio de ítem sobre el eje

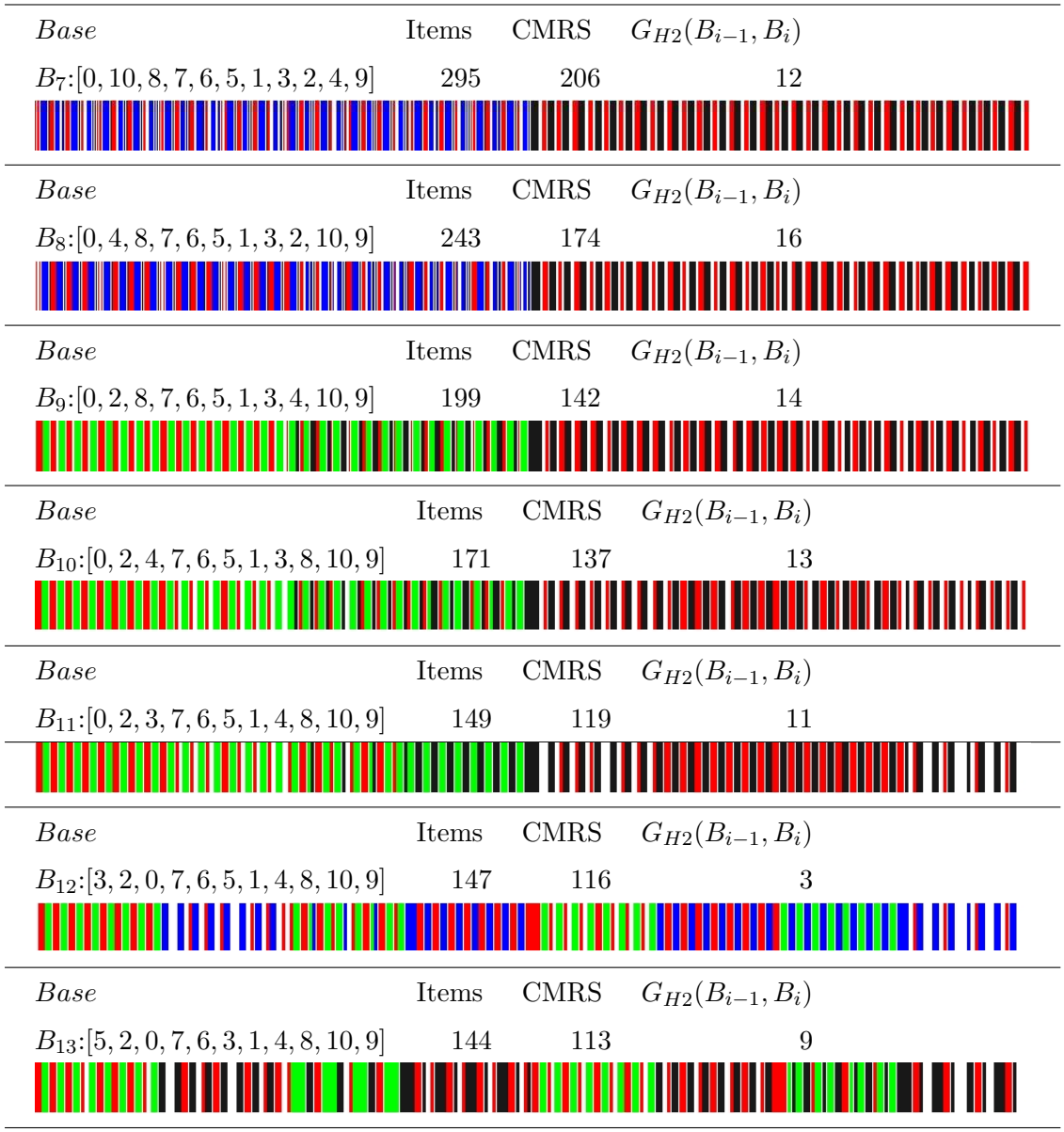

Figura 3.10: Espectros de la *KBM2L* (II)

X, no el número de casos en cada ítem. A partir de B_4 en adelante, el espectro es el usual. Los casos desconocidos (unKB) se representan en blanco.

Observamos que G_{H2} es alta al comienzo, cuando la *Base* actual está lejos de la *Base* óptima. Los *XBase* $B_{11} \rightarrow B_{12}$ y $B_{14} \rightarrow B_{15}$ muestran que un valor de G bajo (= 3) produce una mejora pequeña ($\sim$ 1 %) en la lista. El *XBase* $B_5 \rightarrow B_6$ muestra que un valor de G alto (= 18) produce una mejora mayor ($\sim$ 50 %). Obsérvese también el *XBase* $B_0 \rightarrow B_1$ con $G_{H2}(B_0, B_1) = 13$ y $B_1 \rightarrow B_2$ con $G_{H2}(B_1, B_2) = 19$. Alguna mejora se ha alcanzado, pero el valor $G_{H2}(B_0, B_2) = 12$ muestra que B_2 está más próxima que B_1 a B_0. Esto sugiere una

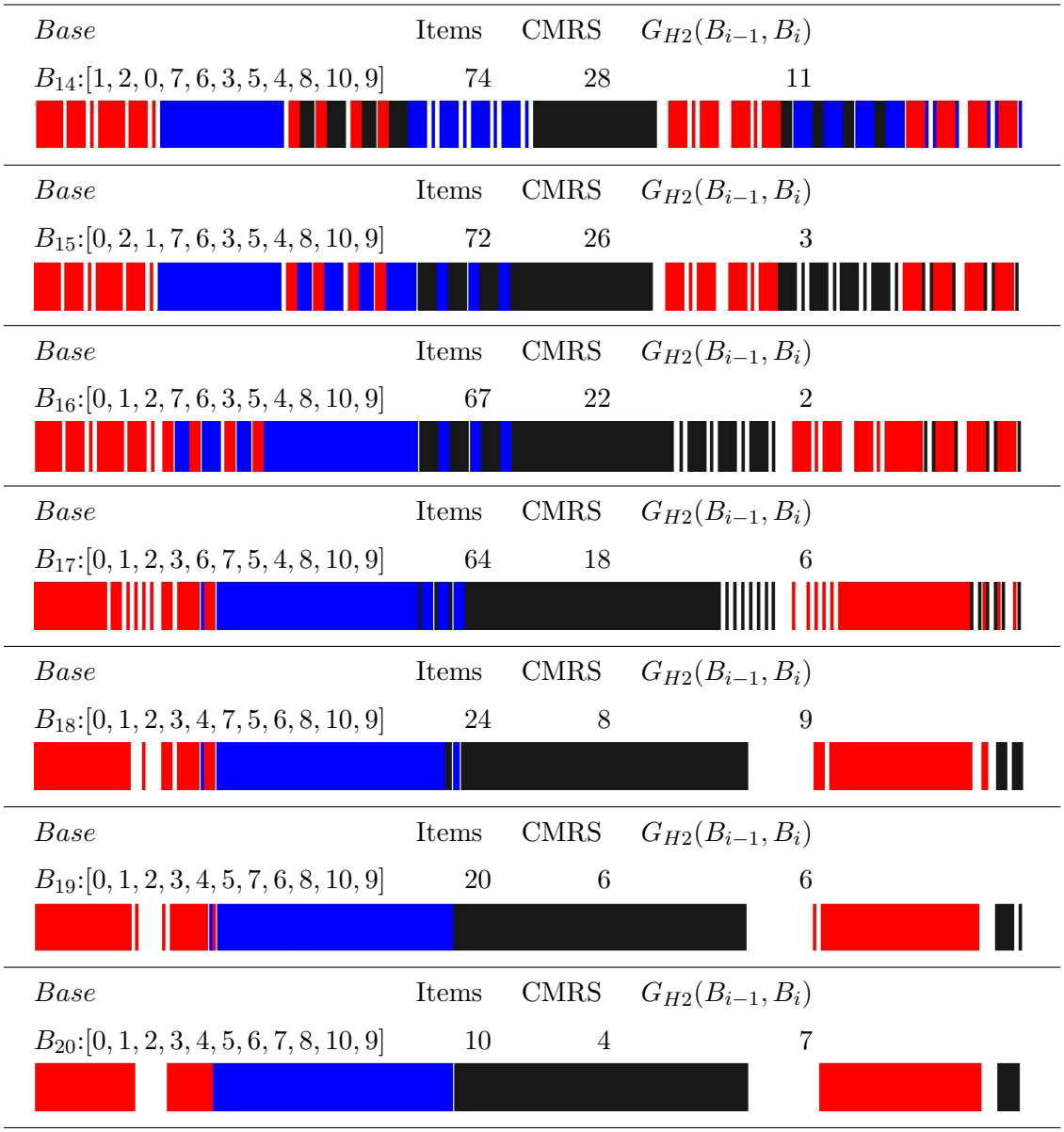

Figura 3.11: Espectros de la *KBM2L* (III)

explicación acerca de la magnitud de la mejora. En esta ocasión la mejora es pequeña, pues aunque la *Base* B_2 no está próxima a B_1, no se aleja de B_0, antes bien se aproxima.

En tanto que el algoritmo progresa, G_{H2} se hace menor. $B_{19} \rightarrow B_{20}$ muestra que valores medios ($= 7$) de G producen una gran mejora ($\sim 50\,\%$).

La *Base* óptima es B_{20}, con 10 ítems. En este ejemplo sabemos que es óptima porque hemos comprobado el CMRS de todas las permutaciones ($11! = 39916800$), pero con dimensiones mayores es imposible. □

De este modo podemos mejorar la traza del algoritmo evitando $XBase$ que mejoran poco, análogamente a la idea de los algoritmos de hormigas (Dorigo et al., 1999), que consideran parte de la traza como una fuente de conocimiento útil para la búsqueda. En nuestro caso se concreta en tener en cuenta cierto número de pasos previos (traza parcial) y aplicar las estrategias correspondientes:

1. En la fase inicial, con una lista muy fragmentada, escogemos primero las $Bases$ del entorno que se alejen de todas las anteriores.

2. En la fase intermedia, con una lista que presenta una moderada coalescencia, escogemos las $Bases$ del entorno próximas a las últimas $Bases$ de la traza pero que no se aproximen a las primeras $Bases$ de la traza.

3. En la fase avanzada, con una lista que presenta una gran coalescencia, escogemos las $Bases$ del entorno próximas a las que hemos considerado recientemente.

Por último, observamos cómo el número de CMRS disminuye al avanzar la búsqueda. Esto implica que la búsqueda se hace más eficiente, pues es más fácil descartar $Bases$. Con listas *KBM2L* malas, el algoritmo debe evitar comprobar muchas $Bases$ que probablemente sean malas (próximas a la inicial), a costa de calcular sólo valores de G, que es muy fácil.

Pseudocódigo del Algoritmo de Entorno Variable (AEV) A continuación mostramos el algoritmo de entorno variable basado en G_{H2}.

```
Fijamos las constantes pequeño, grande, PEQUEÑO y GRANDE

Asignamos la lista de trabajo a LT

Asignamos LT.longitud a Items y LT.Base a B_0

  Repetimos /*Iteración*/ Para todas las Bases tal que Distancia Hamming(B_0, B) = 2 (Att_i
y Att_j han permutado) hacer

0. Si (Items es GRANDE) ∧ (G_H2(B_0, B) es pequeño) ∨
     (Items es PEQUEÑO) ∧ (G_H2(B_0, B) es grande)
     siguiente(B)

1. En otro caso si (Att_i es relevante ∨ Att_j es irrelevante) siguiente(B)

2. Test de la nueva Base B → CMRS

3. Si (CMRS > Items) siguiente(B)

4. En otro caso B → nuevaL.Base /* Lista destino */
```

5. `Copia LT en nuevaL /*` $XBase$`*/`

6. `Si (nuevaL.longitud <  Items) hacer nuevaL →  LT and LT.Base →` B_0

El Paso 0 explora *Bases* lejos de (cerca de) la solución actual cuando estamos lejos de (cerca de) la solución óptima. Este paso es específicamente la heurística G_{H2} y si no se realiza, la búsqueda utiliza la distancia $H2$, (VNA-H2), la cual no tiene en cuenta el peso de los atributos. La instrucción "siguiente(B)" significa que se rechaza esta *Base*. En el Paso 1, explotamos la relevancia de los atributos en términos de la explicación de la respuesta, véase la sección 3.3.2. Un atributo relevante no debe perder mucho peso y un atributo irrelevante no debe ganar mucho peso. El Paso 2 calcula la cota inferior, CMRS, asociada con la *Base* B, véase la sección 3.3.2. En el Paso 3, la lista nueva es rechazada si tiene un CMRS mayor como consecuecia de la copia de algunos casos. En otro caso, el Paso 4 define una lista destino con la nueva *Base*. El Paso 5 realiza la copia de información o $XBase$. En el Paso 6, la búsqueda continúa dentro del entorno centrado en la nueva *Base*.

Las constantes *pequeño* y *grande* son representativas de los valores extremos que puede tomar G_{H2} en el problema concreto. En el Cuadro 3.5, que trata con *Bases* de 5 atributos, observamos que el rango está entre 0 y 7, y podemos fijar *pequeño* = 2 y *grande* = 5. Por otra parte, las constantes *PEQUEÑO* y *GRANDE* son representativas del número de ítems mínimo y máximo que puede presentar potencialmente la lista en cualquier *Base*. Se trata de establecer un rango tentativo basado en el tamaño de la MM asociada a la lista. El valor *PEQUEÑO* puede ser del orden de decenas de veces el logaritmo del tamaño de la MM y el valor *GRANDE* miles de veces. Con 20 atributos binarios la MM tiene un tamaño de 1048576 y $log(1048576) \sim 14$. Entonces, las constantes pueden ser: $PEQUEÑO \sim 200$ y $GRANDE \sim 3000$. Este aspecto del algoritmo es muy dependiente del problema (número de atributos, magnitud de los dominios, modalidades de respuesta,...) y deben elegirse valores de compromiso que no impidan a la heurística alcanzar una solución satisfactoria en un tiempo aceptable.

3.5. Algoritmo Genético

La aplicación de algoritmos genéticos a problemas de optimización combinatoria está bien documentada en varios dominios (Mitchell, 1998). En estos algoritmos, el espacio de búsqueda es representado como una colección de individuos. Estos individuos codifican las soluciones con cadenas de caracteres (o vectores de números, o matrices), las cuales se denominan frecuentemente genes. En nuestro problema, un individuo es una *Base*, es decir, un orden de los atributos (genes).

El propósito de un algoritmo genético es encontrar en el espacio de búsqueda el individuo con mejor material genético. La calidad de un individuo es medida vía una función de evaluación o ajuste, que hace el papel de función objetivo del problema de optimización. La función de evaluación está relacionada en nuestro caso con el espacio de almacenamiento requerido para la información de la tabla en la *KBM2L* asociada con cada *Base*.

El problema de encontrar la *KBM2L* óptima y la descripción de su solución invitan a plantearse un algoritmo genético como posible método de obtención de soluciones. La *Base* óptima es una cadena de códigos de los atributos del esquema que describe secuencias de atributos (genes), las cuales caracterizan el conocimiento recogido en la tabla. Los ítems, o mejor sus descriptores de índice (I_{inf} e I_{sup}), guían la síntesis de la lista mínima. Evitamos la búsqueda al azar construyendo la *Base* óptima a partir de otras *Bases* conocidas.

La idea de un algoritmo evolutivo para este problema parte del análisis de las partes fija y variable del índice de un ítem, véase la sección 2.4. En este contexto, el intercambio de material genético y la permutación de atributos son operaciones similares, pero el algoritmo genético permite abordar la búsqueda con información global.

Se puede construir una población de *Base* y manejar una medida de optimalidad de las mismas. Las permutaciones locales dentro de cada parte, fija y variable, del índice de una familia no afectan mucho a la fragmentación. La *Base* inicial y la nueva son próximas en el sentido que definimos en la sección 3.4. Luego, estas *Bases* no dan mejora de longitud de la lista. Por tanto, hay que considerar permutaciones que intercambien atributos de ambas partes, véase la clasificación de los *XBase* en la sección 3.4 y en particular el tipo 5. Debemos pensar en operaciones de cruce que tengan en cuenta propiedades del problema y de las listas en las *Bases* que integran la población. Veremos en esta sección que en nuestro problema no disponemos de un criterio de evaluación exacta de las *Bases* de la población que evite realizar copias completas (muy costoso). Por tanto, sería muy interesante introducir información del dominio que mejore la calidad de la población, pues la función de evaluación tiene un alcance limitado. Por último, veremos en la sección 3.6 que las técnicas de simulación de la copia, así como las heurísticas locales y globales deben conjugarse para atacar el problema de optimización.

Como el cálculo exacto del tamaño de la lista *KBM2L* es extremadamente costoso, para los problemas en los que aplicaremos el algoritmo genético, la función de evaluación será una cota inferior de este tamaño (véase el "Test de ítem-CMRS"en la sección 3.3). En cada iteración, sólo será necesario almacenar las *Bases* junto a su función de evaluación, pero no las correspondientes listas completas.

Una vez que se ha elegido una población, se determina la calidad de los individuos y al-

gunos de ellos son seleccionados para producir nuevos individuos, los cuales son añadidos a la población. Para todos los nuevos individuos recién creados, existe una probabilidad (cercana a cero) de mutación o cambio en sus genes. Después de esto, algunos individuos son eliminados de la población para mantener siempre un tamaño de población prefijado. Una iteración de este proceso se llama generación.

Los operadores que definen el comportamiento del algoritmo genético son: selección, cruce, mutación y reducción de la población, véase por ejemplo (Larrañaga et al., 1999). Todos ellos son diseñados para mejorar la calidad media de la población, promoviendo el mantenimiento del mejor material genético, es decir, buenos órdenes para los atributos de la *Base*. Detallamos a continuación dichos operadores.

Criterio de selección/reducción. El criterio de selección utilizado para el emparejamiento y eliminación de individuos intenta evitar una presión selectiva fuerte, manteniendo de esta forma la diversidad en la población como una forma de permitir el movimiento por todo el espacio de búsqueda.

La selección se lleva a cabo de acuerdo a la siguiente probabilidad (una función no lineal) de ser seleccionado

$$prob(rank) = q(1-q)^{rank-1},\ rank = 1, 2, \ldots$$

donde *rank* identifica a cada individuo, es decir, es el puesto que ocupa cada individuo en la población de acuerdo con su calidad (véase el siguiente párrafo) y q es una constante entre 0 y 1.

La selección para producir nuevos individuos necesita previamente ordenarlos de mejor a peor. Así, para el mejor individuo, $rank = 1$. Esta forma de selección previene que los mejores individuos monopolicen el proceso de evolución, mientras les da cierta preferencia.

Cuando esta función se usa para descartar individuos de la población, se requiere considerar el orden opuesto, de peor a mejor.

Operador de cruce. Hemos implementado dos operadores de cruce de individuos. El cruce de *corte en un punto* (Banzhaf, 1990) es un operador general que selecciona un punto de corte para dividir los individuos que van a emparejarse en dos partes: f_1 y f_2 del padre y m_1 y m_2 de la madre. Los descendientes se forman mediante el intercambio de estas partes: $f_1 - m_2$ y $m_1 - f_2$. Este operador necesita un mecanismo de reparación, porque algunos atributos pueden repetirse en los descendientes y esto no es correcto en una *Base* válida.

El cruce de *votación* (Fernández del Pozo et al., 2005) es más adecuado para nuestro problema específico. Este operador está basado en el papel que desempeña cada atributo en los

individuos. La idea principal es generar una ordenación de los atributos. Dadas dos *Bases* B_f (padre) y B_m (madre), la puntuación propuesta por ambas *Bases* para cada atributo A_i $(i = 0, 1 \ldots, \delta - 1)$ en la nueva *Base* es:

$$\frac{j}{L_f} + \frac{k}{L_m} = \frac{j \cdot L_m + k \cdot L_f}{L_f \cdot L_m} \tag{3.1}$$

donde j es la posición de A_i en B_f y k en B_m. Los valores L_f y L_m son las longitudes de las respectivas *KBM2L* o sus estimaciones CMRS, véase la sección 3.3. Es decir, L_f y L_m son los valores de la función de evaluación de los individuos a cruzar. Cuanto menor sea la puntuación, más a la izquierda estará el atributo en la *Base* nueva, con mayor peso. Ilustremos los detalles mediante un ejemplo.

Ejemplo 3.10. Sea $B_f = [2, 1, 0, 3, 5, 4]$ con $L_f = 30$ y $B_m = [3, 2, 1, 4, 5, 0]$ con $L_m = 45$. Para el cruce, las *Bases* votan para proponer una nueva *Base* B'. Los atributos tienen las siguientes puntuaciones que se ordenan por medio de (3.1):

$$\begin{array}{llll|llll} A_0\text{:} & \frac{2 \cdot 45 + 5 \cdot 30}{45 \cdot 30} & = 0.177 & & A_1\text{:} & \frac{1 \cdot 45 + 2 \cdot 30}{45 \cdot 30} & = 0.077 \\ A_2\text{:} & \frac{0 \cdot 45 + 1 \cdot 30}{45 \cdot 30} & = 0.022 & & A_3\text{:} & \frac{3 \cdot 45 + 0 \cdot 30}{45 \cdot 30} & = 0.100 \\ A_4\text{:} & \frac{5 \cdot 45 + 3 \cdot 30}{45 \cdot 30} & = 0.233 & & A_5\text{:} & \frac{4 \cdot 45 + 4 \cdot 30}{45 \cdot 30} & = 0.222 \end{array}$$

El mínimo es la puntuación de A_2, seguido de A_1, ..., y finalmente, el máximo es para A_4. Así, la nueva *Base* es $B' = [2, 1, 3, 0, 5, 4]$, donde los empates se resolverían de forma aleatoria.□

Este operador produce un único hijo de cada pareja, padre y madre.

Operador de mutación. El operador de mutación selecciona de forma aleatoria dos atributos que cambian sus posiciones en la *Base*. Este operador se denomina "de intercambio" en la literatura (Goldberg, 1989).

El algoritmo genético se detiene si después de un número fijo (grande) de generaciones, el valor de la función de evaluación no mejora.

Tan sólo se han probado estos operadores para ver si el algoritmo genético era una opción válida, sin haber optimizado todos sus posibles parámetros (operadores, tamaño de la población, tasa de mutación,...). El algoritmo genético propuesto no tiene en cuenta información del dominio que le permitiría generar una población de individuos de mayor calidad. El valor

CMRS como cota de la longitud de la lista puede acelerar la evaluación de las *Bases* y mediante información del dominio (test de importancia de atributos,...) restringir ciertas operaciones de cruce si detectamos que no son adecuadas.

La implementación del algoritmo genético ofrece una infinidad de posibilidades. Por ejemplo, en (Larrañaga et al., 1999) podemos encontrar varios operadores de cruce que no precisan de reparación de la codificación. Están pensados para codificar permutaciones y se pueden aplicar eficientemente a problemas como el que plantea esta tesis o el clásico problema del viajante de comercio.

En problemas de tamaño grande, las estrategias evolutivas (Schewefel, 1981) permitirían desarrollar el segundo nivel de aprendizaje sobre los parámetros del algoritmo genético. Los algoritmos de estimación de distribuciones (Lozano et al., 2006) podrían ser también interesantes, pues trabajan con problemas de optimización combinatoria de complejidad alta.

Pseudocódigo del algoritmo genético El pseudocódigo del algoritmo genético es estándar. Sus operadores principales (selección, cruce, mutación) han sido descritos en esta sección.

```
Generar la población inicial
Calcular la función objetivo para cada individuo
Obtener el mejor individuo
Mientras NO se cumpla la regla de parada
      Seleccionar los padres de la población
      Producir los hijos de los padres seleccionados
      Realizar la mutación de los individuos
      Ampliar la población con los hijos
      Reducir la población extendida a su tamaño inicial
Obtener el mejor individuo encontrado
```

3.6. Experimentos: Comparativa de Estrategias de Optimización

En esta sección realizamos algunos experimentos para probar los algoritmos presentados anteriormente. Nos centramos en el tiempo de ejecución de los algoritmos y en la traza de la solución (el tamaño de la lista en cada paso o iteración) para cada *KBM2L*, usando: (1) el algoritmo de entorno variable (AEV) con la métrica $H2$ (AEV-H2); (2) el AEV con la medida

de proximidad G_{H2} (AEV-G_{H2}); (3) el algoritmo genético (AG) con el operador de votación; y (4) un algoritmo híbrido basado en el AEV-G_{H2} y el algoritmo genético (AG+AEV-G_{H2}). El cuarto algoritmo inicia el AG hasta que alcanza una solución que juzgamos suficientemente buena respecto de la base de partida y lanza entonces una búsqueda local con AEV-G_{H2}.

Usamos un ordenador personal PentiumIII^{TM} a 1-GHz con 512 MB y Windows XP^{TM} y Java^{TM}2 como entorno de programación. Consideramos 10, 90 y 300 atributos A, y dominios con cardinal D igual a 2, 4 y 8, dando lugar a las nueve listas *KBM2L*, k_1, k_2,. . . , k_9, que observamos en el Cuadro 3.7, donde se indican también sus tamaños.

Cuadro 3.7: Descripción de las listas *KBM2L*

	k_1	k_2	k_3
$A = 10$	2^{10}	4^{10}	8^{10}
	k_4	k_5	k_6
$A = 90$	2^{90}	4^{90}	8^{90}
	k_7	k_8	k_9
$A = 300$	2^{300}	4^{300}	8^{300}

El contenido de la lista se genera aleatoriamente en el espacio de desplazamientos como sigue: partimos de la lista vacía y generamos aleatoriamente 20 parejas de desplazamientos que delimitan ítems de respuesta generada también de forma aleatoria. Finalmente se ajusta el número de casos conocidos deseado para que no sea excesivamente grande. El número de casos conocidos de cada tabla se recoge en el Cuadro 3.8 junto al logaritmo en base 2 del cociente entre el número de los casos conocidos y el número total de celdas. El resto de la tabla es desconocido.

Cuadro 3.8: Conocimiento de las listas

	$A = 10$	$A = 90$	$A = 300$
$D = 2$	k_1: 355 (-1.52)	k_4: 243 (-82.0)	k_7: 259 (-291)
$D = 4$	k_2: 267 (-11.9)	k_5: 258 (-171)	k_8: 270 (-591)
$D = 8$	k_3: 222 (-22.2)	k_6: 258 (-261)	k_9: 246 (-892)

Para todas las listas conocemos de antemano la *Base* óptima. El número de ítems para cada tabla óptima es 41. A continuación cambiamos de forma aleatoria la *Base* para intentar encontrar la óptima mediante los algoritmos anteriores. El número de ítems de cada tabla inicial

aparece en el Cuadro 3.9.

Cuadro 3.9: Número de ítems de las listas en la *Base* inicial

	$A = 10$	$A = 90$	$A = 300$
$D = 2$	k_1: 574	k_4: 486	k_7: 518
$D = 4$	k_2: 534	k_5: 516	k_8: 540
$D = 8$	k_3: 444	k_6: 516	k_9: 492

Ahora analizamos los resultados. La Figura 3.12 muestra un estudio comparativo de los cuatro algoritmos, planteados en términos del (logaritmo del) tiempo de ejecución, medido en milisegundos.

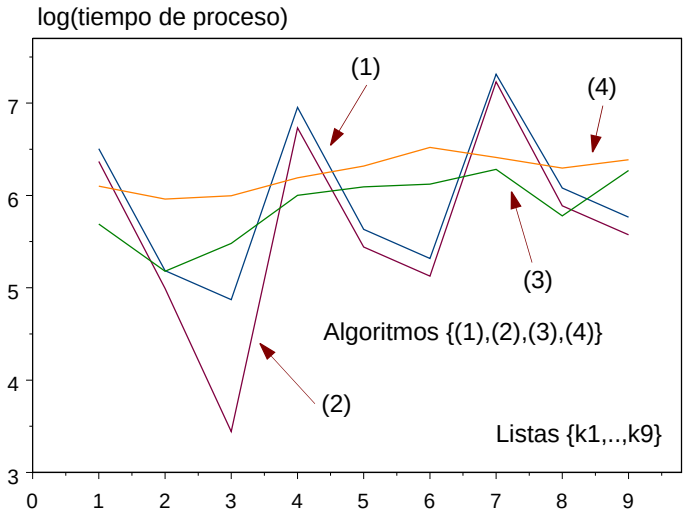

Figura 3.12: Comparativa de resultados del tiempo de ejecución: (1) AEV-H2, (2) AEV-G$_{H2}$, (3) AG y (4) AG+AEV-G$_{H2}$

En lo que respecta al tiempo de ejecución, en general, nuestros experimentos muestran que el AG se comporta mejor que los AEV siempre que hay muchos atributos. Todas las listas, excepto k_1, k_4 y k_9 revelan este resultado, es decir, AEV-G$_{H2}$ es mejor para k_2, k_3, k_5, k_6 y el AG es mejor para k_7 y k_8. La razón detrás de este hecho puede ser la siguiente. Si la dimensión de la tabla crece, el número de miembros del entorno-$H2$ donde se realiza la búsqueda del AEV crece (tiene $\delta\,(\delta - 1)/2$ elementos para δ dimensiones, es decir, un orden cuadrático). Sin embargo, el tamaño de la población del AG se establece usando heurísticas con un orden lineal que es menos complejo: según la evidencia empírica, (Alander, 1992) sugiere un tamaño poblacional entre δ y

2δ. Nosotros, hemos usado δ.

Las Figuras 3.13, 3.14, 3.15 y 3.16 muestran la evolución de la función objetivo (tamaño mínimo de la lista) para las nueve listas conforme el algoritmo progresa. Solamente mostramos los resultados hasta la décima iteración, con el propósito de comparar los tres algoritmos. En la Figura 3.16 mostramos las 10 iteraciones del AG y otras 10 del AEV-G_{H2}. Observamos que para el AG, la función de evaluación es medida en CMRS (a la izquierda de la línea vertical), en lugar de en ítems, tal como explicamos anteriormente. El lado derecho de la línea muestra los verdaderos tamaños de lista, en ítems.

Todos los patrones de evolución son exponenciales decrecientes como consecuencia de la rápida reorganización de la información cuando los atributos relevantes (irrelevantes) ganan (pierden) peso. El ejemplo k_1 muestra una convergencia lenta con las aproximaciones de entorno variable, que es ligeramente más rápida con el AEV-G_{H2}. Este ejemplo podría ser el típico caso que muestra la debilidad de la heurística.

El AEV-G_{H2} es mejor globalmente que el AEV-H2 debido a que G es una medida de comparación de bases refinada. Debemos tener en cuenta que las iteraciones tienen un coste diferente en el AG y el AEV. Es mucho más costosa la primera aproximación: inicialmente consume muchos recursos porque debe trabajar con toda la población, la cual incluye *Bases* malas (véanse los elevados números en el eje Y de los métodos genéticos). También tienen coste diferente porque el AG no calcula el verdadero tamaño de la lista sino una cota inferior. Quizás el AG podría tener un comportamiento mejor si los operadores, especialmente el de cruce, fuesen mejorados para reflejar las características de nuestro problema.

Creemos que la estrategia híbrida (algoritmo AG+AEV-G_{H2}) no merece la pena. Primeramente reproduce el patrón del AG y a continuación una lenta convergencia cuando aplicamos el AEV. Además, en los tiempos de ejecución, se ve siempre superada por el AG.

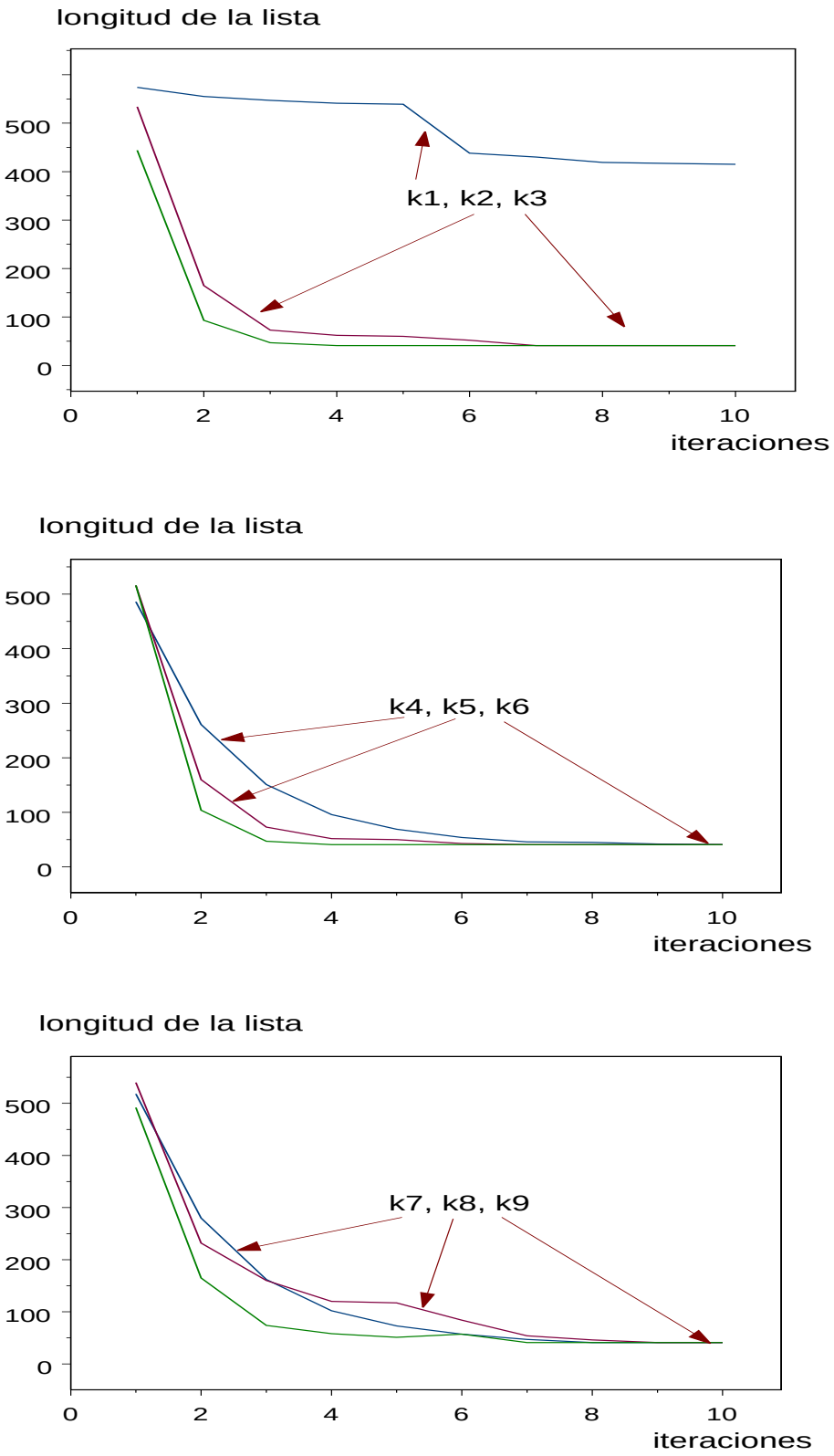

Figura 3.13: Evolución: AEV-H2

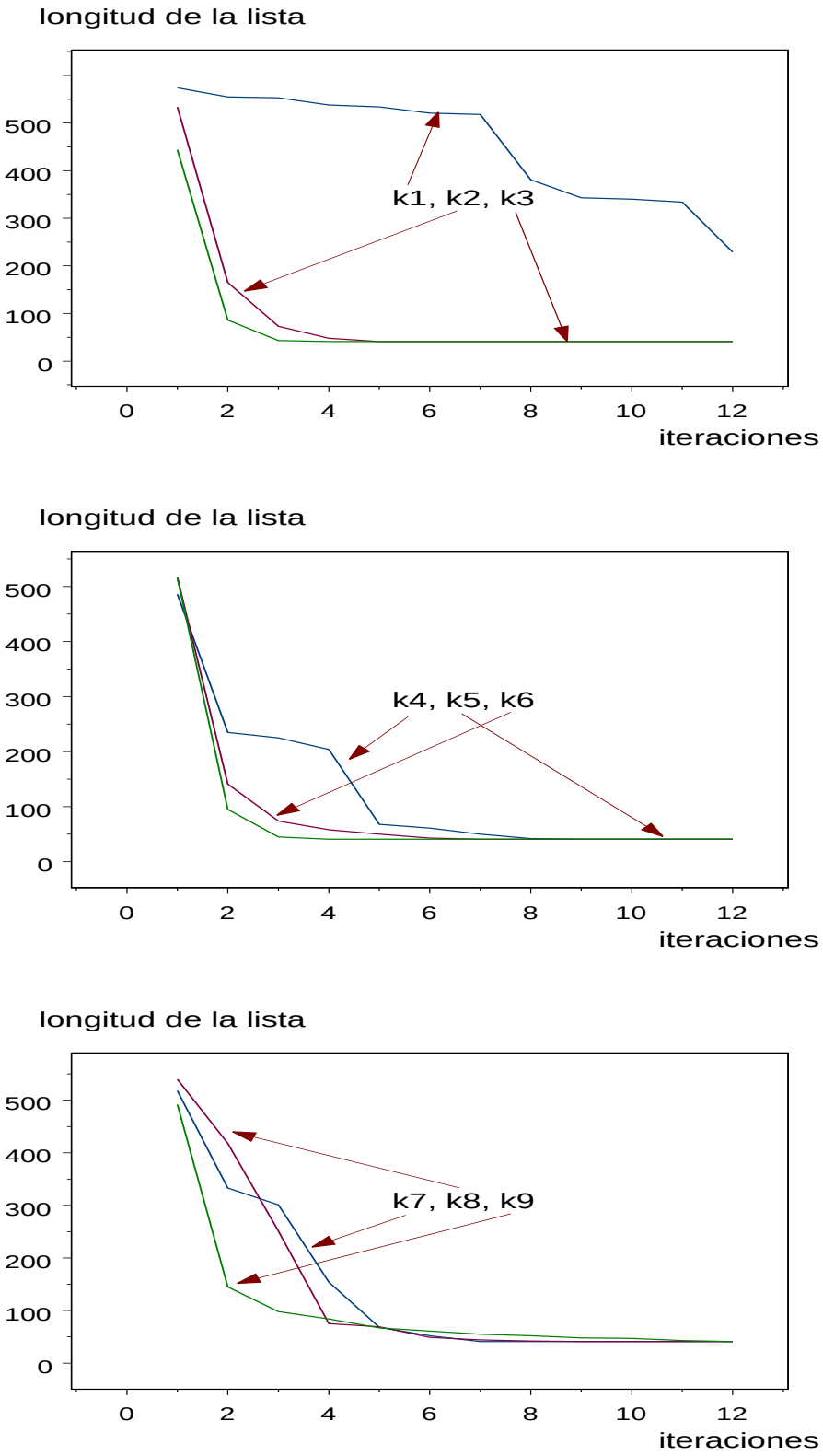

Figura 3.14: Evolución: AEV-G$_{H2}$

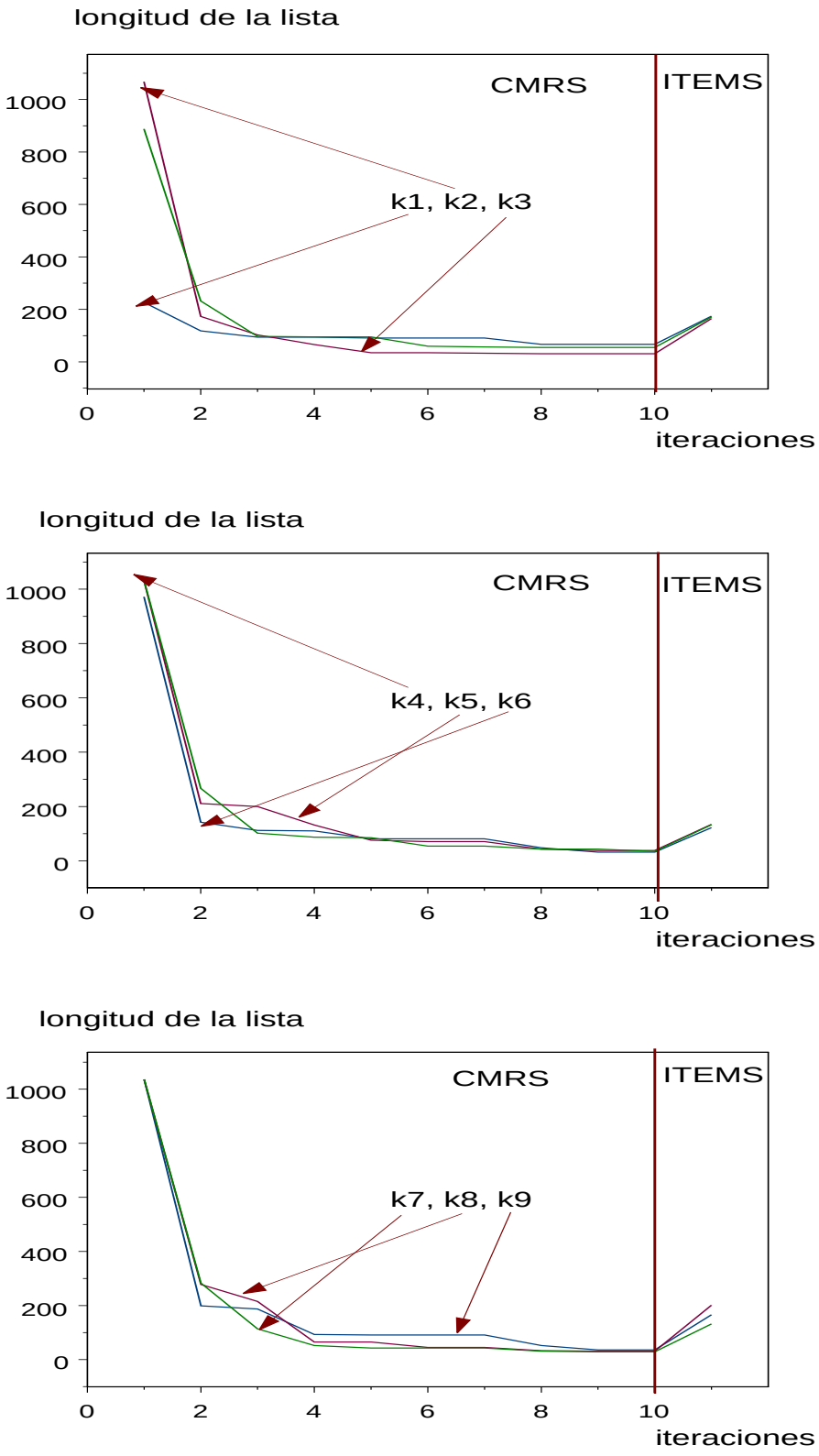

Figura 3.15: Evolución: AG

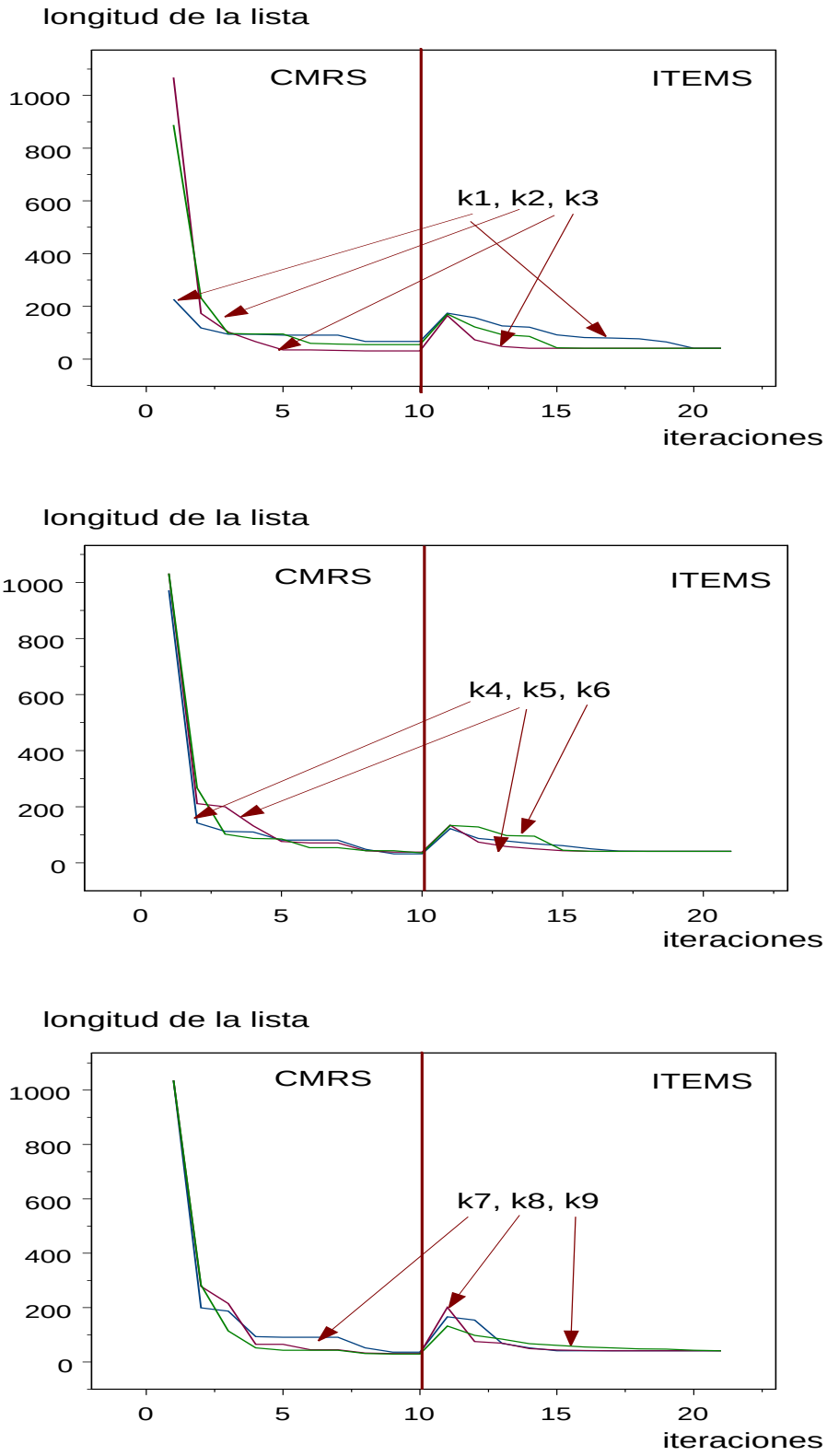

Figura 3.16: Evolución: AG+AEV-G$_{H2}$

Antes de pasar a ver en los siguientes capítulos las aplicaciones, destacamos las siguientes aspectos que han sido desarrollados específicamente para dar un alcance suficiente a las listas *KBM2L* en problemas reales, y algunas cuestiones relativas a la representación del conocimiento:

1. La operación de copia, con su gran coste computacional, se realiza de forma intensiva tanto en el ensayo de soluciones como en la captura inicial del conocimiento previo. La copia rápida y parcial pretenden resolver estas dificultades. La integración de estos mecanismos en modelos de inferencia permite manejar problemas y modelos de complejidad grande o muy grande.

2. La suma del cardinal de los ítems diferentes de *unKB* es el conocimiento de la estructura en el ámbito del modelo. Este número dividido por el tamaño de la MM es la densidad de conocimiento relativo. En los problemas usuales debe ser muy baja por dos razones: la representación de la MM es de tamaño exponencial y la combinatoria implica que haya una enorme cantidad de casos de la tabla imposibles o de escaso interés práctico.

3. El mínimo y máximo del tamaño de todos los ítems conocidos es un indicador de la idoneidad del método para sintetizar o extraer conocimiento, respecto de las *Bases* usadas. El mínimo debe ser próximo a 1 pues si no seguramente habrá algún atributo irrelevante en todos los ítems de la lista y por consiguiente, deberíamos haber simplificado el modelo. El máximo debe ser de un orden próximo al tamaño de la MM pues si no la lista es muy larga y casi no se sintetiza conocimiento.

4. Normalmente el conocimiento debe mostrar una pauta regular en la *Base* óptima. Dos bases distintas pueden presentar tamaños de listas completamente diferentes (en muchos órdenes de magnitud). Puesto que tratamos de comparar las *Bases* respecto de la complejidad de las listas asociadas, introducimos la pseudo–distancia G.

5. Por construcción de los ítems, su cardinal o número de casos está en relación directa al número de índices que deben desarrollarse para recorrer el ítem. Por otra parte, está en relación inversa al número de índices que están fijos. Estos índices, sus valores concretos que direccionan el contenido del ítem particular, pueden verse como explicación del conjunto de casos.

6. Una consecuencia inmediata del apunte anterior es que si en un conjunto de ítems de igual contenido aparecen todos los valores de un índice, el atributo asociado no es relevante en la explicación del contenido, es decir, la respuesta en el contexto definido por el ítem. Y, por otra parte, encontramos un punto de partida para procedimientos de explicación y

aprendizaje de la *Base* óptima. Poder explicar el conocimiento nos facilita la validación y verificación de los sistemas basados en el conocimento en general, y de las TDO y TPC en particular.

3.7. Otras Estructuras de Datos Afines

Esta sección muestra un análisis comparativo entre los conceptos clásicos de matriz *dispersa* y las representaciones proposicionales como el árbol de clasificación, las reglas de producción, las listas de decisiones y los conjuntos aproximados, frente a la lista *KBM2L*.

3.7.1. *KBM2L* y Matrices Dispersas

Existen métodos de almacenamiento de información bajo estructura matricial cuyo objetivo es ahorrar espacio, fundamentalmente porque se repiten muchos valores del contenido de la matriz, por ejemplo, hay muchos ceros, del orden del 99 % del total (Knuth, 1968). En este caso se almacenan sólo los elementos no nulos junto al índice o desplazamiento. Se puede observar que se trata de un caso particular, muy simple, de lo que plantea la estructura *KBM2L*.

Una matriz almacenada mediante un método *disperso* o mediante una estructura *KBM2L* necesitará diferente cantidad de memoria. El método *disperso* está pensado para matrices de dos o unas pocas dimensiones, para información de dominios donde se dan las condiciones oportunas (matrices diagonales, en forma canónica de Jordan, matrices de adyacencia de grafos, sistemas lineales de restricciones,...). Algunos lenguajes de programación como *FORTRAN*, *C++* o paquetes de software matemático como MATLAB (©1994–2003 MathWorks, Inc.) tienen la posibilidad de utilizar estas matrices *dispersas*. Con dos dimensiones no hay más que dos *Bases* y dos representaciones matriciales: la original y su transpuesta.

Ejemplo 3.11. Veamos un ejemplo sencillo de una matriz de 5×5.

$[A, B]$	1	2	3	4	5
1	1	1	0	0	0
2	0	2	1	0	0
3	0	0	3	1	0
4	0	0	0	4	1
5	0	0	0	0	5

Los 25 elementos de la matriz anterior se pueden representar como una lista mediante el método *disperso* en 9 pares de índices y sus respectivos elementos no nulos.

$$(1,1),1;(1,2),1;(2,2),2;(2,3),1;(3,3),3;(3,4)1,;(4,4),4;(4,5),1;(5,5),5.$$

La lista *KBM2L* en la *Base* $[A,B]$ tiene 13 ítems:

$$\langle(1,1),1|\ \langle(1,2),1|\ \langle(2,1),0|\ \langle(2,2),2|\ \langle(2,3),1|\ \langle(3,2),0|\ \langle(3,3),3|\ \langle(3,4),1|\ \langle(4,3),0|\ \langle(4,4),4|$$
$$\langle(4,5),1|\ \langle(5,4),0|\ \langle(5,5),5|.$$

Si sólo interesa ver si es cero y distinto de cero, se construye la lista *KBM2L* siguiente con 9 ítems, donde 1 indica distinto de 0:

$$\langle(1,2),1|\ \langle(2,1),0|\ \langle(2,3),1|\ \langle(3,2),0|\ \langle(3,4),1|\ \langle(4,3),0|\ \langle(4,5),1|\ \langle(5,4),0|\ \langle(5,5),1|.$$

□

El método ideal depende del problema y de los objetivos que se pretendan. La lista *KBM2L* es más general y pensada para problemas de muchas dimensiones, donde las matrices no tienen que ser *dispersas* y se pueden obtener otras ventajas además del almacenamiento compacto.

3.7.2. *KBM2L* y Representaciones Proposicionales

Vamos a considerar algunas de las principales representaciones proposicionales del conocimiento y las técnicas asociadas a su implementación y metodos básicos de inferencia. La razón es que la tabla es la representación proposicional más simple y de hecho, éstas la consideran como el conjunto de ejemplos o datos para aprender las estructuras correspondientes: árboles de clasificación, reglas de producción, listas de decisiones y conjuntos aproximados.

En el informe (Holsheimer y Siebes, 1994) y en (Mitchell, 1997) podemos encontrar una exposición de estas técnicas en el marco del aprendizaje automático. Aquí tratamos de discutir su parecido y sus diferencias con las listas *KBM2L*.

Los **árboles de clasificación**, también conocidos en aprendizaje automático como árboles de decisión, son una herramienta de representación del conocimiento sencilla que ha tenido éxito en sistemas de clasificación supervisada, que conocen a priori las clases. Por ejemplo, el sistema de Quinlan ID3 (Quinlan, 1986) induce árboles de clasificación.

El árbol clasifica ejemplos en un número finito de clases. Los nodos se etiquetan con nombres de atributos. Los arcos se etiquetan con valores posibles de estos atributos. Las hojas se etiquetan con el nombre de las posibles clases. Un objeto es clasificado mediante el camino descendente en el árbol que coincide con la instancia de sus atributos.

Las **reglas de producción** son conocidas en sistemas expertos por sus ventajas: modularidad y generalización (independencia de los detalles particulares del conjunto de datos de

entrenamiento). Las reglas permiten representar, por ejemplo, en FNC los términos de la condición (if) cuyo consecuente ($then$) es la clase. El sistema AQ15 (Michalski et al., 1986) utiliza esta representación.

Las **listas de decisiones** son una generalización de las representaciones en árbol, la FNC y la FND. Una lista de decisión es una lista de pares: $(\Phi_1, C_1)(\Phi_2, C_2)\ldots(\Phi_r, C_r)$, donde Φ_i es una descripción proposicional elemental, cada C_i es una clase y la última descripción Φ_r es la constante *verdadero*. Un objeto es de la clase C_j si el último índice de una descripción Φ_j se instancia a *verdadero* con los atributos del objeto. Tal índice siempre existe porque la última clase es por defecto. Podemos ver esta representación como una regla extendida: *if* Φ_1 *then* C_1 *elseif* Φ_2 *... else* C_r, o podemos pensar que clasifican objetos dando un patrón genérico de excepciones. Las excepciones las soportan los primeros términos de la lista mientras que el patrón general lo representan los últimos. Las listas de decisiones (Rivest, 1987) son usadas en el sistema CN2 (Clark y Niblett, 1989).

Los **conjuntos aproximados** (CA) han sido introducidos por (Pawlak, 1997) para generar reglas o filtrar la información redundante antes de aplicar otras técnicas de aprendizaje automático. A diferencia de ID3, que utiliza para inducir reglas medidas estadísticas de la teoría de la información, esta técnica está basada en la teoría de conjuntos y en la topología. El criterio de ajuste, llamado de indiscernibilidad, de los conjuntos aproximados es un caso especial del criterio de entropía usado en ID3 (Wong et al., 1986). Mediante los conjuntos aproximados, a partir de un conjunto multivariante de objetos, tratamos de inducir las reglas que los clasifican en ciertas clases. La lista *KBM2L*, por el contrario, está orientada a la deducción. Los aspectos similares de ambas técnicas se concretan en la agrupación de los casos por la *respuesta igual* en *KBM2L* / *relación de clase* en CA y por la *explicación común* en *KBM2L* / *relación de equivalencia* en CA.

La lista *KBM2L* es más estricta al crear un ítem, donde todos los casos son de la misma clase, mientras que la aproximación superior de una clase puede ser heterogénea. Además, nuestra aproximación agrupa casos con la misma clase, igual que la relación de clase de los conjuntos aproximados y con el mismo valor de atributos, igual que la relación de equivalencia de los conjuntos aproximados, aunque de diferente forma. Nuestros grupos son puros, todos los casos proceden de la misma clase, mientras que en los conjuntos aproximados las clases se aproximan vía el subconjunto superior, que es mezcla de casos de varias clases, y el subconjunto inferior.

La representación mediante listas *KBM2L* (Fernández del Pozo et al., 2005) es semejante a los árboles (la jerarquía de atributos), sintetiza reglas (ítems de la lista), y clasifica utilizando un orden estricto (*Base* de la lista) en la formulación de las proposiciones.

Nuestro método descansa en una estructura basada en una lista en lugar de estar basada en un árbol. La búsqueda de los mejores candidatos es global, involucrando al conjunto completo de atributos, mientras que los árboles usan una búsqueda local voraz sobre el espacio de hipótesis, las estructuras de todos los posibles árboles. Evitamos la naturaleza jerárquica del proceso de construcción de los árboles y superamos la típica inestabilidad de los árboles a los pequeños cambios (Hastie et al., 2001). Por otra parte, los árboles de clasificación son muy flexibles y pueden ser utilizados con cualquier tipo de dato (métrico, no métrico o mixto). Nuestra metodología está limitada a datos finitos.

Como explicamos en detalle en (Fernández del Pozo et al., 2005), el método *KBM2L* intenta reorganizar una estructura de conocimiento mediante una búsqueda global de buenas estructuras candidatas. Al principio del algoritmo hay ya una clasificación correcta de los casos representados en la TDO o en la TPC. Los casos pueden interpretarse como tipos representativos de pacientes y proposiciones cuya incertidumbre está cuantificada probabilísticamente, respectivamente, e intentamos desvelar el razonamiento subyacente a esta clasificación. A diferencia de las situaciones típicas en aprendizaje automático, estos casos son únicos ya que se corresponden con configuraciones de variables del DI original. Nuestro método se aplica después de que el DI ha sido resuelto, mientras que la mayoría del trabajo en el marco de los DI (búsqueda de variables relevantes para decisiones) se ocupa de las operaciones en la estructura del grafo antes de la evaluación (Lauritzen y Nilsson, 2001). Afirmamos que ambas estrategias son complementarias.

Una diferencia es que la tabla de datos no es vista como un conjunto de entrenamiento de la posible estructura de la *KBM2L*. La tabla se transforma (toda o parte) en la lista. En las tablas que consideramos, y en las listas asociadas también, todos los atributos son conocidos y están todas las combinaciones posibles del producto cartesiano (la representación es exhaustiva). Además, no se repiten casos pues consideramos que todas las entradas de la tabla son exclusivas. La lista *KBM2L* trata en principio con atributos discretos frente a las otras técnicas que pueden tratar con atributos continuos. Este problema lo abordamos en el capítulo 5. La lista *KBM2L* clasifica los casos mediante la respuesta y reserva espacio para los objetos desconocidos presentando la estructura completa de la BC incompleta.

Nuestra representación no considera que haya ruido en la tabla. Tampoco considera que los casos tengan incertidumbre, aunque la respuesta o algún atributo represente la medida de la imprecisión (véase el capítulo 5), es decir, la probabilidad. Necesita conocer todos los valores de los atributos de un objeto (caso) mientras que en árboles y reglas no es preciso si se dispone de un procedimiento de imputación y/o inferencia, como es el caso en las versiones recientes de los sistemas mencionados, véanse en (Quinlan, 1987; Winston, 1992; Quinlan, 1993; Mitchell, 1997)

los sistemas ID3 y C4.5. La inferencia y el tratamiento de datos incompletos mediante listas *KBM2L* lo abordaremos en cierta medida en el capítulo 4.

La construcción de la lista se resuelve en el espacio de la combinatoria de las posibles *Bases*, análogamente a los árboles y otros espacios de hipótesis del aprendizaje automático. De igual modo que en las otras representaciones, la construcción puede ser incremental, pero la medida de bondad de la estructura no está asociada a los datos (ganancia de información basada normalmente en la entropía) sino a la longitud de la lista. Decimos que la lista de menor longitud es la que mejor representa la tabla, véase el capítulo 3. Además, la clasificación más eficiente se realiza sobre una lista *KBM2L* optimizada globalmente y no mediante un procedimiento parcial que asigna de forma irrevocable la posición y la importancia en el clasificador a los atributos, como en ID3.

Creemos que el sistema tiene las ventajas de los árboles (precisión al clasificar y jerarquización flexible y dinámica de los atributos), de las reglas y los conjuntos aproximados (sencillez) y listas de decisión (eficiencia al generalizar y gestión de excepciones mediante ítems pequeños), como justificaremos a lo largo de la tesis.

Ejemplo 3.12. A continuación presentamos el ejemplo sencillo "weather" (Quinlan, 1993) basado en un material didáctico de (Ingargiola, 2003) sobre construcción de modelos de clasificación, publicado en su página web.

Sea el esquema de atributos recogido en el Cuadro 3.10 relativo al problema de clasificar las condiciones, buenas o malas, de juego del golf. Sea un conjunto de observaciones, recogidas en el Cuadro 3.11, acerca de las condiciones (buenes o malas) para la práctica del golf descritas mediante los atributos anteriores.

Cuadro 3.10: Esquema de atributos

Atributo	Valores	
Tiempo	sol, encapotado, lluvia	nominal
Temperatura	64–85	intervalo
Humedad	65–95	intervalo
Viento	sí, no	dicotómica

En primer lugar, veamos el resultado de aplicar el algoritmo ID3, modificado para manejar atributos continuos, para construir el árbol de clasificación siguiente, Cuadro 3.12, donde el atributo *Temperatura* no es relevante.

A continuación vamos a construir una lista *KBM2L* para la tabla. Previamente debemos

Cuadro 3.11: Conjunto de datos

Tiempo	Temperatura	Humedad	Viento	Golf
sol	85	85	no	malas
sol	80	90	sí	malas
encapotado	83	78	no	buenas
lluvia	70	96	no	buenas
lluvia	68	80	no	buenas
lluvia	65	70	sí	malas
encapotado	64	65	sí	buenas
sol	72	95	no	malas
sol	69	70	no	buenas
lluvia	75	80	no	buenas
sol	75	70	sí	buenas
encapotado	72	90	sí	buenas
encapotado	81	75	no	buenas
lluvia	71	80	sí	malas

Cuadro 3.12: Árbol de clasificación, ID3

		Tiempo		
encapotado		sol		lluvia
		Humedad		Viento
	≤ 75	> 75	sí	no
buenas	buenas	malas	malas	buenas

realizar dos transformaciones: **discretizar** los atributos continuos y añadir una Dod. La primera transformación se debe a la naturaleza de la representación de los datos mediante listas que sólo contempla de momento atributos discretos. La discretización implementada es un tanto arbitraria pero se trata de un ejemplo sencillo que no pretende abordar en profundidad todos los aspectos de la representación. En el capítulo 5 veremos el tema, en absoluto trivial, de la discretización de la escala de atributos. La segunda transformación pretende diferenciar las observaciones idénticas respecto de los atributos (Tiempo, Temperatura, Humedad, Viento) pero que tienen

diferente respuesta (malas o buenas condiciones de juego del golf). La discretización es una de las causas de que aparezcan casos iguales con respuesta diferente. Otras podrían ser el ruido, inconsistencias,...

Por otra parte, también es interesante tener en cuenta la frecuencia de los datos multidimensionales, es decir, casos iguales con respuesta igual. La frecuencia o repetición de los datos es registrada mediante una dimensión adicional que llamamos de **observación**. Así evitamos que al insertar los casos en la lista se sobreescriban las respuestas o se pierda información sobre la frecuencia. El dominio de la dimensión de observación es variable y se ajusta al mínimo necesario para evitar la pérdida de información asociada a la sobreescritura. La dimensión de observación, si los datos no tienen carácter temporal es irrelevante y por eso le asignamos peso igual a 1. En el ejemplo basta con 3 valores: $\{x0, x1, x2\}$.

El esquema (y codificación) utilizado se muestra en el Cuadro 3.13.

Cuadro 3.13: Codificación del esquema de atributos y la respuesta

	Atributo	Valores
0	Tiempo	sol (0), encapotado (1), lluvia (2)
1	Temperatura	60–79 (0), 80–89 (1)
2	Humedad	60–79 (0), 80–99 (1)
3	Viento	no (0), sí (1)
4	Observación	x0 (0), x1 (1), x2 (2)
	Respuesta	Valores
	Condiciones	malas (0), buenas (1)

Transformamos la tabla en una lista *KBM2L*. La tabla tiene: 72 celdas o posiciones, 21 ítems y dos modalidades de respuesta conocida. La moda es *unKB*, hay 10 ítems desconocidos y hay 58 casos desconocidos. Hay 11 ítems conocidos: 6 con respuesta 1 y 5 con respuesta 0. Hay 14 casos conocidos: 9 con respuesta 1 y 5 con respuesta 0. La *Base* es: $[0, 1, 2, 3, 4]$ y el vector de pesos es $(24, 12, 6, 3, 1)$, véase la Figura 3.17.

Optimizamos la lista, que consiste en configurar el orden de los atributos de la *Base* que minimiza el número de ítems. Si la lista está muy vacía el número de ítems es igual en casi todas las *Bases* y reconsideramos el objetivo de la optimización (véase el capítulo 6). Ahora, minimizamos el número de ítems adyacentes xy, que tienen diferentes respuestas conocidas y están separados por un ítem con respuesta desconocida, y dualmente (si no coalescencias de ítems) maximizamos el número de ítems adyacentes xx con igual respuesta conocida y están

separados por un ítem con respuesta desconocida. La lista anterior presenta $|xx| = 5$ y $|xy| = 4$. Ahora la nueva lista presenta $|xx| = 7$ y $|xy| = 2$. La *Base* óptima es: $[2, 0, 1, 3, 4]$. El vector de pesos es $(36, 12, 6, 3, 1)$, véase la Figura 3.17.

Generalizamos la respuesta de los ítems desconocidos a partir de los adyacentes si es igual o no existe, como en el caso de los ítems extremos. La lista inferida tiene 6 ítems, la moda es 1, es decir, las condiciones buenas de juego. Hay lógicamente $|xx| = 0$ y el mínimo es $|xy| = 2$. Hay 4 ítems conocidos: 2 con respuesta 1 y 2 con respuesta 0. Hay 60 casos conocidos: 32 con respuesta 1 y 28 con respuesta 0. La *Base* es la óptima pues $|xx|$ no puede ser menor que 0. Véase la Figura 3.17.

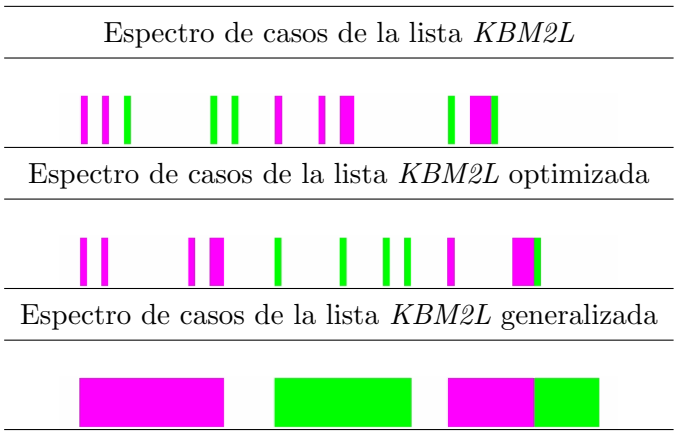

Figura 3.17: Espectros de la lista inicial, optimizada y generalizada

La lista *KBM2L* resultante queda recogida en el Cuadro 3.14 junto a las correspondientes explicaciones de los ítems. Tenemos 32 casos de buenas condiciones, 28 de malas condiciones y 12 desconocidos. Recordemos que si la respuesta es desconocida la explicación es vacía. La *Base* óptima $[2, 0, 1, 3, 4]$, es decir [Humedad, Tiempo, Temperatura, Viento, Observación], refleja un orden de importancia de los atributos.

Comparamos la lista *KBM2L* y las reglas derivadas del árbol de clasificación obtenido mediante el algoritmo ID3, del Cuadro 3.12. ID3 produce 5 reglas y la *KBM2L* 4 ítems, que representan a 60 casos conocidos. En el árbol, el Tiempo es el atributo más relevante o informativo y después la Humedad y el Viento. En la lista, la Humedad y el Tiempo son los atributos más importantes del problema de clasificación de los posibles escenarios de juego del golf, y el Viento y la Temperatura no son relevantes, véase el Cuadro 3.14. El atributo asociado a la dimensión de la frecuencia de observación puede ser relevante pero no debería ser más importante y tener más peso que cualquier otro. El ítem conocido más pequeño es el 5. En su parte variable están: Temperatura, Viento y Observación. Puesto que Temperatura tiene un dominio binario no es

un ADR y no es relevante. Los otros dos atributos de peso inferior a Temperatura recorren sus respectivos dominios en todos los ítems conocidos y por consiguiente, no son relevantes.

El árbol clasifica explícitamente 5 casos de 12, los que se derivan de considerar 3 estados de Tiempo, 2 estados de Humedad, 2 estados de Viento y la irrelevancia de la Temperatura. La lista clasifica 60 de 72, los que se derivan de considerar 3 estados de Tiempo, 2 estados de Humedad, 2 estados de Viento, 2 estados de Temperatura y 3 valores en la dimensión de observación. Es decir, la lista que hemos construido clasifica un 83.3 % frente a un 41.6 % del árbol de clasificación.

Cuadro 3.14: Lista *KBM2L* y explicaciones para el problema del golf

ítem	índice	desplazam.	tamaño	explicación
0	$\langle(0,1,1,0,1),1\vert$	$\langle 19,1\vert$	20	1 → Humedad 60–79, Tiempo sol o encapotado
1	$\langle(0,2,0,0,2),-1\vert$	$\langle 26,-1\vert$	7	$\emptyset$
2	$\langle(1,0,1,1,0),0\vert$	$\langle 45,0\vert$	19	0 → Humedad 60–79, Tiempo sol o lluvia
3	$\langle(1,1,0,0,2),-1\vert$	$\langle 50,-1\vert$	5	$\emptyset$
4	$\langle(1,2,0,0,2),1\vert$	$\langle 62,1\vert$	12	1 → Humedad 80–99, Tiempo encapotado o lluvia
5	$\langle(1,2,1,1,2),0\vert$	$\langle 71,0\vert$	9	0 → Humedad 80–99, Tiempo lluvia

El Cuadro 3.15 muestra la comparación de respuesta que ofrece el árbol de clasificación ID3 y la lista *KBM2L* (optimizada y generalizada) para 24 casos posibles del problema discretizado. Asumimos que el valor de Humedad 0, que codifica el intervalo $(60, 79)$ en la lista, se corresponde con el rango de Humedad menor que 75 que define el árbol. Se manifiesta discrepancia en los casos (Humedad=0, Tiempo=2, Temperatura=1, Viento=0) y (Humedad=1, Tiempo=2, Temperatura=1, Viento=0). El árbol clasifica ambos casos como condiciones buenas de juego y la lista como malas. La lista desconoce la clasificación de los casos: (Humedad=0, Tiempo=1, Temperatura=1, Viento=1), (Humedad=0, Tiempo=2, Temperatura=0, Viento=0) y (Humedad=1, Tiempo=1, Temperatura=0, Viento=0). Dichos casos son buenas condiciones de juego para el árbol. Los casos desconocidos pueden resolverse en la lista mediante proximidad a los ítems conocidos adyacentes. De este modo, en el primero y tercero, la lista y el arbol están de acuerdo y en el segundo discrepan. En resumen, ambos clasificadores discrepan en 3 casos, el

12.5 %.

Cuadro 3.15: Árbol de clasificación y lista *KBM2L*, comparación de la clasificación de casos (Humedad, Tiempo, Temperatura, Viento)

caso	árbol ID3	lista *KBM2L*
(0,0,0,0)	1	1
(0,0,0,1)	1	1
(0,0,1,0)	1	1
(0,0,1,1)	1	1
(0,1,0,0)	1	1
(0,1,0,1)	1	1
(0,1,1,0)	1	1
(0,1,1,1)	1	-1 → 1
(0,2,0,0)	**1**	-1 → **0**
(0,2,0,1)	0	0
(0,2,1,0)	**1**	**0**
(0,2,1,1)	0	0
(1,0,0,0)	0	0
(1,0,0,1)	0	0
(1,0,1,0)	0	0
(1,0,1,1)	0	0
(1,1,0,0)	1	-1 → 1
(1,1,0,1)	1	1
(1,1,1,0)	1	1
(1,1,1,1)	1	1
(1,2,0,0)	1	1
(1,2,0,1)	0	0
(1,2,1,0)	**1**	**0**
(1,2,1,1)	0	0

La discretización del dominio de los atributos continuos condiciona el resultado. Si consideramos ahora una discretización ampliada como la que recoge el Cuadro 3.16 obtenemos la lista óptima generalizada *KBM2L* del Cuadro 3.17 en la misma *Base* que resultó anteriormente. La lista tiene 173 casos conocidos en 5 ítems: 120 casos con respuesta 1 y 53 con respuesta 0. La

moda es 1, condiciones buenas de juego, y clasifica el 80 % del espacio de representación. El espectro resultante se muestra en la Figura 3.18.

Figura 3.18: Espectros de la lista generalizada para una discretización ampliada

Cuadro 3.16: Codificación del esquema de atributos y la respuesta, discretización ampliada

	Atributo	Valores
0	Tiempo	sol (0), encapotado (1), lluvia (2)
1	Temperatura	60–69 (0) 70–79 (1), 80–89 (2)
2	Humedad	60-69 (0) 70–79 (1), 80–89 (2) 90–99 (3)
3	Viento	no (0), sí (1)
4	Observación	x0 (0), x1 (1), x2 (2)
	Respuesta	**Valores**
	Condiciones	malas (0), buenas (1)

Los Cuadros 3.18 y 3.19 muestran la comparación de respuesta que ofrece el árbol de clasificación ID3 y la lista *KBM2L* (optimizada, generalizada) para 24 casos posibles del problema discretizado ampliado. El valor de Humedad=1 se corresponde con el intervalo (70,79) y por tanto con el Tiempo=0, sol. El árbol no obtiene la clasificación directamente y debe inferir la más probable, que es de buenas condiciones. En este caso ambos clasificadores discrepan en 15 casos, el 20.8 %.

Las dos listas son equivalentes respecto del conocimiento. La segunda es más precisa en la representación pero más costosa de construir, almacenar y gestionar. En la segunda, la Temperatura es relevante, según la explicación del ítem 4, buenas condiciones → Humedad 70–79, Tiempo encapotado y Temperatura 60–79, con 7 casos. Si vemos el ítem 4 con mayor detalle observamos un solo caso de Temperatura 70–79 frente a 6 de Temperatura 60-69. Aunque en el árbol ID3 no es relevante la Temperatura aquí vemos la relación entre atributos: Humedad, Tiempo y Temperatura no son independientes. El árbol refleja la clasificación pero no las relaciones en el esquema.

La inferencia de un caso en el árbol consiste en recorrer el camino que coincide con los valores que toman los atributos de dicho caso y asignar la etiqueta de la hoja correspondiente. Si hay atributos desconocidos se estima la probabilidad de los posibles resultados.

Cuadro 3.17: Lista *KBM2L* y explicaciones, discretización ampliada

ítem	índice	desplazam.	tamaño	explicación
0	$\langle(1,1,2,0,1),1\vert$	$\langle 85,1\vert$	86	1 → Humedad 60–69 o 70–79,
1	$\langle(1,2,0,0,2),-1\vert$	$\langle 92,-1\vert$	7	$\emptyset$
2	$\langle(2,0,2,0,0),0\vert$	$\langle 120,0\vert$	28	0 → Humedad 70–79 o 80–89
				Tiempo sol o lluvia
3	$\langle(2,1,2,1,2),-1\vert$	$\langle 143,-1\vert$	23	$\emptyset$
4	$\langle(2,2,1,0,0),1\vert$	$\langle 150,1\vert$	7	1 → Humedad 70–79,
				Tiempo encapotado,
				Temperatura 60–69 o 70–79
5	$\langle(2,2,1,0,2),-1\vert$	$\langle 152,-1\vert$	2	$\emptyset$
6	$\langle(3,0,2,1,0),0\vert$	$\langle 177,0\vert$	25	0 → Humedad 80–89 o 90–99,
				Tiempo sol o lluvia
7	$\langle(3,1,1,0,2),-1\vert$	$\langle 188,-1\vert$	11	$\emptyset$
8	$\langle(3,2,2,1,2),1\vert$	$\langle 215,0\vert$	27	1 → Humedad 90–99,
				Tiempo encapotado o lluvia

En la lista, la inferencia consiste en asignarle la respuesta del ítem correspondiente. Si el ítem es desconocido, inferimos la respuesta a partir de los ítems adyacentes, es decir, las explicaciones "más probables". Para resolver esta situación también podemos recurrir al modelo subyacente que originó la tabla, por ejemplo un DI, o una opinión experta. Si hay atributos desconocidos o no instanciados en el caso objeto de la inferencia, se trata de una *pregunta abierta* a la lista *KBM2L* para la cual hemos propuesto un marco que resuelve de forma general una sesión interactiva de preguntas y respuestas entre la lista y un usuario. En la sección 4.3 desarrollamos este aspecto de las BC basadas en listas *KBM2L*. □

Dentro del aprendizaje automático podemos considerar otras técnicas de representación (inductivas o deductivas) no estrictamente proposicionales. En el ámbito de las representaciones probabilísticas enfocadas a la clasificación encontramos los clasificadores bayesianos (Mitchell, 1997). En el capítulo 4 mostraremos esta técnica aplicada a la gestión de consultas complejas a una BC basada en las listas *KBM2L*.

Cuadro 3.18: Árbol de clasificación y lista *KBM2L*. Comparación de la clasificación de casos (Humedad, Tiempo, Temperatura, Viento)

caso	árbol ID3	lista *KBM2L*	caso	árbol ID3	lista *KBM2L*
(0,0,0,0)	1	1	(2,0,0,0)	0	0
(0,0,0,1)	1	1	(2,0,0,1)	0	0
(0,0,1,0)	1	1	(2,0,1,0)	0	0
(0,0,1,1)	1	1	(2,0,1,1)	0	0
(0,0,2,0)	1	1	(2,0,2,0)	0	0
(0,0,2,1)	1	1	(2,0,2,1)	0	-1 → 0
(0,1,0,0)	1	1	(2,1,0,0)	**1**	-1 → **0**
(0,1,0,1)	1	1	(2,1,0,1)	**1**	-1 → **0**
(0,1,1,0)	1	1	(2,1,1,0)	1	-1 → 1
(0,1,1,1)	1	1	(2,1,1,1)	1	-1 → 1
(0,1,2,0)	1	1	(2,1,2,0)	1	-1 → 1
(0,1,2,1)	1	1	(2,1,2,1)	1	-1 → 1
(0,2,0,0)	1	1	(2,2,0,0)	1	1
(0,2,0,1)	**0**	**1**	(2,2,0,1)	**0**	**1**
(0,2,1,0)	1	1	(2,2,1,0)	1	1
(0,2,1,1)	**0**	**1**	(2,2,1,1)	0	0
(0,2,2,0)	1	1	(2,2,2,0)	**1**	0
(0,2,2,1)	**0**	**1**	(2,2,2,1)	0	0

Cuadro 3.19: Árbol de clasificación y lista *KBM2L*, comparación de la clasificación de casos (Humedad, Tiempo, Temperatura, Viento)

caso	árbol ID3	lista *KBM2L*	caso	árbol ID3	lista *KBM2L*
(1,0,0,0)	-1 → 1	1	(3,0,0,0)	0	0
(1,0,0,1)	-1 → 1	1	(3,0,0,1)	0	0
(1,0,1,0)	-1 → 1	1	(3,0,1,0)	0	0
(1,0,1,1)	-1 → 1	1	(3,0,1,1)	0	0
(1,0,2,0)	-1 → 1	1	(3,0,2,0)	0	0
(1,0,2,1)	-1 → 1	1	(3,0,2,1)	0	0
(1,1,0,0)	1	1	(3,1,0,0)	**1**	-1 → **0**
(1,1,0,1)	1	1	(3,1,0,1)	**1**	-1 → **0**
(1,1,1,0)	1	1	(3,1,1,0)	1	-1 → 1
(1,1,1,1)	1	1	(3,1,1,1)	1	1
(1,1,2,0)	1	1	(3,1,2,0)	1	1
(1,1,2,1)	1	-1 → 1	(3,1,2,1)	1	1
(1,2,0,0)	**1**	-1 → 0	(3,2,0,0)	1	1
(1,2,0,1)	0	0	(3,2,0,1)	**0**	1
(1,2,1,0)	**1**	**0**	(3,2,1,0)	1	1
(1,2,1,1)	0	0	(3,2,1,1)	**0**	1
(1,2,2,0)	**1**	**0**	(3,2,2,0)	1	1
(1,2,2,1)	0	0	(3,2,2,1)	**0**	1

Capítulo 4

KBM2L para Tablas de Decisiones Óptimas y sus Consultas

Presentamos en este capítulo 4 la aplicación de las técnicas descritas en los capítulos 2 y 3 a la construcción y explotación de una BC cuya fuente es el conjunto de TDO asociadas a un problema de decisión.

En la sección 4.1 introducimos un conjunto de técnicas desarrolladas para llevar a cabo la organización óptima de las TDO de un DI utilizando la lista *KBM2L*. Vemos en profundidad la construcción incremental de las listas a partir de las tablas, la extracción de información mediante consultas y la formulación de explicaciones. Esta sección pone de manifiesto la relación entre la representación de las TDO mediante listas *KBM2L* y la capacidad de explicación.

En la sección 4.3 proponemos un marco para resolver eficientemente preguntas a la BC. Describimos dos escenarios para la interacción del usuario con el SAD: el primero contempla la aproximación más simple que denominamos consulta cerrada; el segundo, que hemos llamado consulta abierta, plantea un modelo general de interrogación a la BC como un bucle de preguntas y respuestas que facilita la explotación de una BC real.

Por último, en las secciones 4.2 y 4.4 describimos dos aplicaciones reales de la técnica a las TDO de dos problemas de diagnóstico y tratamiento médico.

4.1. Síntesis de Tablas de Decisiones Óptimas Usando Listas *KBM2L*

Consideramos la construcción de una BC tipo *KBM2L* a partir de un conjunto de TDO. En primer lugar indicaremos las características más importantes de las TDO, que tienen repercusión

en su transformación a *KBM2L*.

La construcción de modelos analíticos de decisión y su implementación como SAD concierne procesos de evaluación, análisis, validación, operación y mantenimiento. Por consiguiente, los SAD demandados hoy día son sistemas basados en el conocimiento muy complejos. Algunos aspectos importantes que contribuyen a esta complejidad son:

1. La estructura complicada de la representación del problema con muchas interrelaciones entre elementos y muchas restricciones, es difícil incluso para las RB (Pearl, 1988) y los DI (Howard y Matheson, 2005).

2. El elevado grado de formalización matemática (Keeney, 1982): distribuciones de probabilidad para modelizar la incertidumbre, funciones de utilidad o valor para modelizar las preferencias del decisor (Ríos-Insua et al., 2002), lógicas formales (proposicional, de primer orden, restricciones (Sterling y Shapiro, 1994)),..., lo cual conduce a muchos parámetros (que a su vez tienen incertidumbre, bajo el punto de vista bayesiano).

3. La evaluación de un modelo de un problema de toma de decisiones tiene una complejidad no polinómica (Cooper, 1990), y su evaluación exacta es computacionalmente muy compleja.

4. Aunque los datos y las entradas se descomponen en partes pequeñas (Raiffa, 1968) (variables, alternativas de decisión, probabilidades conjuntas factorizadas mediante probabilidades condicionadas, funciones de utilidad aditivas y multiplicativas), el resultado descansa sobre un espacio de representación combinatoria. Específicamente, las TDO son exponenciales en términos del número de variables.

5. La modelización del tiempo es también difícil y su propagación a través de la herencia de influencias directas desde el pasado, limita el ámbito y la evaluación de la representación. El tamaño de las TDO en la secuencia temporal puede crecer varios órdenes de magnitud en cada instante de decisión.

Más aún, los SAD se consideran los sistemas basados en el conocimiento orientados al usuario por excelencia. Como tales, éstos deben tener una interfaz y comunicación cercana al usuario y a menudo tienen que dar respuestas en tiempo real.

La explicación de éstos debería dar una descripción de por qué la alternativa de decisión propuesta es óptima junto a nuevas intuiciones sobre la solución del problema. Pese a las dificultades ya indicadas en la sección 2.4.2, aquí mostraremos cómo pueden ser abordadas en sistemas útiles para problemas reales.

Los usuarios están involucrados activamente no solamente durante la construcción del sistema sino también durante la validación. Sin embargo, esto no es suficiente. El usuario puede aceptar/rechazar una propuesta del sistema pero la ayuda será mejor si recibe una buena, razonada, consistente y estructurada explicación. El SAD debe proporcionar explicaciones claras, concisas y completas que traduzcan el mecanismo de razonamiento al dominio del usuario y justifiquen la decisión propuesta (Druzdzel, 1996). Para ello, el SAD dispone del propio DI (la representación gráfica, las TPC y la función de utilidad) y de la evaluación del DI, es decir, las TDO. Para el usuario sólo es práctico ver el grafo, algunas TPC sencillas, algunos elementos de la función de utilidad y los esquemas de la tablas. El contenido de las TDO es demasiado grande y hay que extraer el conocimiento y presentarlo de modo eficiente, combinado con los elementos del DI que contribuyan a la comprensión y explicación de las propuestas del SAD.

Para representar un problema de decisión real se han desarrollado distintos tipos de herramientas, principalmente los árboles de decisión (pudiendo ser asimétricos), pero sobre todo los DI (sin explosión combinatoria). Estas representaciones de relaciones y modelos de conocimiento son equivalentes (Clemen, 1996) (Goodwin y Wright, 1998) y tras ser evaluados dan lugar a las TDO del MGP. Sin embargo, estas representaciones no resuelven las dificultades computacionales y de construcción de la BC; tan sólo las aplazan, tras la representación, al instante de la evaluación. La evaluación o la inferencia es tarea del ordenador y éste puede manejar grandes cantidades de información estructurada sin dificultad. Pero el problema surge de nuevo al pretender recoger, sintetizar, generalizar y explotar los resultados, es decir, las TDO, para construir una BC del SAD. En la TDO no se intuyen relaciones entre variables, que el DI representa explícitamente o implícitamente, y no se muestra una explicación de las políticas óptimas ni de la importancia relativa de las variables ni de sus valores en los resultados. Si las tablas son enormes no disponemos de las tablas completas, y la BC se construye con un conjunto significativo de casos (generalmente según criterio de los expertos), mediante instancias de atributos del esquema.

Las tablas que contienen las decisiones óptimas obtenidas cuando resolvemos problemas de toma de decisiones reales son casi siempre extremadamente grandes. Este hecho plantea problemas no sólo respecto al almacenamiento y gestión de grandes cantidades de información, sino también respecto al uso de las tablas para recuperación del conocimiento, extracción de reglas y explicación del razonamiento. Nuestra propuesta es sintetizar las TDO como MM de almacenamiento mínimo. Las listas *KBM2L* son una estructura nueva, que optimiza las TDO en el sentido de disponer las respuestas iguales de forma adyacente para conseguir un almacenamiento compacto. La lista de menor longitud incluye la misma información que las

TDO originales, pero es más compacta, lo cual es muy útil para explicar el razonamiento experto involucrado en la inferencia realizada mediante el modelo, en nuestro caso el DI.

La evaluación de un DI (Olmsted, 1983) deja, para cada instante de decisión, una TDO asociada con la información acerca de cuál es la mejor alternativa, es decir, la alternativa con máxima utilidad esperada, para cada combinación de variables (valores de variables aleatorias y/u otras decisiones anteriores). El algoritmo de evaluación determina qué variables del DI son las que constituyen el esquema de las distintas TDO. Consideramos una TDO como un conjunto de atributos (una MM) que determina la acción óptima, política o respuesta. Entonces, un aspecto esencial es minimizar el espacio de almacenamiento, que crece enormemente en problemas reales. Cada tabla puede tener miles de millones de filas y típicamente, más de veinte columnas, y el resultado del problema de decisión descansa, por tanto, sobre un espacio de representación combinatorio. En las TDO se supone que está toda la combinatoria o producto cartesiano de las variables del esquema y además no hay repeticiones.

Las TDO que consideramos pueden ser simples o agregadas. Las **simples** tienen una única alternativa de decisión como respuesta, son de respuesta cualitativa o cuantitativa (discretizada) unidimensional. Las TDO **agregadas** se construyen a partir de dos o más tablas simples cuyas respuestas tienen una dependencia temporal, que refleja el DI. La respuesta (agregada) de estas tablas es un vector cuyas componentes son las alternativas óptimas de decisión de todo el problema, es decir, una respuesta multidimensional.

La evaluación produce tantas TDO simples como decisiones hay en el DI, y de acuerdo al orden secuencial de las decisiones, se construyen las TDO agregadas que consideran ya las alternativas óptimas de las decisiones precedentes en las componentes vectoriales de la respuesta. En las secciones 4.2 y 4.4 desarrollaremos dos casos prácticos de construcción de la BC a partir de las TDO. En el contexto de la tesis, la BC del SAD es la lista *KBM2L* óptima que se obtiene a partir de la TDO agregada.

Ejemplo 4.1. Supongamos que tras la evaluación de un DI con dos decisiones (X e Y) hemos obtenido dos TDO. Ambas TDO simples tienen en sus esquemas dos atributos (A y B) de dominio binario y la segunda incluye además en su esquema la decisión X como atributo. La respuesta es igualmente binaria, considerando dos alternativas de decisión. Sean dichas tablas las siguientes:

A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

A	X	B	Y
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

A continuación propagamos la TDO de la primera decisión X sobre la TDO de la segunda decisión Y y obtenemos la TDO agregada siguiente:

A	B	X	Y
0	0	0	1
0	1	1	1
1	0	1	0
1	1	0	0

□

A los atributos de las TDO, los llamamos **variables** (aleatorias y de decisión). La respuesta de la lista en este contexto se llama **alternativa de decisión** óptima, en las TDO simples (escalar), y **política** o **estrategia** óptima, en las TDO agregadas o combinadas (vectorial). En el DI todas las variables las supondremos discretas y con dominios finitos y por tanto, en la lista asociada a sus TDO, también serán discretos tanto los atributos como la respuesta.

Dado el dominio discreto de las decisiones, las listas, en general, son una aplicación del conjunto de índices o desplazamientos en el conjunto de las alternativas. La información es dicha aplicación, que en el caso general, puede no ser conocida en todo el espacio de representación del esquema. En este último caso tenemos una respuesta desconocida *unKB* en ciertas entradas de la TDO. También puede ocurrir que no esté definida y entonces se asume que se trata de una restricción y tenemos una respuesta imposible *coKB* en ciertas entradas de la tabla.

Ilustraremos estas ideas mediante dos aplicaciones médicas. En primer lugar, en la sección 4.2, el SAD *PGNHL* (Lucas et al., 1998; Bielza et al., 2003; Bielza et al., 2006) para la gestión del linfoma gástrico primario no-Hodgkin que pone el contrapunto de un sistema de complejidad media cuya BC es muy compleja, es decir, la lista *KBM2L* óptima es muy larga. En segundo lugar, nuestro SAD *IctNeo* (Bielza et al., 2000; Fernández del Pozo et al., 2001; Gómez, 2002)

para la gestión de la ictericia neonatal, obteniendo excelentes resultados en un sistema muy complejo. Esta aplicación se expondrá al final del capítulo para ilustrar además el sistema de consultas de la sección 4.3.

4.2. Aplicación al Sistema PGNHL

4.2.1. Descripción del Problema y del Sistema de Ayuda a la Decisión

Con el fin de investigar las posibilidades de nuestros métodos dentro del dominio médico, hemos elegido un DI que muestra el tratamiento del linfoma primario no–Hodgkin de estómago, abreviado PGNHL, previamente desarrollado por (Lucas et al., 1998) en colaboración con expertos oncólogos clínicos. Llamaremos también *PGNHL* al modelo (DI) del SAD para este problema (Bielza et al., 2006).

Concretamente, (Lucas et al., 1998) recogen el modelo y describen el SAD en cuestión que nos sirve para experimentar las técnicas descritas previamente. Éste es un modelo clínico real, que refleja la evidencia científica actual acerca de esta enfermedad. Puede ser utilizado para determinar el tratamiento óptimo de pacientes con linfoma gástrico. Las opciones de tratamiento consisten en la prescripción de antibióticos o no, seguido de una cierta cirugía (ninguna, paliativa o curativa) y finalmente una combinación de quimioterapia y/o radioterapia. La parte probabilista del modelo, una RB, puede utilizarse independientemente para diagnóstico y para generar perfiles de riesgo relativos a pacientes específicos. El modelo supera a los modelos comunes de pronóstico basados en regresión logística, ya que es parte de un SAD que puede responder a muchas preguntas clínicas de diversa índole.

El linfoma gástrico primario no-Hodgkin es una enfermedad relativamente rara, que supone el 5 % del total de tumores de estómago. Esta enfermedad se asocia a una infección crónica por la bacteria *Helicobacter Pylori* (Eidt et al., 1994). No está aún completamente claro cuáles son los factores influyentes en el diagnóstico de los pacientes con esta enfermedad. La naturaleza rara de la enfermedad hace difícil recoger datos de un número suficiente de pacientes que puedan ser utilizados para determinar esos factores. Lo que está claro, sin embargo, es que se necesita algún tipo de ayuda a la decisión para los médicos en el tratamiento de estos pacientes. Los modelos que se han usado en el pasado para este propósito, son normalmente capaces de producir el diagnóstico de la enfermedad, pero no pueden usarse para seleccionar el tratamiento.

Para superar estas limitaciones, (Lucas et al., 1998) con ayuda de dos oncólogos construyeron diversos DI. Estos modelos están pensados para ser usados en pacientes con PGNHL diagnósticado histológicamente. Nosotros hemos tomado para este estudio la versión más compleja de

DI con tres nodos de decisión que será brevemente revisado a continuación. El primer nodo de decisión es el tratamiento para combatir la helicobacteria (TH), que corresponde a la decisión de prescripción o no de antibióticos contra la *Helicobacter Pylori*. La segunda decisión se refiere a llevar a cabo Cirugía (C), es decir, la total o parcial resección del estómago. Sus alternativas son tres: curativa, es decir, la eliminación completa de la masa tumoral localizada; paliativa, es decir, eliminación incompleta; o no realizar intervención quirúrgica. La última decisión, es la ejecución de un plan de quimioterapia y/o radioterapia (PQR) y se refiere a la selección de quimioterapia (Q), radioterapia (R), quimioterapia seguida de radioterapia (Q.más.R), o no intervención.

El modelo de DI se muestra en la Figura 4.1. Consiste en 17 nodos de azar (elipses), un nodo de valor (rombo) y 3 nodos de decisión (rectángulos), 42 arcos, 8282 parámetros de probabilidad donde la TPC mayor tiene 3840 entradas, y una tabla inicial para el nodo de valor de tamaño 144. Los nodos a la izquierda de los nodos de decisión, véase la Figura 4.1, representan la información del pretratamiento; aquellos que están a la derecha representan el postratamiento. Las variables requeridas para la compresión de la sección siguiente se muestran en el Cuadro 4.1.

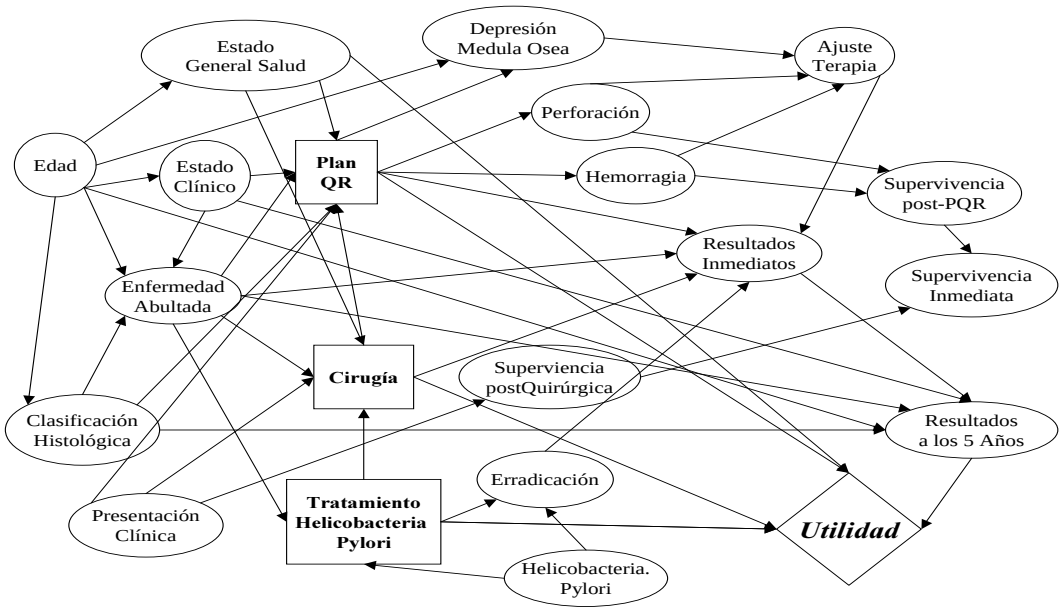

Figura 4.1: Diagrama de influencia del PGNHL

La estructura del diagrama se determinó a partir de las relaciones causales conocidas y las (in)dependencias probabilísticas encontradas en la literatura. Las probabilidades fueron educidas con ayuda de relaciones probabilistas cualitativas y lógicas contra las cuales fueron comprobadas las asignaciones numéricas. Para las dos TPC más largas asociadas a los resultados inmediatos (RI) y resultados a los cinco años (R5), se usó un modelo aditivo especial para hacer más fácil

Cuadro 4.1: Dominio de variables

Variables	**Dominios**
TRATAMIENTO–HELICOBACTERIA (TH)	No, Sí
CIRUGÍA (C)	Nada, Curativa, Paliativa
PLANIFICACIÓN–QT-RT (PQR)	Nada, R, Q, Q.más.R
ESTADO-SALUD-GENERAL (ESG)	Pobre, Medio, Bueno
FASE–CLÍNICA (FC)	I, II1, II2, III, IV
ENFERMEDAD-ABULTADA (EA)	Sí, No
CLASIFICACIÓN–HISTOLÓGICA (CH)	Grado.Bajo, Grado.Alto
HELICOBACTER–PYLORI (HP)	Ausente, Presente
PRESENTACIÓN–CLÍNICA (PC)	Nada, Hemorragia, Perforación, Obstrucción

la asignación.

Mediante una RB obtenida a partir del DI, convirtiendo cada nodo de decisión en un nodo de azar (Cooper, 1988), se comprobó la precisión de las asignaciones comparando las distribuciones de probabilidad marginales a priori y a posteriori de la red con los datos de frecuencias suministrados por la literatura y por la experiencia clínica. Además, se utilizó una base de datos de 137 pacientes para evaluar la precisión del modelo. También se aprendió una nueva distribución de probabilidad conjunta a partir de la base de datos y sus probabilidades marginales a priori se compararon con las de la red original.

La educción de la utilidad, desde la perspectiva del paciente, se realizó utilizando dos métodos: asignación directa y loterías de referencia (Farquhar, 1984). Se obtuvieron varias funciones de utilidad que fueron refinadas examinando el funcionamiento del sistema respecto de los tratamientos óptimos propuestos para algunos pacientes típicos. Finalmente se obtuvo una función de utilidad clínicamente fiable.

Si bien el desarrollo de modelos de decisión como éste es complejo, y necesita mucho tiempo, y nos enfrenta a dificultades computacionales en la fase de evaluación (Bielza et al., 2000), dichos modelos han demostrado ser muy útiles. Pueden utilizarse para dar información del diagnóstico acerca de pacientes específicos, dadas ciertas características particulares del paciente y, quizás, de las decisiones terapéuticas. Pueden ser aplicados en la determinación de tratamientos óptimos. Más aún, razonando en el sentido opuesto, asumiendo que los resultados finales del tratamiento son conocidos, los modelos pueden usarse para generar perfiles probabilísticos que se ajusten a estos resultados finales. Obviamente, todas estas capacidades son útiles en la guía del proceso de toma de decisiones clínico. Sin embargo, comprender las recomendaciones de tratamiento

generadas por el SAD para la población de pacientes *completa* no es nada sencillo. A los médicos les sería beneficioso tener explicaciones claras y concisas de los resultados del sistema, que podrían justificar esos resultados y ayudarles a comprenderlos (o rechazarlos). Además, esto produciría una forma alternativa de validación del sistema.

En (Lucas et al., 1998) ya se realizó una evaluación preliminar de la precisión del modelo PGNHL por medio de un estudio clínico doble ciego. La actual investigación, donde usamos listas *KBM2L* para comprender mejor las bases del tratamiento del modelo PGNHL, complementa este estudio previo.

4.2.2. Resultados

La evaluación del DI da lugar a tres TDO, cada una conteniendo el tratamiento óptimo para cada combinación de variables de las tablas. La primera tabla se refiere a la decisión TH, que depende de cuatro atributos: FC, EA, CH y HP. Usando las variables en este orden, es decir, usando la *Base* [FC, EA, CH, HP], construimos una lista *KBM2L*. Tiene 17 ítems que cubren el conjunto completo de 40 casos; 32 casos tienen TH = No, organizados en 9 ítems; 8 casos tienen TH = Sí, agrupados en 8 ítems.

La segunda tabla para la decisión C tiene 7 atributos, los de la decisión previa TH y dos nuevos (ESG y PC). La *Base* [ESG, TH, FC, EA, CH, HP, PC] configura la lista *KBM2L* con 385 ítems, que cubren un conjunto total de 960 casos. La distribución de estos ítems es: 193 para C = Nada (663 casos) y 192 para C = Curativa (297 casos).

La última tabla, para la decisión PQR, tiene 8 atributos. Construimos la lista *KBM2L* respecto de la *Base* [ESG, C, TH, FC, EA, CH, HP, PC] y contiene 678 ítems cubriendo un conjunto total de 2880 casos. Encontramos 164 ítems de PQR = Nada (490 casos), 188 de PQR = R (862 casos), 280 de PQR = Q (1404 casos) y 46 de PQR = Q.más.R (124 casos).

Esta información puede ser visualizada adecuadamente con ayuda del gráfico del espectro, véase la Figura 4.2.

A continuación aplicamos el algoritmo de entorno variable a cada lista *KBM2L* para mejorar las *Bases* actuales. Las listas mejoradas de cada decisión se muestran en la Figura 4.3. Usamos un ordenador personal PentiumIII^{TM} a 1-GHz con 512 MB y Windows XP^{TM} y Java^{TM}2 como entorno de programación. Para TH, C y PQR tenemos unos tiempos de proceso para la optimización AEV de 2.5, 164 y 2238 segundos, respectivamente, y se han requerido 12, 78 y 168 *XBase*.

El siguiente paso es agregar las tres TDO simples asociadas con las listas previas para producir una única lista global con el conocimiento completo del protocolo clínico. Primeramente

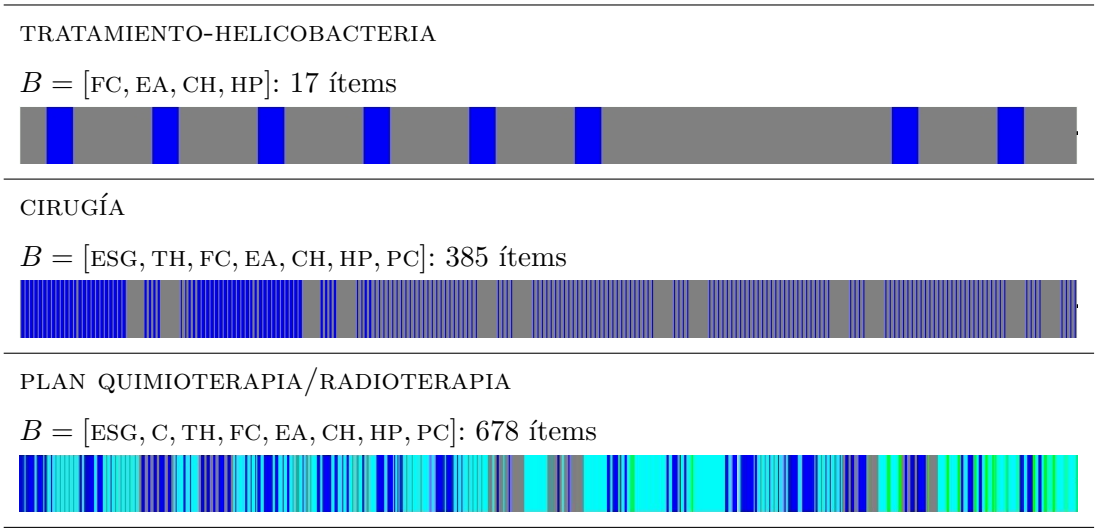

Figura 4.2: Espectro *KBM2L* de las TDO simples no optimizadas

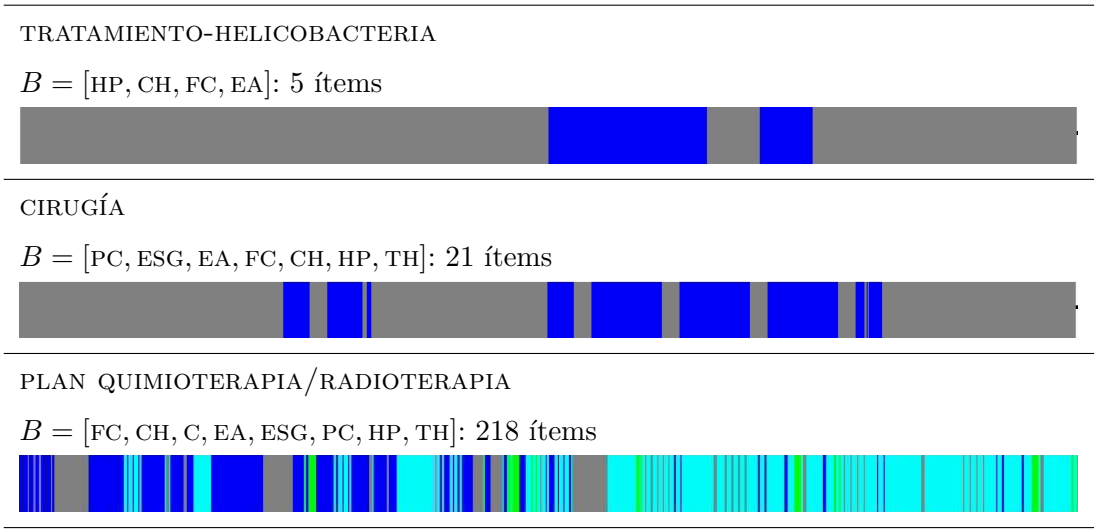

Figura 4.3: Espectro *KBM2L* de las TDO simples optimizadas

las decisiones óptimas relativas a TH son propagadas sobre la TDO de la decisión C. Eliminamos las filas de la TDO de C que no contemplan decisiones óptimas en la TDO de TH y añadimos el valor óptimo de TH a la respuesta de C. Propagamos la lista (TH,C) sobre la TDO de PQR análogamente. La lista final tiene 6 atributos pues TH, C y PQR son ahora componentes de la respuesta vectorial de la tabla agregada. La terna (TH, C, PQR) es la alternativa óptima combinada para cada caso. Dependiendo de la fase del tratamiento y de la información disponible, el SAD accederá a la lista (TH), (TH, C) o (TH, C, PQR).

Como resultado, la *Base* es B_0 = [EA, HP, ESG, FC, PC, CH] y su lista *KBM2L* tiene

340 ítems. Nótese que en esta lista agregada, tenemos más decisiones posibles que en una lista simple relativa a un tratamiento escalar, es decir, potencialmente hasta $2 \cdot 3 \cdot 4 = 24$, aunque tan sólo se han obtenido 13 al evaluar el modelo. La cirugía paliativa no se aplica y algunas otras combinaciones tampoco. La información más relevante de esta lista se muestra en el Cuadro 4.2.

Cuadro 4.2: Información de la tabla agregada

Alternativas Óptimas (TH,C,PQR)	#Casos	#Ítems en B_0	#Ítems en B_{final}
(No,Nada,Nada)	21	12	7
(No,Nada,R)	68	44	31
(No,Nada,Q)	179	88	40
(No,Curativa,Nada)	48	41	26
(No,Curativa,R)	45	44	34
(No,Curativa,Q)	19	11	13
(No,Curativa,Q.más.R)	4	4	4
(Sí,Nada,Nada)	3	3	1
(Sí,Nada,R)	34	34	13
(Sí,Nada,Q)	27	27	9
(Sí,Curativa,Nada)	22	22	10
(Sí,Curativa,R)	7	7	6
(Sí,Curativa,Q)	3	3	1

Teniendo en cuenta la cardinalidad de cada dominio de los atributos, el número de posibles combinaciones (casos) es 480. Por consiguiente, el conocimiento representado por los 340 ítems parece que está considerablemente fragmentado y en consecuencia hay razones para intentar mejorar. Nuevamente mediante AEV, después de 107 $XBase$, tras 273 segundos de proceso, obtenemos una lista más corta y un conocimiento refinado acerca de los atributos decisivos. La lista tiene 195 ítems, que representa una reducción del 42.6 %. La $Base$ óptima es $B_{final} =$ [HP,CH,PC,FC,EA,ESG]. El Cuadro 4.2 muestra la distribución de casos e ítems. La Figura 4.4 muestra los dos espectros asociados con B_0 y B_{final}.

El Cuadro 4.3 muestra parte de la lista final con algunos ítems significativos. La parte fija de cada ítem, es decir, su explicación, aparece en negrita. Esta lista puede leerse como la representación de 195 reglas (ítems) que indican la política global óptima (consecuente) como función de los atributos clave (antecedente). Los atributos están ordenados de mayor a menor

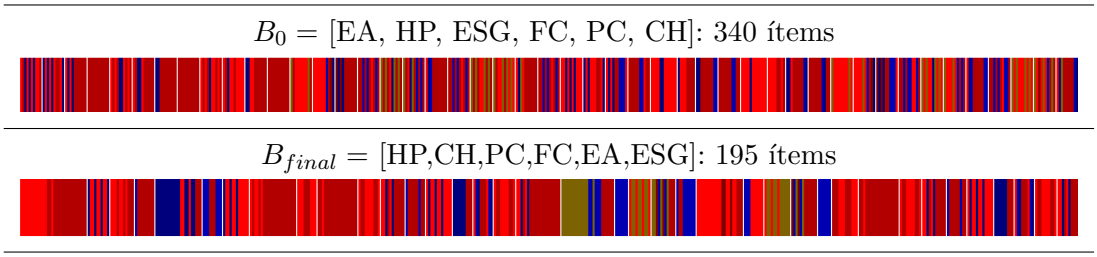

Figura 4.4: Espectros no optimizados y optimizados para la tabla agregada

importancia de izquierda a derecha en el antecedente conforme a la ordenación de la *Base* y al correspondiente peso, véase la ecuación (2.1).

Consideremos la regla 80, (TH=No, C=Curativa, PQR=Q), comparada con la regla 81, (TH=No, C=Curativa, PQR=Q.más.R). Ambas reglas incluyen un solo caso pues sus partes variables son vacías. El estado de salud general del paciente ESG toma los valores "Medio" y "Bueno" respectivamente, lo cual explica la diferencia entre las dos reglas. La razón es que si el estado de salud del paciente es bueno, puede seleccionarse un tratamiento más agresivo y eficaz.

La diferencia entre la regla 125, (TH=No, C=Curativa, PQR=Q), y la regla 127, (TH=Sí, C=Curativa, PQR=Q), puede explicarse apuntando a la fase clínica (FC), que es distinta en ambos ítems. Claramente el sistema ha decidido que el tratamiento para la fase más avanzada de la enfermedad (FC = IV), en el contexto común de una versión de linfoma de grado bajo que progresa lentamente (CH=Grado.Bajo), debe ser más agresivo que para la fase menos avanzada (FC = III), donde ambas fases de la enfermedad son esencialmente incurables. Ésta es obviamente una elección que podría estar abierta a debate entre los oncólogos.

Los médicos no están únicamente interesados en comparar reglas sino también en encontrar el contexto en el que se aplica un tratamiento. Consideremos, por ejemplo, el tratamiento $T \equiv$ (TH=No, C=Curativa, PQR=Q.más.R), el cual se manifiesta en 4 ítems como muestra el Cuadro 4.2, incluyendo el ya mencionado ítem 81. Centrándonos en este tratamiento, buscamos una nueva *Base* distinguiendo ahora sólo un par de posibles tratamientos T y $\neg T$, véase la sección 2.4.2. La nueva *Base* obtenida es [ESG, CH, PC, EA, FC, HP], con sólo tres ítems (de tamaños 456, 4 y 20 casos). Observamos que el atributo ESG ha ganado importancia en la explicación clínica de la respuesta específica que supone este estudio local. Los casos asociados con T están ahora agrupados en un único ítem (el segundo) y la explicación resultante es ESG = Bueno, CH = Grado.Alto, PC = Perforación y EA = No.

Una posible interpretación del resultado es la siguiente. Como el paciente presenta una perforación en el estómago es necesario realizar una cirugía curativa. Esto se combina con quimioterapia y radioterapia que es efectiva en las fases tempranas del linfoma de bajo grado y en

Cuadro 4.3: La lista *KBM2L* óptima

#	⟨ Descripción del ítem, (Política óptima) |
0	⟨***HP*: Ausente, *CH*: Grado.Bajo, *PC*: Nada**, *FC*: II1, *EA*: No, *ESG*: Bueno, (*TH*: No, *C*: Nada, *PQR*: R)|
1	⟨***HP*: Ausente, *CH*: Grado.Bajo, *PC*: Nada, *FC*: II2, *EA*: Cí**, *ESG*: Medio, (*TH*: No, *C*: Nada, *PQR*: Q)|
2	⟨***HP*: Ausente, *CH*: Grado.Bajo, *PC*: Nada, *FC*: II2, *EA*: Cí, *ESG*: Bueno**, (*TH*: No, *C*: Nada, *PQR*: R)|
3	⟨***HP*: Ausente, *CH*: Grado.Bajo, *PC*: Nada**, *FC*: IV, *EA*: No, *ESG*: Bueno, (*TH*: No, *C*: Nada, *PQR*: Q)|
...	...
80	⟨***HP*: Ausente, *CH*: Grado.Alto, *PC*: Perforación, *FC*: IV, *EA*: No, *ESG*: Medio**, (*TH*: No, *C*: Curativa, *PQR*: Q)|
81	⟨***HP*: Ausente, *CH*: Grado.Alto, *PC*: Perforación, *FC*: IV, *EA*: No, *ESG*: Bueno**, (*TH*: No, *C*: Curativa, *PQR*: Q.más.R)|
...	...
118	⟨***HP*: Presente, *CH*: Grado.Bajo, *PC*: Hemorragia, *FC*:IV**, *EA*: No, *ESG*: Bueno, (*TH*: Sí, *C*: Nada, *PQR*: Q)|
119	⟨***HP*: Presente, *CH*: Grado.Bajo, *PC*: Perforación**, *FC*:II1, *EA*: No, *ESG*: Medio, (*TH*: Sí, *C*: Curativa, *PQR*: Nada)|
120	⟨***HP*: Presente, *CH*: Grado.Bajo, *PC*: Perforación**, *FC*:II2, *EA*: Sí, *ESG*: Pobre, (*TH*: Sí, *C*: Curativa, *PQR*: R)|
121	⟨***HP*: Presente, *CH*: Grado.Bajo, *PC*: Perforación, *FC*:II2, *EA*: Sí**, *ESG*: Bueno, (*TH*: Sí, *C*: Curativa, *PQR*: Nada)|
122	⟨***HP*: Presente, *CH*: Grado.Bajo, *PC*: Perforación, *FC*:II2, *EA*: No, *ESG*: Pobre**, (*TH*: Sí, *C*: Curativa, *PQR*: R)|
123	⟨***HP*: Presente, *CH*: Grado.Bajo, *PC*: Perforación, *FC*:II2, *EA*: No**, *ESG*: Bueno, (*TH*: Sí, *C*: Curativa, *PQR*: Nada)|
124	⟨***HP*: Presente, *CH*: Grado.Bajo, *PC*: Perforación, *FC*:III, *EA*: Sí**, *ESG*: Bueno, (*TH*: No, *C*: Nada, *PQR*: R)|
125	⟨***HP*: Presente, *CH*: Grado.Bajo, *PC*: Perforación, *FC*:III, *EA*: No**, *ESG*: Bueno, (*TH*: No, *C*: Curativa, *PQR*: Q)|
126	⟨***HP*: Presente, *CH*: Grado.Bajo, *PC*: Perforación, *FC*:IV, *EA*: Sí**, *ESG*: Bueno, (*TH*: Sí, *C*: Nada, *PQR*: Nada)|
127	⟨***HP*: Presente, *CH*: Grado.Bajo, *PC*: Perforación, *FC*: IV, *EA*: No**, *ESG*: Bueno, (*TH*: Sí, *C*: Curativa, *PQR*: Q)|
128	⟨***HP*: Presente, *CH*: Grado.Bajo, *PC*: Obstrucción, *FC*: I, *EA*: Sí, *ESG*: Pobre**, (*TH*: Sí, *C*: Curativa, *PQR*: Nada)|
...	...
193	⟨***HP*: Presente, *CH*: Grado.Alto, *PC*: Obstrucción, *FC*: II2, *EA*: No, *ESG*: Pobre**, (*TH*: No, *C*: Curativa, *PQR*: Nada)|
194	⟨***HP*: Presente, *CH*: Grado.Alto, *PC*: Obstrucción**, *FC*: IV, *EA*: No, *ESG*: Bueno, (*TH*: No, *C*: Nada, *PQR*: Q)|

las fases avanzadas del linfoma de alto grado. Los antibióticos no se administran dado que no hay infección.

Los patrones de regularidad que podemos distinguir teniendo en cuenta los tratamientos óptimos en las TDO, no sólo dependen del problema particular, sino de la organización interna de las TDO. La lista *KBM2L* es una herramienta para mostrar diferentes organizaciones de las tablas. Una buena organización de las tablas reduce los requerimientos de memoria necesarios para almacenarlas, pero, más importante aún en el dominio del problema (médico,...), muestra cuáles son los atributos clave que pueden explicar las propuestas del SAD. Éste es ciertamente el caso en las aplicaciones del modelo PGNHL.

Durante el refinamiento de un DI, los expertos (médicos,...) involucrados en el proceso de construcción pueden estudiar si las explicaciones generadas para los tratamientos óptimos sugeridos están de acuerdo con su opinión. Con base en estas intuiciones pueden abordarse mejoras en ciertas partes del modelo (influencias, probabilidades, utilidades, nuevas variables y/o simplificaciones en el grafo).

4.3. Consultas a una Lista *KBM2L*

En esta sección nuestro método de minimización del espacio de almacenamiento de grandes TDO es combinado con una propuesta de sistema de consultas para que el SAD responda a preguntas de expertos acerca de la acción preferida, para una instanciación dada de los atributos de las TDO (Fernández del Pozo y Bielza, 2002a). La principal dificultad será precisar la respuesta asociada a preguntas con instancias incompletas. Más aún, las TDO frecuentemente sólo incluyen un subconjunto del problema completo debido a problemas computacionales, lo que conduce a respuestas desconocidas en las TDO.

Durante el funcionamiento normal del SAD, el experto le preguntará acerca de cuál es la mejor recomendación dado un conjunto de valores del esquema (o parte). El decisor experto usa las TDO como BC. En una sesión típica con el experto, el SAD lleva a cabo conjuntamente con él un diálogo que describimos mediante las siguientes tareas (Figura 4.5):

(A) formular una consulta en el dominio de la BC,

(B) traducirla al formalismo de la BC,

(C) implementar la recuperación o acceso a la respuesta,

(D) construir eficientemente la respuesta,

(E) comunicar la(s) respuesta(s) y/o sugerir mejoras en la pregunta formulada,

y el SAD espera información del usuario para continuar con alguna de las tareas o terminar la sesión.

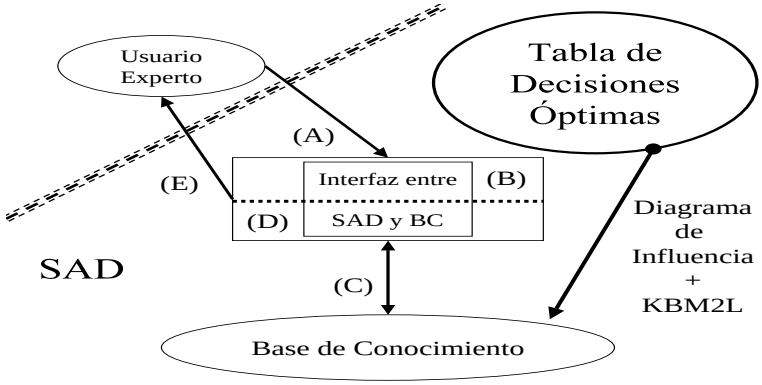

Figura 4.5: Sesión típica entre experto y SAD

Por lo que respecta a (A) y (B), hay un amplio abanico de posibles formulaciones de consultas. Nosotros proponemos una simple clasificación en dos grupos dependiendo de si todo el conjunto de atributos es instanciado –consulta cerrada– o no –consulta abierta–. Una consulta cerrada corresponde a un usuario que plantea una pregunta específica y definida, pues conoce toda la información relativa a los atributos. Una pregunta abierta es más general, puesto que incluye valores indefinidos de los atributos, porque pudieran ser difíciles o caros de obtener o porque su valor no fuese fiable. Esta clasificación es similar a la propuesta en (Martínez et al., 2001), aunque los autores tratan con *Sistemas de Información Geográfica* y están más preocupados por la actualización y acceso eficiente de los datos desde el punto de vista físico (meramente como una base de datos), que desde un punto de vista lógico (como una BC).

Las consultas se consideran instanciaciones de atributos, y por consiguiente, están relacionadas con el índice. El usuario espera una respuesta en términos de la política óptima. Pero debemos volver a recordar que normalmente la BC está incompleta y en ciertos contextos (muy amplios) se ignora cuál es la política óptima. La calidad de la BC depende, entre otras características, de la probabilidad de que el usuario necesite ayuda en dichos contextos.

Profundicemos en este punto con mayor detalle. La evaluación completa del problema de decisión puede ser muy costosa (en términos del tiempo y de la memoria requeridos) de modo que debamos recurrir a resolver un conjunto de subproblemas como alternativa, cada uno como resultado de la instanciación de algunas variables (Bielza et al., 2000). Ese conjunto de subproblemas puede no ser exhaustivo, dando lugar a políticas óptimas desconocidas para algunas combinaciones de atributos, es decir, aquellas asociadas con los subproblemas no resueltos. Esto

es habitual en grandes problemas de toma de decisiones. De hecho, el propio proceso de construcción de la *KBM2L* óptima también trata con subproblemas y con políticas desconocidas, véase la sección 4.4, donde mostramos un problema cuya complejidad requiere la división en subproblemas y la selección de un subconjunto cuya evaluación es factible. En resumen, no solamente distiguimos entre consultas cerradas y abiertas, sino también entre respuestas conocidas y desconocidas. Los diferentes escenarios se analizan en las siguientes secciones, presentando así una implementación posible de las tareas (C)-(E) del diálogo experto/SAD.

4.3.1. Consultas Cerradas

En una consulta cerrada todo el conjunto de atributos está instanciado. Entonces el sistema sólo necesita buscar el correspondiente elemento de la lista *KBM2L* y recuperar la política óptima, que se presenta al usuario como respuesta, junto a la explicación correspondiente. El mecanismo es el siguiente.

Sean $A_0, \ldots, A_{\delta-1}$ los atributos y $\Omega_0, \ldots, \Omega_{\delta-1}$ sus respectivos dominios. Ω_R denota el dominio de la política con las posibles alternativas. $\mathbf{Q}$ denota una consulta cerrada, es decir $\mathbf{Q} = (a_0, \ldots, a_{\delta-1}), a_i \in \Omega_i, i = 0, \ldots, \delta - 1$. Supongamos que hemos obtenido la lista *KBM2L* óptima con respecto a la *Base* $B = [A_0, \ldots, A_{\delta-1}]$ y $\overrightarrow{w}$ es su respectivo vector de pesos. Si esta lista tiene h ítems, entonces la lista es $\langle q_0, r_0 | \langle q_1, r_1 | \cdots \langle q_{h-1}, r_{h-1} |$, donde, $0 \leq q_i \leq W_0 \cdot \delta_0 - 1, q_i < q_{i+1}\, \forall i$ (desplazamientos), $r_i \in \Omega_R, r_i \neq r_{i+1}\, \forall i$ (políticas).

El procedimiento de recuperación de la respuesta (tarea (C) anterior) consiste en proyectar $\mathbf{Q}$ en el espacio del desplazamiento $\{0, 1, \ldots, W_0 \cdot \delta_0 - 1\}$ y derivar su política óptima de la lista *KBM2L*. Es decir, si $\langle \cdot, \cdot \rangle$ denota el producto escalar de vectores, calculamos $\langle \mathbf{Q}, \overrightarrow{w} \rangle = f(\mathbf{Q}) = q$ (véase la ecuación 2.1), y siempre que $q \in (q_{i-1}, q_i)$ entonces la respuesta es r_i.

El experto no sólo está interesado en conocer cuál es la mejor recomendación para un cierto caso, sino también en una explicación de por qué es la óptima. Así, finalmente, la respuesta r_i mostrada por el sistema al experto debe ser completada a través de la solicitud de una explicación.

Para respuesta *unKB* no hay explicación, aunque formalmente los ítems desconocidos tengan parte fija. Por ello, si r_i es desconocida, entonces una solución eficiente es invocar la evaluación del DI y resolver el correspondiente subproblema que hace a r_i conocida y actualizar la BC.

4.3.2. Consultas Abiertas

Nuestra propuesta establecerá un diálogo automático e interactivo entre el SAD y el experto para extraer información del experto y de la BC y reducir la incertidumbre generada por

la consulta abierta. Típicamente, el proceso involucrará el aprendizaje de una RB (Buntine, 1996) a partir de la parte relevante de la TDO respecto de la consulta y el cálculo de algunas probabilidades condicionadas interesantes que serán revisadas apropiadamente por el experto.

Observamos al experto desde el punto de vista de un agente interesado en conocer la política óptima para un problema de toma de decisiones y que formula consultas al sistema. El experto y el SAD mantienen un diálogo que consiste en consultas, respuestas y explicaciones. Para las preguntas cerradas el experto recibe una respuesta definida y precisa. Para las consultas abiertas esta tarea es más complicada debido a la imprecisión del experto. No todos los atributos están instanciados. Las razones posibles pueden ser la falta de fiabilidad de los valores disponibles de algunos atributos, conocimiento no disponible, el elevado coste de la extracción/obtención de ciertos datos o simplemente un interés en hacer consultas complejas relativas a dominios completos de sólo algunos atributos. Por ejemplo, en un escenario clínico, los médicos tienen acceso a datos administrativos como sexo, edad, ..., pero pueden no tener acceso a atributos como el tratamiento inicial recibido o algunos resultados de pruebas. También, pueden estar interesados en preguntar todo el rango de pesos de todos los posibles pacientes. Así, una consulta abierta es, por ejemplo $\mathbf{OQ} = (*, a_1, *, ..., a_{\delta-1})$, donde $*$ denota valores de atributos no instanciados.

En principio, el SAD busca los correspondientes elementos de la lista *KBM2L* y recupera la política óptima (o políticas) para presentársela(s) al usuario como respuesta. Supongamos, como antes, que la lista óptima se obtiene respecto a la *Base* $B = [A_0, ...A_{\delta-1}]$ y que la consulta es abierta con respecto a los atributos i y j, es decir, la consulta es $\mathbf{OQ} = (a_0, a_1, ..., *_i, ..., *_j, ..., a_{\delta-1})$. En realidad, $\mathbf{OQ}$ es un conjunto de consultas cerradas $\mathbf{Q}_i$, a saber,

$$\mathbf{OQ} = \{\mathbf{Q}_i = (a_0, a_1, ..., x, ..., y, ..., a_{\delta-1}) : x \in \Omega_i, y \in \Omega_j\} = \bigcup_{i=1}^{\delta_i \times \delta_j} \mathbf{Q}_i,$$

donde $\delta_i \times \delta_j$ es el producto de los tamaños de los dominios de los atributos i y j.

Entonces, el procedimiento de recuperación de la respuesta consistirá en aplicar la técnica descrita en la sección 4.3.1 a cada $\mathbf{Q}_i$, calculando $\langle \mathbf{Q}_i, \overrightarrow{w} \rangle = f(\mathbf{Q}_i) = p_i$, para dar el conjunto de desplazamientos $P = \{p_1, p_2, ..., p_{\delta_i \times \delta_j}\}$, con sus respectivas respuestas $S = \{s_1, s_2, ..., s_{\delta_i \times \delta_j}\}$.

Distinguimos cuatro posibles situaciones:

(i) todas las s_i son iguales y conocidas; esta situación implica una respuesta precisa (s_i) y el SAD no requiere interacción adicional con el experto;

(ii) todas las s_i son iguales pero desconocidas;

(iii) hay al menos dos valores diferentes entre las s_i's y todas son conocidas;

(iv) hay al menos dos valores diferentes entre las s_i's y además algunas son desconocidas.

Excepto la *Situación (i)*, el resto involucra varias posibles respuestas y/o respuestas inciertas requiriéndose un refinamiento de la consulta que ayude al experto. Y es aquí donde las tareas (D) y (E) señaladas anteriormente juegan un importante papel.

La información para el experto consta de dos conjuntos P y S, que representan registros simples de la BC. Se obtienen accediendo a la lista en su $Base$ óptima. Nótese, que esta base es la mejor para la BC *completa*, tanto para minimizar los requerimientos de almacenamiento como para maximizar la eficiencia de la explicación del conocimiento. No obstante, esta $Base$ puede no ser útil con respecto a la porción de la BC *relativa a la consulta abierta*.

El peso de los atributos cambia dependiendo de su posición en la $Base$, y la posición más a la derecha es la que consideramos de menor peso. Proponemos $XBase$ de modo que los atributos abiertos de la consulta se muevan a las posiciones más a la derecha. La consulta es la misma, pero su orden de los atributos implica trabajar en una nueva $Base$ con los atributos abiertos situados en las posiciones de menor peso. Desde el punto de vista semántico, este movimiento está también de acuerdo con la idea de relegar los atributos abiertos a las posiciones de menor importancia tal y como parece indicar la consulta, esto es, el experto no les ha asignado valor y no muestra un interés significativo por estos atributos. La nueva $Base$ se llamará $Base$ **operativa**, pues nos facilita una configuración para operar con la lista en el diálogo experto/SAD. Dada una consulta abierta, en general, hay varias $Bases$ operativas, todas igualmente válidas. Podemos elegir la $Base$ que conduzca al menor esfuerzo computacional asociado al $XBase$ y transposición del conocimiento, ensayando con unas pocas en tiempo lineal.

Un $XBase$ puede ser interpretado como un cambio de punto de vista en la consulta y respuesta. Para el SAD, la $Base$ óptima representa una buena organización del contenido completo de la BC. Para el experto, la $Base$ operativa le ofrece una vista organizada de las respuestas a su consulta abierta, apareciendo los registros consecutivos. Así, esta $Base$ será óptima para la explicación de las respuestas. Esto ilustra cómo una consulta puede ser complicada en una $Base$ y más fácil en otra. Puede tener cierta analogía con el dominio humano del conocimiento, donde una pregunta simple para un experto puede ser complicada para un no experto. Un experto tiene una buena descripción del problema: un buen modelo conceptual, unos pocos hechos relevantes, y una buena organización de la información que le permite analizar los hechos y explicar las respuestas que proponga a posibles preguntas. De hecho, por esto es considerado experto.

Ahora, trabajando en la $Base$ operativa, en el nuevo conjunto de desplazamientos P' todos los p'_i's son consecutivos y podemos introducir algunas reglas basadas en distancias en el espacio del desplazamiento para hacer más precisas las recomendaciones al experto.

La *Situación (ii)* puede sorprender ya que preguntamos al SAD por algo que desconoce (del problema en el que nos facilita ayuda), pero es a causa de que ciertos subproblemas asociados no han sido evaluados. De todas formas intentamos ofrecer una solución. Específicamente, la situación (ii) puede resolverse prediciendo qué respuesta está asociada con el desplazamiento más cercano a P' basado en la distancia euclídea en el espacio del desplazamiento, es decir en $\{0, 1, ..., W_0 \cdot \delta_0 - 1\}$, como sigue. Sean p^l, p^u el mínimo y máximo desplazamiento, respectivamente, incluidos en P'. Supongamos que q^l es el máximo desplazamiento con política conocida (digamos r) que precede a p^l en la lista *KBM2L* operativa. Análogamente, q^u es el desplazamiento menor con respuesta conocida (digamos s) que sigue a p^u en la lista *KBM2L* operativa. Todos los registros que son iguales a (p', s), con $p' \in P'$, $s \in S$, pertenecen al mismo ítem en la lista *KBM2L* operativa, mientras que los desplazamientos q^l y q^u están situados en sus ítems adyacentes. Entonces, calculamos $d_1 = |p^l - q^l|$ y $d_2 = |p^u - q^u|$. Si $d_1 < d_2$, entonces la respuesta es r; de otro modo la respuesta es s.

La *Situación (iii)* presenta diferentes valores de la política en S. Podemos dar una respuesta inmediata al experto basada en estadísticas sobre la distribución del valor de la política (mediana, moda, ...). Sin embargo, preferimos una respuesta más inteligente. Como primera propuesta, que llamaremos **algoritmo** $\mathcal{A}1$, el SAD solicita al experto que instancie el atributo abierto más a la izquierda con respecto de la *Base* óptima. Ésta será la instanciación de atributos más eficiente para reducir la incertidumbre de la respuesta multivaluada. Dicho atributo es el que tiene el peso más grande de entre los atributos abiertos, lo cual implica que tiene mayor probabilidad de estar en la parte fija de los índices de los ítems, que es la que explica la política fijada en cada ítem. Así, los atributos más a la izquierda, cuando son instanciados hacen menos probable que la consulta abierta conduzca a múltiples modalidades de la respuesta. Si es necesario, entonces, se procede a la instanciación del segundo atributo abierto más a la izquierda, y así sucesivamente. Esta aproximación se ajusta bien a problemas con muchos atributos pero con pocos atributos abiertos.

Para muchos atributos abiertos, digamos más de 10, tenemos suficiente información para realizar inferencias automáticas vía un proceso de aprendizaje. Así, proponemos centrarse otra vez en los registros de interés de la lista *KBM2L* operativa y aprender de ellos las relaciones probabilísticas entre los atributos y la política. La estructura a aprender será una RB (véase, por ejemplo (Pearl, 1988)), pues tiene una semántica clara para realizar muchas tareas de inferencia necesarias para sistemas inteligentes (diagnóstico, explicación, aprendizaje, ...). Entonces, la estructura resultante nos facilitará la *Base* para comenzar el diálogo SAD/experto que nos permitirá mejores intuiciones sobre el problema y refinar la consulta a la luz de nueva evidencia

hasta que la respuesta satisfaga al experto.

Por simplicidad, sean $\mathbf{X} \subset \mathcal{R}^{n_1}, \mathbf{Y} \subset \mathcal{R}^{n_2}, n_1 + n_2 = n$ que denotan, respectivamente, a los atributos instanciados y no instanciados de la consulta. Nuestro **algoritmo** $\mathcal{A}2$ sigue los siguientes pasos:

S0. Inicialización $\mathbf{X}^0 = \mathbf{X}, \mathbf{Y}^0 = \mathbf{Y}$.

S1. El SAD extrae los datos (registros) que cumplen $\mathbf{X}^0$ en la lista *KBM2L* operativa.

S2. El SAD aprende una RB de los datos (la estructura y las probabilidades condicionadas).

S3. El SAD calcula $P(R = r|\mathbf{X}^0), \forall r \in \Omega_R$ en la RB. El experto establece un criterio de decisión, normalmente la moda de la distribución, para elegir entre las r's. Sea m^0 este valor, que consideramos evidencia a propagar por la red.

S4. El SAD calcula $P(Y_j = y|R = m^0, \mathbf{X}^0), \forall j = 1, ..., n_2, \forall y \in \Omega_{Y_j}$ en la RB. El experto establece un criterio de decisión, normalmente la varianza mínima, para elegir entre los Y_j's. Sea $\widetilde{\mathbf{Y}}^0$ el vector resultante de Y_j's, con las coordenadas dadas por las instanciaciones del experto, como por ejemplo, la moda de Y_j.

S5. Extendemos el vector $\mathbf{X}^0$ como $\mathbf{X}^1 = \mathbf{X}^0 \cup \widetilde{\mathbf{Y}}^0$. Actualizamos $\mathbf{Y}^1 = \mathbf{Y}^0 \setminus \widetilde{\mathbf{Y}}^0$.

Los pasos *S3* y *S4* se repiten hasta que el experto esté satisfecho, o bien $\mathbf{Y}^j$ tenga pocos componentes y se llama al algoritmo $\mathcal{A}1$ para continuar. Si el algoritmo se detiene en $\mathbf{X}^j$, entonces m^j es la respuesta devuelta por el sistema. El experto puede siempre revisar las decisiones hechas en *S3* y *S4* de la última iteración si no está de acuerdo con los resultados provisionales. Más aún, el SAD vigilará para advertir al experto de las probabilidades condicionadas de eventos imposibles (registrados en la BC).

En el paso *S2* se aprende una RB (véase la revisión (Buntine, 1996)) mediante un algoritmo de aprendizaje de la estructura, donde el algoritmo *K2* (Cooper y Herskovits, 1992) es el estándar. K2 es un algoritmo voraz de búsqueda de padres y comienza asumiendo que un nodo no tiene padres. Usa una métrica bayesiana para medir la bondad de las diferentes estructuras en relación a los datos. Asume un orden total (que no es revisado durante el proceso de aprendizaje) en los nodos, y para cada nodo, K2 devuelve de forma incremental cuáles son sus padres más probables dados los datos. El algoritmo trabaja con tamaños del conjunto de datos bastante razonables, como en el contexto que proponemos. Pero, de hecho, podríamos tener incluso tamaños muy grandes de donde podríamos muestrear para que el aprendizaje fuera computacionalmente tratable.

Los criterios de decisión del experto en los pasos *S3* y *S4* podrían ser distintos. Son elecciones del experto. Con respecto a los criterios sugeridos:

1. en el paso *S3*, elegimos la respuesta más probable dado el conjunto de atributos instanciados para la consulta;

2. en el paso *S4*, elegimos el atributo(s) en el que tenemos más confianza o el atributo(s) de variabilidad menor que un umbral fijado.

Entonces, se instancian en la consulta con su moda, dando lugar a una nueva y más precisa pregunta que subsume a la pregunta inicial. Los pasos siguientes permiten actualizar continuamente la probabilidad.

El SAD habla mediante la RB y sus probabilidades, dando primeramente información acerca de la respuesta y, después, acerca de la probabilidad de cada atributo abierto, dada la respuesta elegida por el experto. Esta ayuda tanto para respuestas como para consultas permite al experto replantear y mejorar su consulta hasta que está más definida alcanzando respuestas más precisas. Ésta es la tarea (E) mencionada anteriormente.

La *Situación (iv)* también presenta diferentes valores de la política en S, algunos de los cuales pueden ser, sin embargo, desconocidos. Las políticas desconocidas son el resultado de tener un sistema que puede legítimamente llamarse basado en el conocimiento, debido a su tamaño significativo (Henrion et al., 1991), pero con la imposibilidad de ser completamente evaluado. El experto juega un importante papel en la decisión de qué partes del problema deben ser evaluadas, sugiriendo cuál es el núcleo de conocimientos que le interesan. Como consecuencia, sus consultas probablemente tendrían que ver con el núcleo, y tendrán así una respuesta conocida. Por tanto, la situación (iv) puede resolverse como la situacion (iii), intentando evitar políticas desconocidas vía, por ejemplo, el algoritmo $\mathcal{A}1$.

4.4. Aplicación al Sistema IctNeo

4.4.1. Descripción del Problema y del Sistema de Ayuda a la Decisión

IctNeo es un SAD complejo para la gestión de la ictericia neonatal, un problema médico muy común en el cual la bilirrubina se acumula al no ser eliminada por el hígado a una velocidad normal (Newman y Maisels, 1992). Su primera versión está ya implementada en el Servicio de Neonatología del Hospital General Universitario Gregorio Marañón de Madrid (Gómez, 2002).

El modelo incluye el proceso de ingreso, tratamiento y alta del paciente. Sus principales objetivos son: incluir numerosos factores con incertidumbre y decisiones, definir cuándo es mejor

realizar y/o cambiar el tratamiento, disminuir los costes del diagnóstico y de las fases terapéuticas, disminuir los riesgos debidos, por ejemplo, a una completa transfusión de sangre, que algunas veces se necesita, y tener en cuenta las preferencias de los médicos y los padres (decisiones médicas compartidas (Frosh y Kaplan, 1999)). El sistema trabaja con casos cerrados, comparando sus recomendaciones con las acciones *ya* realizadas por los doctores al paciente en cuestión. El hospital esperaba contar con una herramienta automática para este problema de toma de decisiones y una ayuda para mejorar la gestión de la ictericia, obteniendo una mejor compresión del problema.

IctNeo representa y resuelve el problema por medio de un DI, véase la Figura 4.6. Aunque es muy simple conceptualmente, la aplicación de la metodología de los DI puede ser extremadamente crítica para problemas grandes. El desarrollo de nuestro DI fue muy complejo y consumió mucho tiempo incluyendo tareas relativas a la estructuración del problema y la adquisición de conocimiento (probabilidades y utilidades), véase (Bielza et al., 2000) y (Gómez et al., 2000).

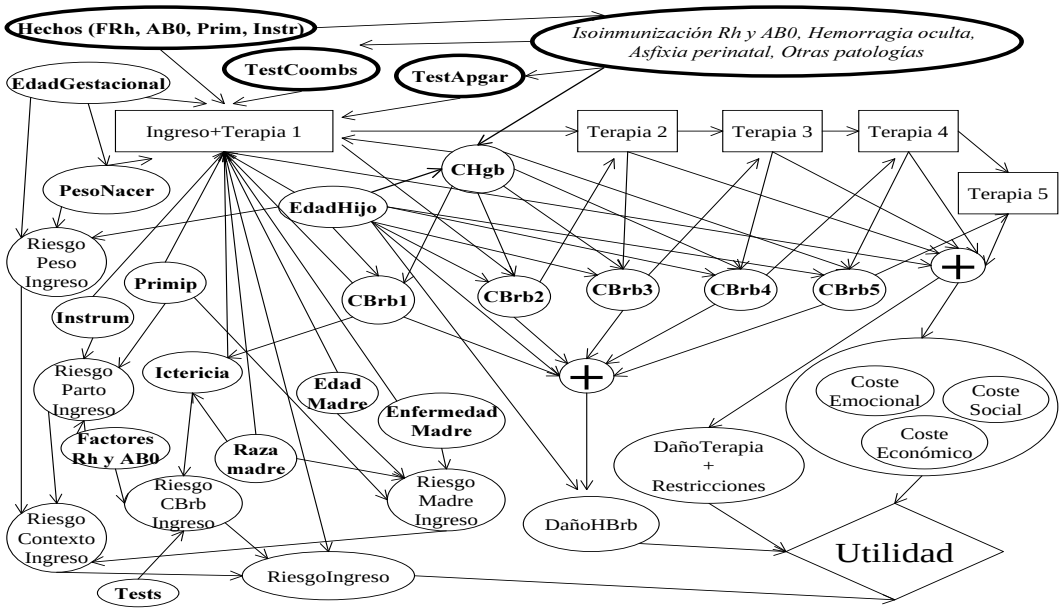

Figura 4.6: Diagrama de influencia de *IctNeo*, donde las elipses en negrita representan grupos afines de variables

La evaluación del diagrama fue también muy díficil. En nuestro caso, nuestro gran DI es computacionalmente intratable, principalmente porque las necesidades de espacio de almacenamiento (de las tablas) crecen enormemente durante el proceso de evaluación del problema debido a las herencias de los nodos relativas al borrado de nodos de azar e inversión de arcos

(los requisitos de memoria son del orden de 10^{24} posiciones de memoria).

IctNeo añade algunos procedimientos al algoritmo de evaluación estándar (Ríos-Insua et al., 2000). Una forma de proceder es evaluar el DI mediante instanciación de evidencia en algunos nodos (Ezawa, 1998), lo que equivale a resolver el problema para cada paciente particular una vez que esta evidencia se ha propagado y se ha actualizado el DI. Por tanto, evaluamos tantos diagramas como casos diferentes sean estudiados, pero las TDO son más pequeñas.

De todas formas, las tablas son todavía demasiado grandes, y deben ser accedidas por el SAD, que propone los tratamientos almacenados en la BC. Por ello, el sistema *IctNeo* ha motivado nuestras ideas y servirá para mostrar los resultados. Detallamos a continuación esta información.

Actualmente, como hemos dicho anteriormente, *IctNeo* está instalado en el hospital y funciona con unos requisitos mínimos de almacenamiento (1 megabyte). Esta versión tiene un DI con cinco nodos de decisión. Cubre las primeras 96 horas de vida del recién nacido, que son las más críticas. El diagrama, véase la Figura 4.6, tiene 61 nodos, 137 arcos, 5586 parámetros de probabilidad, con una TPC de mayor tamaño de 748 entradas, y una tabla inicial de utilidad en el nodo de valor con 5400 entradas.

Evaluamos parcialmente el diagrama mediante la instanciación de la evidencia en 17 nodos. Seleccionamos un conjunto de instancias representativo del problema real para los médicos. Cuando dividimos el proceso de evaluación mediante instanciaciones de la evidencia, elegimos el número de atributos adecuado para manejar el tamaño de las TDO y producir el menor impacto en la propagación de la incertidumbre (Ezawa, 1998). Cada evaluación mediante instanciación llevó aproximadamente una hora en un ordenador personal con procesador Pentium^{TM} II a 200-MHz y 128 MB. El tamaño de memoria máximo alcanzado durante la evaluación fue de 10^6 posiciones de memoria (un número real necesita 10 bytes). La estrategia óptima está soportada en cinco TDO. Esto conduce a 67 posibles tratamientos completos (de 5 etapas) para un paciente dado. La primera decisión depende de diecinueve variables, mientras que la segunda fase depende de dieciocho variables, de la primera decisión y de otras dos variables más. Con el propósito de ilustrar la técnica de gestión de las TDO utilizando listas *KBM2L*, sólo mostramos aquí la segunda decisión, véase el Cuadro 4.4. La primera columna indica el conjunto de atributos que constituye el esquema de la lista correspondiente, mientras que la segunda contiene sus respectivos dominios o los valores instanciados. La instanciación de ciertos atributos determina el conjunto de subproblemas que evaluamos. El esquema del resto de decisiones es análogo a éste de la segunda decisión.

Una entrada en las TDO es una instancia completa de un caso (paciente). Todos los atributos toman ciertos valores de sus dominios. Cuando el caso en cuestión es localizado en las tablas,

Cuadro 4.4: Estructura de la TDO de la segunda decisión

Nombre de los atributos	Dominio de los atributos
Terapia 2	SinTerapia2, AltaMédica, Observación-12h Fototerapia-6h, Fototerap-12h, Fototerap-24h Exanguinotransfusión
Atributos instanciados	
Edad del Hijo (EdadHijo)	menos de 36 horas, entre 36 horas y 72 h., más de 72 h.
Factor Rh del Hijo (FactorRhHijo)	Negativo, Positivo
Factor Rh de la Madre (FactorRhMadre)	Negativo, Positivo
Factor AB0 del Hijo (FactorABoHijo)	CERO, A, B, AB
Factor AB0 del Madre (FactorABoMadre)	CERO, A, B, AB
Test de Apgar (TestApgar)	entre 0 y 3, 4 y 7, 8 y 10
TCIF Rh del Hijo (TCIFRhHijo)	Negativo, Positivo
TCIF Rh de la Madre (TCIFRhMadre)	Negativo, Positivo
TCIF AB0 del Hijo (TCIFAB0Hijo)	Negativo, Positivo
Edad Gestacional (EdadGestación)	menos de 36 semanas, 36 s., más de 36 s.
Peso al Nacer (PesoNacer)	entre 500 y 1000 gr., entre 100 y 1500 gr., entre 1500 y 2500 gr., más de 2500 gr.
Madre Primípara? (Primípara)	Sí, No
Parto Instrumental (PartoInstrumental)	Natural, Instrumental
Ictericia (Ictericia)	Normal, Amarillo (A), Pies–A, A–Calabaza
Raza de la Madre (Raza)	Caucasiana, Gitana, Asiática, Negra
Edad de la Madre (EdadMadre)	entre 15 y 18 años, 19 y 35 a., más de 35 a.
Enfermedad de la Madre (EnfermedadMadre)	No, Sí
Atributos no instanciados	
Concentr. de Bilirrubina 1 (CBrb1)	Normal, Patológica, Muy Patológica
Concentr. de Bilirrubina 2 (CBrb2)	Normal, Patológica, Muy Patológica
Terapia 1	SinIngreso, Observación-6h, Obs-12h, Obs-24h Fototerapia-6h, Fototerap-12h, Fototerap-24h
Concentr. de Hemoglobina (CHgb)	Normal, Patológica, Muy Patológica

TCIF: Test de Coombs de Isoinmunización del Factor Rh/AB0

A: Piel amarilla

la utilidad esperada $U_{terapia,i}$, de cada posible alternativa i es mostrada. Por ejemplo, véase el Cuadro 4.5.

Veamos qué tamaño tienen las TDO. Teniendo en cuenta el cardinal de cada dominio de los

Cuadro 4.5: Decisiones en *IctNeo*, alternativas y utilidades

Terapia 1		Terapia 2		Terapia 3	
SinIngreso	$U_{1,0}$	SinTerapia 2	$U_{2,0}$	SinTerapia 3	$U_{3,0}$
Observación-6h	$U_{1,1}$	AltaMédica	$U_{2,1}$	AltaMédica	$U_{3,1}$
Observación-12h	$U_{1,2}$	Observación-12h	$U_{2,2}$	Observación-12h	$U_{3,2}$
Observación-24h	$U_{1,3}$	Fototerapia-6h	$U_{2,3}$	Fototerapia-6h	$U_{3,3}$
Fototerapia-6h	$U_{1,4}$	Fototerapia-12h	$U_{2,4}$	Fototerapia-12h	$U_{3,4}$
Fototerapia-12h	$U_{1,5}$	Fototerapia-24h	$U_{2,5}$	Fototerapia-24h	$U_{3,5}$
Fototerapia-24h	$U_{1,6}$	Exanguinotransfusión	$U_{2,6}$	Exanguinotransfusión	$U_{3,6}$

Terapia 4		Terapia 5	
SinTerapia 4	$U_{4,0}$	SinTerapia 5	$U_{5,0}$
AltaMédica	$U_{4,1}$	AltaMédica	$U_{5,1}$
Observación-12h	$U_{4,2}$	ContinúaIngresado	$U_{5,2}$
Fototerapia-12h	$U_{4,3}$		
Fototerapia-24h	$U_{4,4}$		

17 atributos a instanciar, el número de instancias a realizar es:

$$\delta_{EdadHijo} \cdot \delta_{FactorRhHijo} \cdot \delta_{FactorRhMadre} \cdot \delta_{FactorAB0Madre} \cdot \delta_{FactorAB0Hijo} \cdot \delta_{TestApgar} \cdot \delta_{TestCoombsRhHijo} \cdot \delta_{TestCoombsRhMadre} \cdot \delta_{TestCoombsAB0Hijo} \cdot \delta_{EdadGestacion} \cdot \delta_{PesoNacer} \cdot \delta_{MadrePrimipara} \cdot \delta_{PartoInstrumental} \cdot \delta_{Ictericia} \cdot \delta_{EdadMadre} \cdot \delta_{RazaMadre} \cdot \delta_{EnfermedadMadre} =$$

$$3 \cdot 2 \cdot 2 \cdot 4 \cdot 4 \cdot 3 \cdot 2 \cdot 2 \cdot 2 \cdot 3 \cdot 4 \cdot 2 \cdot 2 \cdot 4 \cdot 3 \cdot 4 \cdot 2 = 21233664 \text{ instancias}$$

El número de entradas (combinaciones) para la tabla de Terapia 2 es:

$$\delta_{CBrb1} \cdot \delta_{CBrb2} \cdot \delta_{Terapia1} \cdot \delta_{CHgb} =$$
$$3 \cdot 3 \cdot 7 \cdot 3 = 189 \text{ entradas}$$

Por consiguiente, la tabla de Terapia 2 completa tiene $21233664 \cdot 189 = 4013162496$ entradas. Cada entrada tiene siete utilidades esperadas asociadas. Así, se necesitan $4013162496 \cdot 7 =$

28092137472 números reales. Cada número precisa de 10 bytes y de ahí se requiere un total de 280921374720 bytes $\sim$ 280 gigabytes. El tamaño de la tabla es excesivo.

La TDO de Terapia 5 es incluso peor. Recoge hasta 5,309,410,000,000 combinaciones diferentes. Más de 150 terabytes serían necesarios en el sistema de ficheros. En resumen, hay una necesidad clara de evaluar el problema de forma práctica. En primer lugar seleccionamos con ayuda del médico los subproblemas (instancias)de mayor interés y tratamos de resumir el contenido de las TDO mediante las listas *KBM2L*.

El problema requiere, a juicio de los médicos de 1000 instancias (se pueden evaluar en paralelo), obteniendo las 5 TDO finales con un tamaño de 18 megabytes. Los resultados son posteriormente compuestos de forma incremental en la estructura *KBM2L*, según explicamos en el capítulo 2. Así, es posible potencialmente proporcionar un procedimiento de inferencia de modelos muy complejos a partir de evaluaciones parciales instanciadas. Mostramos ahora los resultado de hacer esto por medio del mecanismo de aprendizaje descrito en el capítulo 3.

4.4.2. Resultados de la *KBM2L*

Recordemos que hemos seleccionado 17 atributos (EdadHijo, FactorRhHijo, ...) para *IctNeo*, cada uno con dos, tres o más valores en sus dominios, es decir, varios millones de instancias. La evaluación de cada subproblema (instancia del problema) genera 5 tablas con 9 (Terapia 1), 189 (Terapia 2), 3969 (Terapia 3), 83349 (Terapia 4) y 250047 (Terapia 5) entradas almacenándose en 18 megabytes. Consideramos en la instancia sólo atributos que pertenecen al esquema de todas las TDO del modelo. De esta forma evitamos introducir evidencia del futuro en las TDO iniciales. Seleccionamos mil instancias y las fusionamos en la *KBM2L*. El proceso de aprendizaje nos ha permitido construir la *KBM2L* para esta *Base* con las instancias del problema. Las 1000 instancias son copiadas paso a paso en la *KBM2L*.

La *KBM2L* de Terapia 2 tiene 260 ítems que cubren un conjunto de 189000 casos evaluados en el espacio total de la tabla de 4013162496 combinaciones. Encontramos ítems desconocidos (*unKB*) porque no hemos evaluado la totalidad del problema. Realmente evaluamos un conjunto de instancias que representa sólo el 0.0047 % de la tabla. Los ítems *unKB* en este caso no son interesantes para los médicos y si algunos de ellos lo fuesen deberíamos evaluar y aprender los correspondientes subproblemas.

La *Base* inicial de la lista *KBM2L* correspondiente a la tabla de Terapia 2 es, después del procedimiento de aprendizaje de las TDO suministradas por la instanciación y la propagación de la alternativa óptima de la Terapia 1:

$$B_1 = [\mathit{CBrb1}, \mathit{CBrb2}, \mathit{CHgb}, \mathit{EdadHijo}, \mathit{FactorRhHijo}, \mathit{FactorRhMadre}, \mathit{FactorAB0Hijo},$$

FactorAB0Madre, *TestApgar*, *TCIFRhHijo*, *TCIFRhMadre TCIFAB0Hijo*, *EdadGestacion*, *PesoNacer*, *Primípara*, *PartoInstrumental*, *Ictericia*, *EdadMadre*, *RazaMadre*, *EnfermedadMadre*].

Utilizamos una tabla de doble columna para presentar un fragmento de la lista, véase el Cuadro 4.6 para la política óptima de Terapia 1 y 2 (agregada). La columna izquierda muestra las propuestas del sistema. La columna derecha incluye los atributos asociados con cada propuesta. La parte fija de cada ítem se muestra en tipo de fuente negrita.

La fase de optimización muestra que la *Base* inicial no es óptima y alterando el orden de los atributos podemos conseguir una lista más corta y refinar el conocimiento sobre los atributos decisivos.

Después de unos 300 $XBase$, mediante el AEV, obtenemos una representación del conocimiento con tan sólo 16 granos (ítems). La *Base* óptima es:

B_f = [*CBrb1*, *CBrb2*, *CHgb*, *EdadGestacional*, *PesoNacer*, *Primípara*, *EdadHijo*, *PartoInstrumental*, *FactorRhHijo*, *FactorRhMadre*, *FactorAB0Hijo*, *FactorAB0Madre*, *TestApgar*, *TCIFRhHijo*, *TCIFRhMadre*, *TCIFAB0Hijo*, *Ictericia*, *EdadMadre*, *RazaMadre*, *EnfermedadMadre*]

B_f conduce a una lista más corta, óptima, véase el Cuadro 4.7. Esta lista puede leerse como un conjunto de 16 reglas que indican la política global óptima como función de los atributos clave, la parte fija del ítem.

Por ejemplo, la regla 7 recomienda realizar una Fototerapia de 24h como Terapia 2 si el paciente presenta un nivel Patológico de CBrb2 y se le sometió a Fototerapia de 24 horas como Terapia 1 debido a los niveles Patológicos de CBrb1 y CHgb. El resto de características del paciente (su edad de gestación, el peso al nacer adecuado a su edad gestacional, que sea hijo de madre Primípara,...) no son relevantes para el tratamiento.

En la lista podemos estudiar la influencia de la variación de los valores de la explicación sobre la política óptima. Los tratamientos de las reglas 6 (Terapia 1: Fototerapia24h, Terapia 2: Fototerapia12h) y 7 (Terapia 1: Fototerapia24h, Terapia 2: Fototerapia24h) son diferentes debido a que la CBrb2 es normal en la regla 6. Respecto de las reglas 6 (Terapia 1: Fototerapia24h, Terapia 2: Fototerapia12h) y 11 (Terapia 1: Fototerapia12h, Terapia 2: Fototerapia12h) son diferentes debido a que CHgb es normal en la regla 11.

Los resultados han alcanzado los siguientes objetivos:

1. Una importante reducción de los requisitos de memoria para almacenar la BC.

Cuadro 4.6: (fragmento de) Lista *KBM2L* de la TDO de Terapia 1 y 2, 260 ítems

Terapia 2 Terapia 1	Atributos
unKB	...
SinTerapia 2 SinIngreso	**CBrb2:Normal,CBrb1:Normal,** **CHgb:Normal,EdadHjo1:menos de 36 horas,EdadGestación:36 semanas,** **PesoNacer:más de 2500 gr.,Primípara:No,PartoInstrumental:Natural,** FactorRhHijo:Negativo,FactorRhMadre:Negativo, FactorAB0Hijo:AB,FactorRhMadre:AB,TestApgar:entre 8 y 10, TCIFRhHijo:Negativo,TCIFRhMadre:Negativo,TCIFAB0Hijo:Negativo, Ictericia:Amarillo,EdadMadre:entre 19 y 35,RazaMadre:Caucasiana, EnfermedadMadre:No
unKB	...
Observ-6h Fototerapia6h	**CBrb2:Normal,CBrb1:Patológica**, **CHgb:Normal,EdadHjo1:menos de 36 horas,EdadGestación:36 semanas,** **PesoNacer:más de 2500 gr.,Primípara:No,PartoInstrumental:Natural,** ...
unKB	...
SinTerapia 2 SinIngreso	**CBrb2:Normal,CBrb1:Normal**, **CHgb:Normal,EdadHjo1:menos de 36 horas,EdadGestación:36 semanas,** **PesoNacer:más de 2500 gr.,Primípara:No,PartoInstrumental:Natural,** ...
unKB	...
Observ-6h Fototerapia6h	**CBrb2:Normal,CBrb1:Patológica**, **CHgb:Patológica,EdadHjo1:menos de 36 horas,EdadGestación:36 semanas,** **PesoNacer:más de 2500 gr.,Primípara:No,PartoInstrumental:Natural,** ...
unKB	...
SinTerapia 2 SinIngreso	**CBrb2:Normal,CBrb1:Normal**, **CHgb:Patológica,EdadHjo1:menos de 36 horas,EdadGestación:36 semanas,** **PesoNacer:más de 2500 gr.,Primípara:No,PartoInstrumental:Natural,** ...
unKB	...
Alta Observación12h	**CBrb2:Normal,CBrb1:High**, **CHgb:Patológica,EdadHjo1:menos de 36 horas,EdadGestación:36 semanas,** **PesoNacer:más de 2500 gr.,Primípara:Sí,PartoInstrumental:Natural,** ...
unKB	...
SinTerapia 2 SinIngreso	**CBrb2:Patológica,CBrb1:Normal**, **CHgb:Patológica,EdadHjo1:menos de 36 horas,EdadGestación:36 semanas,** **PesoNacer:más de 2500 gr.,Primípara:No,PartoInstrumental:Natural,** ...
unKB	...
Exanguinotransfusión Fototerapia6h	**CBrb2:Muy Patológica,CBrb1:Muy Patológica**, **CHgb:Patológica,EdadHjo1:menos de 36 horas,EdadGestación:menos de 36 semanas,** **PesoNacer:más de 2500 gr.,Primípara:No,PartoInstrumental:Natural,** ...
unKB	...
SinTerapia 2 SinIngreso	**CBrb2:Normal,CBrb1:Normal**, **CHgb:Patológica,EdadHjo1:menos de 36 horas,EdadGestación:36 semanas,** **PesoNacer:más de 2500 gr.,Primípara:No,PartoInstrumental:Instrumental,** ...
unKB	...
Exanguinotransfusión Fototerapia6h	**CBrb2:Muy Patológica,CBrb1:Muy Patológica,Terapia 1:Fototerapia6h**, **CHgb:Patológica,EdadHjo1:menos de 36 horas,EdadGestación:36 semanas,** **PesoNacer:más de 2500 gr.,Primípara:Sí,PartoInstrumental:Instrumental,** ...
unKB	...
Fototerapia12h Fototerapia12h	**CBrb2:Patológica,CBrb1:Muy Patológica**, **CHgb:Patológica,EdadHjo1:menos de 36 horas,EdadGestación:36 semanas,** **PesoNacer:más de 2500 gr.,Primípara:No,PartoInstrumental:Instrumental,** ...
unKB	...
Exanguinotransfusión Fototerapia6h	**CBrb2:Muy Patológica,CBrb1:Muy Patológica**, **CHgb:Patológica,EdadHjo1:menos de 36 horas,EdadGestación:36 semanas,** **PesoNacer:más de 2500 gr.,Primípara:Sí,PartoInstrumental:Instrumental,** ...
unKB	...

Cuadro 4.7: Lista *KBM2L* óptima de la TDO del sistema *IctNeo* (Terapia 1 y 2), 16 ítems

Regla	Politica óptima	Atributos evaluados
1	SinTerapia 2 SinIngreso	**CBrb2=Normal, CBrb1=Normal, CHgb=Normal**, EdadGestacional:36 semanas,...
	...	$unKB$
2	Alta Observación6h	**CBrb2=Normal, CBrb1=Normal, CHgb=Normal**, EdadGestación:36 semanas,...
	...	$unKB$
3	Alta Observación12h	**CBrb2=Normal, CBrb1=Patológica, CHgb=Normal**, EdadGestación:36 semanas,...
4	Observación6h Fototerapia6h	**CBrb2=Normal, CBrb1=Patológica, CHgb=Normal**, EdadGestación:36 semanas,...
5	Observación12h Fototerapia6h	**CBrb2=Normal, CBrb1=Patológica, CHgb=Patológica**, EdadGestación:36 semanas,...
6	Fototerapia12h Fototerapia24h	**CBrb2=Normal, CBrb1=Patológica, CHgb=Patológica**, EdadGestación:36 semanas,...
7	Fototerapia24h Fototerapia24h	**CBrb2=Patológica, CBrb1=Patológica, CHgb=Patológica**, EdadGestación:36 semanas,...
	...	$unKB$
8	Observación12h Fototerapia12h	**CBrb2=Patológica, CBrb1=Patológica, CHgb=Patológica**, EdadGestación:36 semanas,...
9	Fototerapia24h Fototerapia24h	**CBrb2=Normal, CBrb1=Patológica, CHgb=Normal**, EdadGestación:36 semanas,...
	...	$unKB$
10	Fototerapia24h Fototerapia12h	**CBrb2=Normal, CBrb1=Patológica, CHgb=Patológica**, **EdadGestación:menos de 36 semanas**,...
11	Fototerapia12h Fototerapia12h	**CBrb2=Normal, CBrb1=Patológica, CHgb=Normal**, EdadGestación:36 semanas,...
	...	$unKB$
12	Observación12h Fototerapia12h	**CBrb2=Normal, CBrb1=Patológica, CHgb=Normal**, EdadGestación:más de 36 semanas,
13	Fototerapia12h Fototerapia6h	**CBrb2=Normal, CBrb1=Patológica, CHgb=Normal**, EdadGestación:36 semanas,...
14	Alta Observación12h	**CBrb2=Normal, CBrb1=Patológica, CHgb=Normal**, , EdadGestación:más de 36 semanas,...
15	Observación12h Fototerapia24h	**CBrb2=Normal, CBrb1=Patológica, CHgb=Normal**, **EdadGestación:menos de 36 semanas**,...
	...	$unKB$
16	Exanguinotransfusión Fototerapia6h	**CBrb2=Muy Patológica, CBrb1=Patológica**, **CHgb=Muy Patológica**, EdadGestación:36 semanas, PesoNacer:más de 2500 gr., Primípara:No,...
	...	$unKB$

2. La reducción de las TDO en varios órdenes de magnitud, desde 260 a 16 ítems para la lista *KBM2L* de Terapia 1 y 2. En el Cuadro 2 recogemos los resultados obtenidos sobre las cinco TDO. La primera fila muestra los tamaños en número de casos, la segunda fila muestra el número de ítems de la lista *KBM2L* inicial y la tercera fila muestra el tamaño de la lista en la *Base* óptima.

3. La disponibilidad de las reglas de toma de decisiones del protocolo explícitamente.

4. Un medio de observar la influencia de la variación de los valores de los atributos relevantes en las reglas de toma de decisiones y, por tanto, un procedimiento práctico de obtención

	Terapia 1	Terapia 2	Terapia 3	Terapia 4	Terapia 5
Tamaño de la tabla	9000	189000	3969000	83349000	250047000
Tamaño de la *KBM2L* inicial	12	260	1020	6320	8202
Tamaño de la *KBM2L* óptima	7	16	21	30	11

de explicaciones de las políticas óptimas que hemos obtenido al evaluar el DI.

4.4.3. Consultas

Las consultas abiertas y sus respectivas respuestas y explicaciones son los elementos básicos del diálogo en la interacción experto/SAD (Fernández del Pozo y Bielza, 2002a). Ahora describiremos cómo se desarrolla el diálogo en sesiones típicas. Nuestro problema de decisión es una simplificación de *IctNeo*; se refiere a un médico que decide acerca del ingreso de un paciente y dos posibles acciones terapéuticas, es decir, tres alternativas: r_0: No Ingresa, r_1: 12 horas de Observación y r_2: 12 horas de Fototerapia. La TDO tiene 12 atributos, véase el Cuadro 4.8, y 82944 casos. La *Base* óptima es $B = [A_0, A_1, ..., A_{11}]$, con los pesos respectivos $\overrightarrow{w} = (27648, 13824, 6912, 3456, 1152, 576, 192, 64, 16, 8, 4, 1)$. Cada consulta se codificará conforme al orden de la *Base* óptima. Nuestra BC tiene 1656 ítems distribuidos sobre 27736 casos conocidos y 55208 casos desconocidos.

Cuadro 4.8: Atributos y dominios

Atributos	Dominios
Concentr. de Bilirrubina(A_0)	Normal(0), Patológica(1), MuyPatológica(2)
Factores Rh del Hijo(A_1)	Negativo(0), Positivo(1)
Factores Rh de la Madre(A_9)	Negativo(0), Positivo(1)
Primípara?(A_2)	Primípara(0), Multipara(1)
Parto Instrumental?(A_3)	Natural(0), Instrumental(1)
Edad de la Madre(A_4)	15-18(0), 19-35(1), >35(2)
TCIF Rh del Hijo(A_5)	Negativo(0), Positivo(1)
TCIF Rh de la Madre(A_{10})	Negativo(0), Positivo(1)
Test de Apgar de 5 min.(A_6)	0-3(0), 4-7(1), 8-10(2)
Concentr. de Hemoglobina(A_7)	Normal(0), Patológica(1), MuyPatológica.(2)
Peso al Nacer(A_8)	0.5-1.0(0), 1.0-1.5(1), 1.5-2.5(2), >2.5(3) kg
Ictericia(A_{11})	Normal(0), A.(1), Pies–A.(2), A–Calabaza(3)

Consideremos a continuación las cuatro situaciones descritas en el esquema de diálogo en la sección 4.3, véanse los Cuadros 4.9 y 4.10.

1. *Situación (i)*: la consulta del médico es (0,0,0,0,0,0,1,0,*,*,*,*), con cuatro atributos abiertos a la derecha de la *Base*, de modo que coinciden las *Bases* óptima y operativa. La respuesta es que r_1 es la política óptima. Su explicación es la parte fija del índice del correspondiente ítem, esto es, A_0 es 0,..., A_7 es 0, véase el Cuadro 4.9.

2. *Situación (ii)*: la consulta del médico es (0,0,0,0,0,1,2,0,3,*,*,*). En este caso, la respuesta y su explicación no están disponibles (ND) puesto que la consulta cubre sólo respuestas desconocidas, véase el Cuadro 4.9. Entonces, encontramos los desplazamientos más próximos con respuesta conocida. Se sigue que $q^l = 447, q^u = 1151$, ambos con política r_1. De este modo obtenemos $d_1 = 561, d_2 = 128$, y la respuesta es r_1. q^u está localizado en un ítem con 64 registros entre los desplazamientos 1151 y 1215. La parte fija de este ítem es $(0, 1, 2, 3, 4, 5, 6, 7)$ con valores $(0, 0, 0, 0, 1, 0, 0, 0)$, que es la explicación de la respuesta r_1.

3. *Situación (iii)*: la consulta del médico es (0,0,1,0,1,1,*,*,*,*,*,*), con seis atributos abiertos, véase el Cuadro 4.10. Hay dos posibles respuestas, cada una con su propia explicación. Podemos llamar a los algoritmos $\mathcal{A}1$ o $\mathcal{A}2$.

   Respecto a $\mathcal{A}1$, el SAD sugiere al médico que instancie uno de los seis atributos abiertos, siendo el A_6 un atributo eficiente con el cual empezar, puesto que está más a la izquierda en la *Base*. Si esto no es posible, al médico se le requeriría los valores de los atributos $A_7, \ldots, A_{11}$, en este orden, intentando minimizar el tamaño del actual conjunto de alternativas en la respuesta. En este caso, el médico dice que el valor de A_6 es 0, y el SAD responde con r_2 como política óptima. Con un único valor de política, la sesión termina. La explicación es (0,0,1,0,1), véase el Cuadro 4.10.

4. *Situación (iv)*: la consulta del médico es (0,0,0,0,0,0,0,*,*,*,*,*). Las posibles respuestas son r_1 o desconocido, véase el Cuadro 4.10. La aplicación de $\mathcal{A}1$ implica solicitar los valores de los atributos $A_7, \ldots, A_{11}$, en este orden. El médico dice que el valor de A_7 es 0 y el SAD responde con r_1, y la sesión termina. La explicación es (0,0,0,0,0,0,0,0), véase el Cuadro 4.10.

Para ilustrar el algoritmo $\mathcal{A}2$ retomamos la situación (iii), dado que el conjunto de registros involucrados en la consulta abierta de la situación (iv) es demasiado pequeño para aprender una RB. La consulta planteada en la situación (iii) involucra a más de 500 registros. El algoritmo K2 está integrado en nuestro software de gestión de listas *KBM2L*. Los cálculos de probabilidad

Cuadro 4.9: Detalles de la consulta en las situaciones (i) y (ii)

	Situación (i)	Situación (ii)
Consulta abierta	(0,0,0,0,0,0,1,0,*,*,*,*)	(0,0,0,0,0,1,2,0,3,*,*,*)
Indice menor	(0,0,0,0,0,0,1,0,0,0,0,0)	(0,0,0,0,0,1,2,0,3,0,0,0)
Indice mayor	(0,0,0,0,0,0,1,0,3,1,1,3)	(0,0,0,0,0,1,2,0,3,1,1,3)
p^l, p^u desplazamientos	192, 255	1008, 1023
Parte fija	(0,. . . ,7)	Explicación ND
Respuesta	r_1	desconocida

Cuadro 4.10: Detalles de la consulta en las situaciones (iii) y (iv)

	Situación (iii)	Situación (iv)
Consulta abierta	(0,0,1,0,1,1,*,*,*,*,*,*)	(0,0,0,0,0,0,0,*,*,*,*,*)
Indice menor	(0,0,1,0,1,1,0,0,0,0,0,0)	(0,0,0,0,0,0,0,0,0,0,0,0)
Indice mayor	(0,0,1,0,1,1,2,2,3,1,1,3)	(0,0,0,0,0,0,0,2,3,1,1,3)
p^l, p^u desplazamientos	8640, 9215	0, 191
Parte fija	(0,. . . ,3) para r_1	(0,. . . ,7) para r_1
	(0,. . . ,4) para r_2	Explicación ND para r desconocida
Respuesta	r_1 o r_2	r_1 o desconocida

se realizaron mediante Hugin Expert Software (Hugin, 2002), una vez que la RB aprendida fue exportada al formato *NET* propio de Hugin. La RB aprendida tiene la topología de un clasificador naive-Bayes (Domingos y Pazzani, 1996) con los arcos en sentido inverso, es decir, con cinco nodos explicativos (desde A_0 a A_4) como padres del nodo de política. La RB da las probabilidades 0, 0.014, 0.986 a r_0, r_1 y r_2, respectivamente, dados los datos. El médico elige la moda es decir, $m^0 = r_2$, con explicación (0,0,1,0,1) como anteriormente. La sesión termina, puesto que el médico está satisfecho, no requiriendo el cálculo de $\widetilde{\mathbf{Y}}^0$.

Como conclusión final podemos decir que la síntesis de conocimiento obtenida mediante la lista *KBM2L* es útil desde el punto de vista computacional y semántico. Este último facilita tareas relativas a la validación del SAD, la construcción de explicaciones de alto nivel, la revelación de interrelaciones entre atributos y el análisis de consultas complejas.

Capítulo 5

KBM2L para Tablas de Probabilidad

El capítulo 5 muestra la aplicación de las técnicas del capítulo 2 al estudio, análisis y gestión de tablas de probabilidad. En particular, veremos TPC complejas.

En la sección 5.1 describimos los problemas relacionados con las tablas de las distribuciones de probabilidad en los MGP.

En la sección 5.2 exponemos dos propuestas o vistas de la TPC bajo el formalismo de listas *KBM2L* y mostramos la sintaxis y semántica de ambas. Tratamos de abordar diversos aspectos de la representación de la incertidumbre mediante TPC usando listas *KBM2L*: la representación de restricciones sintácticas y semánticas, la representación de intervalos de probabilidad (imprecisión), la discretización del intervalo de probabilidad [0,1] y la manipulación del esquema original.

En la sección 5.3 mostramos las aplicaciones que nos parecen más interesantes: educción, validación y extracción de conocimiento probabilístico. Por último, terminamos en las secciones 5.4 y 5.5 ilustrando todas las ideas con los problemas reales médicos de PGNHL e IctNeo, los mismos que se usaron en el capítulo 4.

5.1. Tablas de Probabilidad Condicionada

El objeto de este capítulo es una nueva aproximación para estructurar, implementar y analizar TPC (Fernández del Pozo y Bielza, 2002b). Estas tablas representan la incertidumbre en los MGP, como RB y DI. Una RB es un grafo dirigido acíclico cuyo nodos son los conceptos del problema y cuyos arcos representan influencia probabilística condicional, cuantificada por medio de una TPC asociada a cada nodo. La propagación de probabilidades nos permite realizar inferencias sobre algunos nodos dada alguna evidencia (Pearl, 1988). Un DI es un grafo dirigido acíclico, con nodos de decisión, con TPC igual que las RB asociadas a los nodos de azar y con

tablas de utilidades esperadas como principales estructuras de datos. La evaluación nos permite inferir las políticas óptimas del problema de decisión (TDO) y las distribuciones de probabilidad a posteriori (TPC) de las variables influyentes.

Tratar con estos modelos en IA y AD implica manejar TDO, TPC y tablas de utilidades esperadas muy grandes. Nos centraremos en las TPC. Tanto las RB como los DI necesitan asignar mediante educción (Henrion et al., 1991) y/o mediante aprendizaje (Buntine, 1996) TPC para representar el conocimiento, realizar inferencias y tomar decisiones. Además, son necesarios su almacenamiento y gestión de modo eficiente.

La TPC de un determinado nodo tiene una complejidad exponencial de acuerdo al tamaño del conjunto de atributos que son predecesores o padres. Esta complejidad puede reducirse eficientemente explotando la independencia entre variables, como hacen las representaciones de RB y DI y sus respectivos algoritmos de inferencia. Sin embargo, cuando una TPC muestra ciertas regularidades no capturadas por las estructuras de la RB o el DI, la complejidad de la TPC se puede reducir más aún. Estas regularidades son independencias que se verifican sólo en ciertos contextos, llamadas *independencias específicas del contexto* (Boutilier et al., 1996). La mayor parte de la literatura ha tratado esto proponiendo extensiones de las estructuras usuales que capturan estas independencias y soportan eficientemente los algoritmos. Véase (Cano et al., 2000), (Heckerman, 1990) y (Boutilier et al., 1996) dentro del ámbito de las RB y (Smith et al., 1993) y (Shenoy, 2000) en el marco de los DI. En general, la mayoría de las propuestas están basadas en estructuras de árbol.

El marco de las listas *KBM2L* se ajusta bien a las TPC, donde la representación de la incertidumbre se plantea sobre la combinatoria de todas las configuraciones y las coordenadas o índices del contexto condicionante o escenario. Las TPC grandes de alta dimensionalidad muestran regularidades y parámetros de valores dispersos (tal vez muchos ceros) en el espacio de representación de la tabla.

En este capítulo adaptamos el marco general de las *KBM2L* para trabajar con TPC mostrando diferentes opciones dependiendo de si los parámetros de probabilidad juegan el papel de variable respuesta o de atributo explicativo. En este último caso la lista representa la lógica de las proposiciones de la tabla. Conseguiremos dos logros importantes. Primero, representaciones compactas de las distribuciones de probabilidad y segundo, aportaremos conocimiento de las variables relevantes en los distintos contextos. De este modo permitiremos dar soporte a la educción y validación de las distribuciones, durante y después de la construcción del modelo, como parte del AS y diagnóstico del modelo.

Extendemos la notación para la TPC con el fin de destacar cuál es el atributo condicionado

de la tabla original y qué atributo representa el parámetro de probabilidad. Para el atributo condicionado usamos paréntesis (x), y para los parámetros de probabilidad en el papel de variable explicativa, corchetes $[p]$.

La codificación/decodificación de la TPC tiene presente la notación, junto a que tanto los atributos como sus valores están codificados como números naturales: los atributos $0, 1, 2, \ldots$ y los valores del atributo i: $v_{i,0}, v_{i,1}, v_{i,2}, \ldots$ Si la naturaleza de los atributos, su dominio, no es discreto debemos discretizarlo y acotarlo.

5.2. Tablas de Probabilidad Condicionada Usando *KBM2L*

En esta sección tratamos de resolver la representación del conocimiento de la TPC mediante listas *KBM2L*.

En esta tesis, consideramos una TPC como un conjunto de cláusulas proposicionales que miden la incertidumbre de una variable aleatoria X en un contexto $\bar{Y} = \bar{y}$ de un modelo de decisión. La notación usual es $P(X = x|\bar{Y} = \bar{y}) = p$, donde X es una variable aleatoria discreta, $\bar{Y}$ es un vector de variables aleatorias discretas y $p \in$[0, 1] es un número real que representa la probabilidad de $X = x$ dado $\bar{Y} = \bar{y}$, un contexto o estado del problema.

Nótese que el contenido de una TPC, es decir, los valores de la probabilidad, son datos de tipo real, pero como hemos dicho antes, la *KBM2L* es útil para atributos y respuesta discretos, con unos pocos valores de dominio. Por tanto, en las siguientes subsecciones, propondremos dos posibles maneras de representar esta clase de tabla usando el almacenamiento de la *KBM2L*: tomando como respuesta el valor p de probabilidad y tomando como respuesta el valor lógico {verdadero, falso} que indica la declaración probabilista de las entradas de la TPC. Cuando implementemos ambas, consideraremos una discretización del valor real de probabilidad para hacer más fácil la coalescencia de ítems, en la primera forma, y para evitar pesos infinitos W_i, en la segunda. Pensamos que la discretización de la probabilidad la puede establecer el experto como un parámetro de resolución útil para la construcción de la propia TPC y puede ser refinado posteriormente a través del AS.

A pesar de que nuestra representación trabaja con cualquier cardinalidad de los atributos, resulta conveniente usar sólo atributos binarios codificando con variables binarias los atributos multivariantes. Esto se lleva a cabo como en otros ámbitos (por ejemplo, la regresión logística, (Fahrmeir et al., 1994)) incrementando el número de variables, e introduciendo restricciones de consistencia en la *KBM2L* para controlar qué nuevas configuraciones (entradas de la TPC nueva) están permitidas y cómo reflejan las originales.

Por ejemplo, supongamos un atributo A cuyo dominio es {0,1,2}. Podemos **descomponer**

el dominio para obtener atributos binarios de varias formas. Una manera es: A_0 con dominio $\{0, 1 \vee 2\}$ y A_1 con dominio $\{1, 0 \vee 2\}$. Pero necesitamos introducir algunas restricciones de consistencia: si por ejemplo $A_0 = 0$ y $A_1 = 1$, entonces la lista debe almacenar *imposible* (restricción sintáctica), etiquetando la respuesta del caso como *coKB*, véase el Cuadro 5.1. Otra posibilidad es definir A_0 con dominio $\{0, \neg 0\}$, A_1 con dominio $\{1, \neg 1\}$, y A_2 con dominio $\{2, \neg 2\}$. La descomposición del dominio aumenta la dimensión de la representación, véase el Cuadro 5.1. La respuesta de los casos en una tabla cuyo esquema ha sido descompuesto toma el valor imposible en las combinaciones de atributos restringidas.

Cuadro 5.1: Codificaciones correctas de A

A_0	A_1	A	A_0	A_1	A_2	A	A_0	A_1	A_2	A
0	1	*coKB*	0	1	2	*coKB*	$\neg 0$	1	2	*coKB*
0	$0 \vee 2$	0	0	1	$\neg 2$	*coKB*	$\neg 0$	1	$\neg 2$	1
$1 \vee 2$	1	1	0	$\neg 1$	2	*coKB*	$\neg 0$	$\neg 1$	2	2
$1 \vee 2$	$0 \vee 2$	2	0	$\neg 1$	$\neg 2$	0	$\neg 0$	$\neg 1$	$\neg 2$	*coKB*

Algunas representaciones binarias del esquema original son *incompletas*, por ejemplo, A_0 con dominio $\{0 \vee 1, \neg(0 \vee 1)\}$ y A_1 con dominio $\{2, \neg 2\}$, y algunas, como por ejemplo A_0 con dominio $\{0 \vee 1, 1 \vee 2\}$ y A_1 con dominio $\{0 \vee 2, 1 \vee 2\}$ son *indefinidas*, véase el Cuadro 5.2. Estos esquemas no preservan la semántica original, es decir, no es posible recuperar el esquema original a partir de ellos. Incompleta es peor que indefinida pues hay valores que se pierden, por ejemplo, el 0 y 1 no pueden obtenerse combinando valores de A_0 y A_1 en el Cuadro 5.2. Indefinida significa que hay combinaciones que no determinan un valor del atributo original y eso implica una redundancia no deseable. Con *coKB*, no intentamos representar sólo restricciones sintácticas sino también configuraciones de dominio imposibles.

Cuadro 5.2: Codificaciones incorrectas de A

A_0	A_1	A (incompleta)	A_0	A_1	A (indefinida)
$0 \vee 1$	2	*coKB*	$0 \vee 1$	$0 \vee 2$	0
$0 \vee 1$	$\neg 2$	$0 \vee 1$	$0 \vee 1$	$1 \vee 2$	1
$\neg(0 \vee 1)$	2	2	$1 \vee 2$	$0 \vee 2$	2
$\neg(0 \vee 1)$	$\neg 2$	*coKB*	$1 \vee 2$	$1 \vee 2$	$1 \vee 2$

Nótese que un XB en el esquema binario artificial es equivalente a una permutación de los

valores del dominio del atributo original. Así, podemos conseguir más coalescencia en la lista a costa de tener un esquema de mayor complejidad.

5.2.1. La Dimensión Probabilidad como Respuesta

La forma más natural de gestionar una TPC es construir un índice con los atributos condicionantes $\bar{Y}$ y el atributo condicionado X, y construir una respuesta (continua) p, con el parámetro de probabilidad del MGP. Con la notación mencionada, definimos formalmente el ítem como $\langle(x)\bar{y}, p|$. Con dominios binarios {0,1}, la lista vacía es $\langle(1)\bar{1}$, *unKB*|. $(0)\bar{0}$ es *MinIndice* y $(1)\bar{1}$ es *MaxIndice*, con menor y mayor desplazamiento respectivamente. La etiqueta *unKB* significa que no hay conocimiento acerca de p en el correspondiente ítem de la lista.

Todas las dimensiones del índice tienen dominios finitos. Pero es difícil agrupar casos al tener una respuesta continua en la *KBM2L*. Una posibilidad es discretizar los valores de p. Así, si X es la ausencia o presencia de una patología, una discretización de p como *nula/baja/media/alta/máxima* probabilidad de padecer esta patología podría permitirnos compactar la TPC y obtener diagnósticos. Esto típicamente lo sugiere el experto humano de acuerdo al nivel deseado de discretización y queda completamente especificado con la declaración de la correspondencia de valores discretos (cualitativos) y valores del intervalo [0,1]. En este caso, perdemos precisión para permitir una representación más compacta y una construcción más sencilla de la TPC utilizando el conocimiento del experto. Otra posibilidad es discretizar los valores de p de forma dinámica, es decir, diferentes discretizaciones para diferentes valores de $\bar{y}$. (Cano et al., 2000) proponen la media de proposiciones similares. Una estrategia posible en este sentido sería establecer un tamaño máximo para la *KBM2L* e intentar suavizar el espectro en la base óptima, es decir, acotar el número de CMRS por desplazamiento ajustando una discretización si se supera un valor fijado. En este capítulo no discretizamos dinámicamente pues es bastante más complicado y creemos que el AS del modelo es lo que serviría para decidir cuál es la discretización más eficiente. Al cardinal del dominio de la discretización de p lo llamaremos K.

Por último, la discretización que estamos considerando se refiere únicamente a los valores de p para la definición de un dominio, ya sea de respuesta o de dimensión (véase la sección 5.2.2), y asumimos que las variables involucradas en el esquema de la TPC son todas discretas. Tal vez deberíamos decir *cuantificación de la probabilidad* si p es la respuesta de la lista pero seguimos hablando de discretización, pues en la lista, p puede ser una dimensión (en la segunda representación). La discretización de modelos de probabilidad, si hubiese variables continuas, es un problema diferente para el que hay varias propuestas, por ejemplo (Kozlov y Koller, 1997).

Interpretación de los ítems Supongamos que X es binaria. Aquí los ítems de la lista *KBM2L* son $\langle(x)\bar{y}, p|$ representando la dependencia probabilística entre los atributos de una manera explícita (posiblemente discretizada). Esta representación evita la expresión común de dos situaciones diferentes: la distribución uniforme (p es igual $\forall x$ en algunos contextos $\bar{y}$) y la ausencia de conocimiento. La uniformidad se trata con $((0)\bar{y}$,0.5) y $((1)\bar{y}$,0.5),si X es una variable binaria. Por otro lado, la falta de conocimiento se expresa con $((0)\bar{y}$,*unKB*) y $((1)\bar{y}$,*unKB*). En general, la complementariedad de $((0)\bar{y}, p)$ se expresa con $((1)\bar{y}, 1-p)$.

Sea $\langle\bar{y}_1(x)\bar{y}_2, p|$ un ítem genérico. Si x pertenece a la parte fija entonces existen dos ítems $\langle\bar{y}_1(0)\bar{y}_2, p|$, $\langle\bar{y}_1(1)\bar{y}_2, 1-p|$ o $\langle\bar{y}_1(1)\bar{y}_2, p|$, $\langle\bar{y}_1(0)\bar{y}_2, 1-p|$. Si x no pertenece a la parte fija, entonces $p = 0.5$ pues el ítem contiene todos los casos que constituyen la distribución de probabilidad condicionada $P(X = x|\bar{Y} = \bar{y}_1\bar{y}_2) = 0.5$ $(x = 0, 1)$ y la masa de probabilidad se distribuye uniformemente. El subconjunto de atributos que no está contenido en la parte fija de un ítem representa todos los escenarios donde la medida de incertidumbre es p.

No es difícil generalizar la interpretación anterior a un esquema con dominio de X no binario.

5.2.2. La Declaración de la Tabla como Respuesta

Esta otra definición de *KBM2L* tiene una respuesta discreta. La respuesta es codificada con tres valores $\{-1, 0, 1\}$. Con -1 representamos el valor *unKB*, 0 es *falso* y 1 es *verdadero*. Esta opción tiene la siguiente formalización de un ítem: $\langle[p](x)\bar{y}$,{*falso*,*verdadero*}$|$. El valor *falso* en $\ldots\langle[p_i](x_j)\bar{y}_k$,*verdadero*$|$ $\langle[p_l](x_m)\bar{y}_n$,*falso*$|\ldots$, significa que $P(X = x|\bar{Y} = \bar{y}) \neq p_l$ $\forall p_l, x, \bar{y}$ perteneciente al ítem. La lista vacía es $\langle[1](1)\bar{1}$,*unKB*$|$.

Ahora necesitamos una discretización del parámetro de probabilidad (es parte del índice) para calcular el desplazamiento de los ítems. Para definir el índice previo y posterior de un índice dado necesitamos cierto nivel de discretización K: 11, 101, 1001,... valores para p. Es decir, $[\lfloor p * K\rfloor - 1](1)\bar{1}$ es considerado como el índice inmediatamente anterior a $[\lfloor p * K\rfloor](0)\bar{0}$. Si fijamos la resolución de la discretización en 101, la lista vacía es $\langle[100](1)\bar{1}$,*unKB*$|$.

Pensamos que es interesante que la lista permita representar las TPC en las distintas situaciones: completas/incompletas y precisas/imprecisas. El valor *unKB* permite manejar el conocimiento parcial de las TPC. En cuanto a la imprecisión del valor de p en las entradas de las TPC, consideramos el modelo más sencillo: una distribución uniforme en un intervalo $[p_i, p_j]$.

De las dos opciones presentadas en el capítulo, afirmamos que ésta es la mejor representación de la lista *KBM2L* para TPC, cuando hay imprecisión en los valores de p, $P(X = x|\bar{Y} = \bar{y}) \in [p_i, p_j]$, que representamos mediante un ítem en la lista siguiente: $\ldots\langle(x)\bar{y}[p_i - 1]$,*falso*$|$ $\langle(x)\bar{y}[p_j]$,*verdadero*$|\ldots$, donde $[p_i, p_j]$ es un conjunto discreto.

Por otra parte, para que la primera definición presentada representase la imprecisión, debería considerar los intervalos de probabilidad como respuesta con todos los valores de la discretización pertenecientes a los intervalos, es decir, $\langle (x)\bar{y}, p_k \in \{p_i\}_i|$, donde realmente $\{p_i\}_i$ no es un intervalo, es un subconjunto de valores entre dos valores extremos. La coalescencia es prácticamente imposible bajo esta vista de la lista y el criterio de agregación de los casos por igualdad de la respuesta se cuestiona, pues pueden presentarse intersecciones parciales de los intervalos cuya interpretación es difícil. Además, aún cuando es muy intuitiva para valores precisos de p, puede producir una coalescencia de los casos baja, a menos que estemos interesados en una discretización muy burda en los umbrales o niveles de p (véase la sección 5.4).

Interpretación de los ítems Sean $\langle \bar{y}_1(x)\bar{y}_2[p]\bar{y}_3$,*verdadero*$|$ y $\langle \bar{y}_1[p]\bar{y}_2(x)\bar{y}_3$,*verdadero*$|$ dos ítems genéricos. Si p pertenece a la parte fija, entonces este ítem tiene una interpretación similar a la anterior representación, con p en la respuesta. Si p no pertenece a la parte fija, el ítem representa un intervalo de probabilidad. Cuando la respuesta es *falsa*, representa una restricción semántica sobre la TPC, mientras que *coKB* representa una restricción sintáctica, véanse los Cuadros 5.1 y 5.2 y la sección 5.5.

Los ítems de esta vista representan el conocimiento lógico probabilista: $\langle [p](x)\bar{y}$, {*falso, verdadero*}$| = \langle [p](x)\bar{y}, \{0,1\}|$. La respuesta es independiente del dominio, en el sentido de que siempre es un valor lógico {*falso, verdadero*}. Cualquier TPC puede transformarse en una lista *KBM2L* en ambas vistas pero la respuesta de la vista primera depende de la discretización.

Otras situaciones de interés se representan del modo siguiente. Los complementarios con $((0)\bar{y}[p], 1)$ y $((1)\bar{y}[1-p], 1)$, la uniformidad con $((0)\bar{y}[\frac{1}{2}], 1)$ y $((1)\bar{y}[\frac{1}{2}], 1)$, y la ignorancia o *unKB* con $((x)\bar{y}[p], unKB)$.

Ejemplo 5.1 La TPC de *Supervivencia post-PQR* (SP-PQR) del modelo PGNHL, véase el Cuadro 5.3, nos servirá para mostrar las dos vistas. La variable SP-PQR ({NO, SI}) está condicionada a las variables Perforación ({NO, SI}) y Hemorragia ({NO, SI}).

Codificamos NO con 0 y SI con 1. Consideramos varios valores de K. Especificamos la base $[x, \bar{y}]$ = [SP-PQR, Perforación, Hemorragia] para todas las listas que detallamos a continuación.

1. Discretizamos con $K = 3$, es decir, el 0 es probabilidad [0,0.30), 1 es [0.30,0.70) y 2 es [0.70,1].

   La lista *KBM2L* con la dimensión de probabilidad como respuesta es:

   $\langle (0)11, 0|$ $\langle (1)11, 2|$

Cuadro 5.3: TPC de (SP-PQR | Perforación, Hemorragia)

SP-PQR	NO	SP-PQR	SI
P(NO,NO,NO)	0.0	P(SI,NO,NO)	1.0
P(NO,NO,SI)	0.10	P(SI,NO,SI)	0.90
P(NO,SI,NO)	0.20	P(SI,SI,NO)	0.80
P(NO,SI,SI)	0.25	P(SI,SI,SI)	0.75

2. Discretizamos con $K = 11$, es decir, el 0 es probabilidad [0,0.1), 1 es [0.10,0.2), 2 es [0.2,0.3) ... 9 es [0.90,1) y 10 es 1.

   La lista *KBM2L* con la dimensión de probabilidad como respuesta es:

   $\langle(0)00, 0|$ $\langle(0)01, 1|$ $\langle(0)11, 2|$ $\langle(1)00, 10|$
   $\langle(1)01, 9|$ $\langle(1)10, 8|$ $\langle(1)11, 7|$

3. Discretizamos con $K = 21$, es decir, el 0 es probabilidad [0,0.05), 1 es [0.05,0.1), ... 19 es [0.95,1) y 20 es 1.

   La lista *KBM2L* con la dimensión de probabilidad como respuesta es:

   $\langle(0)00, 0|$ $\langle(0)01, 2|$ $\langle(0)10, 4|$ $\langle(0)11, 5|$
   $\langle(1)00, 20|$ $\langle(1)01, 18|$ $\langle(1)10, 16|$ $\langle(1)11, 15|$

4. Una lista *KBM2L* con la declaración de la tabla como respuesta y discretizando análogamente con $K = 3$ es:

   $\langle[0](0)11,$ *verdadero*$|$ $\langle[2](0)11,$ *falso*$|$ $\langle[2](1)11,$ *verdadero*$|$

5. Una lista *KBM2L* con la declaración de la tabla como respuesta y discretizando con $K = 11$ es:

   $\langle[0](0)00,$ *verdadero*$|$ $\langle[1](0)00,$ *falso*$|$ $\langle[1](0)01,$ *verdadero*$|$ $\langle[2](0)01,$ *falso*$|$
   $\langle[2](0)11,$ *verdadero*$|$ $\langle[7](1)10,$ *falso*$|$ $\langle[7](1)11,$ *verdadero*$|$ $\langle[8](1)01,$ *falso*$|$
   $\langle[8](1)10,$ *verdadero*$|$ $\langle[9](1)00,$ *falso*$|$ $\langle[9](1)01,$ *verdadero*$|$ $\langle[10](0)11,$ *falso*$|$
   $\langle[10](1)00,$ *verdadero*$|$ $\langle[10](1)11,$ *falso*$|$

6. Una lista *KBM2L* con la declaración de la tabla como respuesta y discretizando con $K = 21$ es:

⟨[0](0)00, *verdadero*| ⟨[2](0)00, *falso*| ⟨[2](0)01, *verdadero*| ⟨[4](0)01, *falso*|
⟨[4](0)10, *verdadero*| ⟨[5](0)10, *falso*| ⟨[5](0)11, *verdadero*| ⟨[15](1)10, *falso*|
⟨[15](1)11, *verdadero*| ⟨[16](1)01, *falso*| ⟨[16](1)10, *verdadero*| ⟨[18](1)00, *falso*|
⟨[18](1)01, *verdadero*| ⟨[20](0)11, *falso*| ⟨[20](1)00, *verdadero*| ⟨[20](1)11, *falso*|

□

5.3. Aplicaciones en Tablas de Probabilidad Condicionada

En esta sección mostramos la construcción y/o educción, validación y extracción de conocimiento en TPC usando listas *KBM2L*.

5.3.1. Educción de Tablas de Probabilidad Condicionada Usando *KBM2L*

La construcción de MGP requiere la cuantificación de las relaciones de influencia probabilista que recoge la estructura de la red. Si todas las variables son discretas, la cuantificación consiste en asignar las TPC correspondientes a cada relación. Hay tres formas de obtener las TPC, todas compatibles. La primera, mediante sesiones de educción entre el analista y los expertos. En (Kadane y Wolfson, 1997) se recoge una revisión de distintas experiencias de educción y en (O'Hagan, 1998) algunos ejemplos prácticos. A los expertos se les interroga mediante entrevistas estructuradas con alguna metodología específica acerca de los parámetros del modelo que constituyen las TPC. En general, hablamos de educción en un problema durante la fase de adquisición de conocimiento (si utilizamos un DI se trata de educir el marco conceptual, la estructura, las probabilidades, las utilidades,...) de un sistema basado en el conocimiento y de los diversos modelos que lo componen. Especialmente cercanos nos resultan los trabajos (Ríos-Insua et al., 2000) y (Gómez, 2002) relativos al SAD *IctNeo* y (Lucas, 1996; Lucas et al., 1998; Lucas et al., 2000) relativos a diversos problemas médicos donde es particularmente difícil asignar las TPC. Por ello, también se pueden utilizar modelos canónicos, por ejemplo puertas OR, (Díez, 1993) para simplificar las sesiones de educción, si el problema se ajusta a las hipótesis de dichos modelos.

La segunda forma de obtener las TPC es mediante conjuntos de datos para estimar las probabilidades, por ejemplo a partir de las frecuencias relativas. En general, se trata de utilizar técnicas de AA y Estadística para obtener o refinar RB utilizando datos. Una revisión sobre el aprendizaje de redes probabilistas puede consultarse en (Buntine, 1996) y concretamente para el algoritmo K2, mencionado en la sección 4.3 de (Cooper y Herskovits, 1992).

La tercera forma hace uso de la literatura disponible sobre el problema. La literatura aporta mucha información probabilística pero generalmente no es adecuada para un problema particular y debe ser revisada por el experto y contrastada con datos si es posible (Lucas, 1996).

Estos métodos pueden combinarse en el proceso de educción de un modelo. Por ejemplo, es habitual asignar las dependencias mediante un experto y las probabilidades mediante aprendizaje (Lacave, 2003).

Proponemos la lista *KBM2L* y su espectro como herramientas útiles y complementarias a otros métodos para realizar la asignación de probabilidades de parámetros precisos y gestionar la imprecisión de los parámetros de probabilidad, por ejemplo, como intervalos de probabilidad. Los expertos pueden declarar reglas generales y restricciones respecto de la TPC y analizar la lista en diferentes *Bases*.

Seguidamente, mostramos algunos ejemplos de cómo pueden representarse diferentes declaraciones del experto. Especificamos la *Base* $[x, \bar{y}]$ o $[p, x, \bar{y}]$ como un subíndice.

Ejemplo 5.2 El experto puede decir *Cuando* Y_j *es* 0, X *es* 1 *con probabilidad 0.95*, cuya expresión es:

$$((1), \ldots, 0_j, \ldots, 0.95) \text{ (un caso genérico preciso).}$$

En la *Base* $[X, Y_j, \ldots]$ esto induce:

$$\ldots, \langle (0)1_j, 1, 1, \ldots, 1, *|, \langle (1)0_j, 1, 1, \ldots, 0.95|, \ldots$$

También dice X *puede ser* 0 *ó* 1 *pero* 2 *no es posible si* Y_j *es* 1, cuya expresión es:

$$((2), \ldots, 1_j, \ldots, \textit{coKB}) \text{ (un caso restringido).}$$

Y también, el experto puede decir *Pienso que la probabilidad en este tipo de situaciones,* $x\bar{y}$*, involucra una probabilidad próxima a 0.25, pero no lo sé exactamente* (un caso impreciso). Siguiendo la primera alternativa de representación esto supone tener varios casos, por ejemplo $((x)\bar{y}, 0.24)_{[x,\bar{y}]}$, $((x)\bar{y}, 0.25)_{[x,\bar{y}]}$, $((x)\bar{y}, 0.26)_{[x,\bar{y}]}$, y los casos complementarios correspondientes. Pero esto define una respuesta multivaluada $(0.24 \vee 0.25 \vee 0.26)$ o bien un intervalo de probabilidades imprecisas ([0.24,0.26]). Pensamos que esta representación no es adecuada porque produce una discretización irregular con intervalos de amplitud no homogénea y, sobre todo, un dominio de respuesta inadecuado con valores de respuesta no excluyentes. Por la primera consideración (discretización irregular) se hace más difícil la interpretación del espectro de la lista, pues parece que le damos más importancia a cierta magnitud de la probabilidad (que tratamos con mayor resolución). En cuanto a la segunda (dominio de respuesta inadecuado), queda indeterminada la

coalescencia de casos que se basa en la igualdad de respuestas adyacentes, que ahora pueden ser iguales, parecidas o estar en inclusión. Este aspecto ya lo hemos mencionado en la sección 5.2.2.

La solución es, siguiendo la segunda alternativa de representación, construir un ítem como $\ldots, \langle(x)\bar{y}[0.23], \mathit{falso}|_{[x,\bar{y},p]}, \langle(x)\bar{y}[0.26], \mathit{verdadero}|_{[x,\bar{y},p]}, \langle(x)\bar{y}[0.27], \mathit{falso}|_{[x,\bar{y},p]}, \ldots$ Entonces, podríamos insertarlo en la correspondiente lista, que en general está expresada en una *Base* diferente y buscaríamos la lista óptima, con menor espacio de almacenamiento. □

Las tareas para realizar la educción de las TPC mediante listas *KBM2L* son: codificación de las declaraciones del experto y las restricciones, representación del espectro y visualización de los ítems, sus límites, parte fija (relevante),... (Fernández del Pozo et al., 2001). La lista ayuda a recoger todo el conocimiento experto de modo consistente y da pistas acerca de cuál es la información necesaria para construir la TPC. La lista óptima nos permite sintetizar la información en un conjunto pequeño de preguntas relevantes y breves para mejorar el modelo de probabilidad asociado a la TPC. El analista puede formular preguntas al experto sobre regiones del espacio de representación de la lista *KBM2L* con falta de conocimiento y comprobar posibles inconsistencias. Se plantean varias revisiones y optimizaciones de la lista hasta que la TPC está completa. En resumen, pensamos que la lista permite articular un proceso de validación, del conocimiento parcial educido hasta el momento, que acompaña al proceso de educción para hacerlo más eficiente y robusto. En la sección 5.5 mostramos un ejemplo de educción de una TPC para el SAD *IctNeo*.

5.3.2. Validación de Tablas de Probabilidad Condicionada Usando *KBM2L*

Un SBC, en particular un SAD, debe ser verificado y validado para garantizar su competencia en las tareas que tiene que realizar (Preece y Shinghal, 1994). La verificación trata de garantizar que el sistema no tiene errores y la validación trata de garantizar que el sistema resuelve el problema que motivó su diseño y construcción (Bohem, 1981). El AD y los MGP facilitan la verificación gracias al marco normativo que proporcionan para la representación del problema y la evaluación del modelo. La validación es más compleja pues se trata de un aspecto semántico. En problemas reales es imposible ensayar exhaustivamente el sistema y debemos seguir alguna estrategia y metodología para asegurarnos de que el sistema está validado con un grado alto de confianza pero con un coste razonable.

La idea que proponemos es estudiar la BC para generar, más o menos automáticamente, preguntas a un experto o hipótesis que podamos contrastar con un conjunto de datos y análisis estadísticos. Las preguntas deben ser eficientes, es decir, pocas pero muy significativas.

En este capítulo, la BC es el conjunto de TPC que hay en el DI. La lista *KBML* resume un conjunto de parámetros muy grande de las TPC. Proponemos ahora que esta estructura sirva para que el experto valide la TPC o que lo haga una muestra de datos sobre la lista correspondiente. Presentamos dos tipos de estrategias: primero, intentamos estudiar ítems grandes y el papel de los atributos (su pertenencia a la parte fija o variable del ítem); segundo, intentamos analizar ítems pequeños con índices similares y respuestas distintas.

Los ítems grandes involucran pocos atributos fijos y por tanto, las preguntas formuladas al experto pueden ser simples (al menos sintácticamente). Los ítems pequeños involucran muchos atributos fijos y por consiguiente, mediante la comparación de dos de ellos con sólo pequeñas diferencias en su parte fija, las preguntas formuladas podrían derivarse fácilmente a partir de esas diferencias. Por tanto, el análisis de la lista *KBM2L* y de las explicaciones de los ítems nos permite obtener un conjunto de preguntas para realizar la validación. Primero, realizamos una jerarquía por importancia de los atributos (peso) y alcance de la pregunta (número de casos) y después seleccionamos las que a priori tengan según el experto más impacto en la tarea de encontrar fallos.

El proceso de validación es similar al proceso de educción, pero en la validación suponemos que la TPC está dada e intentamos extraer las reglas generales para los expertos desde la lista óptima. Ellos aceptarán o no los resultados. La Educción y la Validación son procedimientos recíprocos en la construcción de los sistemas basados en conocimiento. En la sección 5.4 mostramos un ejemplo de validación de una TPC para el SAD *PGNHL*.

5.3.3. Extracción de Conocimiento de una Tabla de Probabilidad Condicionada Usando *KBM2L*

En grandes TPC, la lista *KBM2L* es útil para descubrir patrones generales, los principales componentes de las relaciones probabilísticas. Podemos explorar varias reglas o descripciones acerca del problema y mostrar su ajuste con la TPC que fue construida usando datos o evaluando un modelo. Concretamente, la evaluación de un DI produce TPC muy grandes, distribuciones a posteriori, que pueden ser analizadas con nuestra herramienta para explicación y AS. En TPC de muy alta dimensionalidad podemos ver más claramente las ventajas de esta técnica que trataremos de ilustrar en la tesis a través de un ejemplo sencillo en la sección 5.4.

5.3.4. Ejemplos de Representación de una Tabla de Probabilidad Condicionada Usando *KBM2L*

Unos ejemplos nos servirán para ilustrar los detalles de las dos técnicas de representación propuestas en el problema de asignación de una TPC usando *KBM2L*, véase el Cuadro 5.4 como resumen.

Cuadro 5.4: Resumen de las representaciones de TPC usando estructuras *KBM2L* con dominios binarios y una discretización de 101

respuesta	$\rightarrow p$	respuesta	$\rightarrow$ {*falso,verdadero*}
ítems	lista vacía	ítems	lista vacía
$\langle(x)\bar{y}, p\vert$	$\langle(1)\bar{1}$,*unKB*$\vert$	$\langle[p](x)\bar{y}, \{0,1\}\vert$	$\langle[100](1)\bar{1}$,*unKB*$\vert$

Ejemplo 5.3 Supongamos la siguiente TPC simple para X dado Y: $P(x_0|y_0) = a$, $P(x_1|y_0) = 1-a$, $P(x_0|y_1) = b$, $P(x_1|y_1) = 1-b$, con variables o atributos: $X : \{x_0, x_1\}$; $Y : \{y_0, y_1\}$. Usamos los valores ordinales $\{0, 1\}$ del dominio en lugar de $\{x_0, x_1\}$ y $\{y_0, y_1\}$ ($x_0 = 0$, $x_1 = 1$, $y_0 = 0$, $y_1 = 1$).

Por motivos de claridad en la exposición, asumimos que $0 < a < b < 0.5$, pero cualesquiera $a, b \in [0, 1]$ serán posibles.

La asignación de probabilidades utilizando una lista *KBM2L* pasa por considerar la discretización del intervalo $[0, 1]$. En el primer tipo de representación favorecemos la compactación de la lista. Por otra parte, en el segundo tipo, es necesaria para que el dominio de la dimensión p sea finito. Si la dimensión p es la de mayor peso, no es preciso tener un dominio finito (δ_0 no interviene en el cálculo del desplazamiento, véase la expresión 2.1), pero entonces imponemos una restricción al conjunto de bases que puede servirnos para configurar la lista.

La discretización de la probabilidad con $K = 101$ valores implica que la codificación de a, b, $1-a$, $1-b$ es como sigue: $A = \lfloor a * K \rfloor$, $B = \lfloor b * K \rfloor$, $\bar{B} = \lfloor (1-b) * K \rfloor$, $\bar{A} = \lfloor (1-a) * K \rfloor$.

El proceso de asignación puede realizarse mediante valores exactos, pero es más realista que las afirmaciones del experto se refieran a intervalos o proposiciones imprecisas como en el Ejemplo 5.2. Por tanto, tenemos en cuenta dos posibilidades:

1. Los parámetros son precisos, a =0.15 y b =0.45. Entonces, $A = 15$, $B = 45$, $\bar{B} = 55$, $\bar{A} = 85$.

2. Los parámetros son desconocidos, pero conocemos una cota inferior y superior y entonces $a \in [0.12, 0.16]$ y $b \in [0.40, 0.45]$. Así, $A = \{12, ..., 16\}$ y $B = \{40, ..., 45\}$.

Vamos a representar ambas posibilidades con las dos técnicas. La lista *KBM2L* discretizada, con la dimensión de probabilidad como respuesta, es $(\langle(0)0, A|\ \langle(0)1, B|\ \langle(1)0, \bar{A}|\ \langle(1)1, \bar{B}|)_{[x,y]}$, es decir,

$$(\langle(0)0, 15|\ \langle(0)1, 45|\ \langle(1)0, 85|\ \langle(1)1, 55|)_{[x,y]}.$$

Todos los ítems tienen tamaño igual a 1 y la *Base* es la mejor.

La segunda posibilidad es

$$(\langle(0)0, \{12, \ldots, 16\}|\ \langle(0)1, \{40, \ldots, 45\}|\ \langle(1)0, \{84, \ldots, 88\}|\ \langle(1)1, \{55, \ldots, 60\}|)_{[x,y]}.$$

La respuesta está definida sobre los conjuntos de valores que se derivan de los intervalos de los parámetros. Se trata de una representación compacta de las (30) posibles TPC que resultan de considerar asignaciones a los parámetros compatibles con los intervalos.

A continuación mostramos la lista *KBM2L* discretizada, con la declaración de la tabla como respuesta. Recordemos la notación y denotemos el tamaño de los ítems, su número de casos, como un exponente:

$$\langle[probabilidad](atributo\ condicional)atributos\ de\ \ contexto,\ \{-1, 0, 1\}|^{num\ casos\ de\ un\ item}$$

La lista *KBM2L* simple para X, es decir, el caso preciso es:

$$\begin{array}{llll}
(\langle[14](1)1, 0|^{60} & \langle[15](0)0, 1|^{1} & \langle[45](0)0, 0|^{120} & \langle[45](0)1, 1|^{1} \\
\langle[55](1)0, 0|^{41} & \langle[55](1)1, 1|^{1} & \langle[85](0)1, 0|^{118} & \langle[85](1)0, 1|^{1} \\
\langle[100](1)1, 0|^{61})_{[p,x,y]} & & &
\end{array}$$

La lista *KBM2L* compleja e imprecisa es:

$$\begin{array}{llll}
(\langle[11](1)1, 0|^{48} & \langle[12](0)0, 1|^{1} & \langle[12](1)1, 0|^{3} & \langle[13](0)0, 1|^{1} \\
\langle[13](1)1, 0|^{3} & \langle[14](0)0, 1|^{1} & \langle[14](1)1, 0|^{3} & \langle[15](0)0, 1|^{1} \\
\langle[15](1)1, 0|^{3} & \langle[16](0)0, 1|^{1} & \langle[40](0)0, 0|^{96} & \langle[40](0)1, 1|^{1} \\
\langle[41](0)0, 0|^{3} & \langle[41](0)1, 1|^{1} & \langle[42](0)0, 0|^{3} & \langle[42](0)1, 1|^{1} \\
\langle[43](0)0, 0|^{3} & \langle[43](0)1, 1|^{1} & \langle[44](0)0, 0|^{3} & \langle[44](0)1, 1|^{1} \\
\langle[45](0)0, 0|^{3} & \langle[45](0)1, 1|^{1} & \langle[55](1)0, 0|^{41} & \langle[55](1)1, 1|^{1} \\
\langle[56](1)0, 0|^{3} & \langle[56](1)1, 1|^{1} & \langle[57](1)0, 0|^{3} & \langle[57](1)1, 1|^{1} \\
\langle[58](1)0, 0|^{3} & \langle[58](1)1, 1|^{1} & \langle[59](1)0, 0|^{3} & \langle[59](1)1, 1|^{1} \\
\langle[60](1)0, 0|^{3} & \langle[60](1)1, 1|^{1} & \langle[84](0)1, 0|^{94} & \langle[84](1)0, 1|^{1} \\
\langle[85](0)1, 0|^{3} & \langle[85](1)0, 1|^{1} & \langle[86](0)1, 0|^{3} & \langle[86](1)0, 1|^{1} \\
\langle[87](0)1, 0|^{3} & \langle[87](1)0, 1|^{1} & \langle[88](0)1, 0|^{3} & \langle[88](1)0, 1|^{1} \\
\langle[100](1)1, 0|^{49})_{[p,x,y]} & & &
\end{array}$$

El $XBase$ $[p, x, y] \rightarrow [y, x, p]$ para esta lista la reduce de 45 a 9 ítems:

$$(\langle 0(0)[11], 0|^{12} \quad \langle 0(0)[16], 1|^{5} \quad \langle 0(1)[83], 0|^{168} \quad \langle 0(1)[88], 1|^{5}$$
$$\langle 1(0)[39], 0|^{52} \quad \langle 1(0)[45], 1|^{6} \quad \langle 1(1)[54], 0|^{110} \quad \langle 1(1)[60], 1|^{6}$$
$$\langle 1(1)[100], 0|^{40})_{[y,x,p]}$$

La Figura 5.1 muestra el espectro de ambas listas.

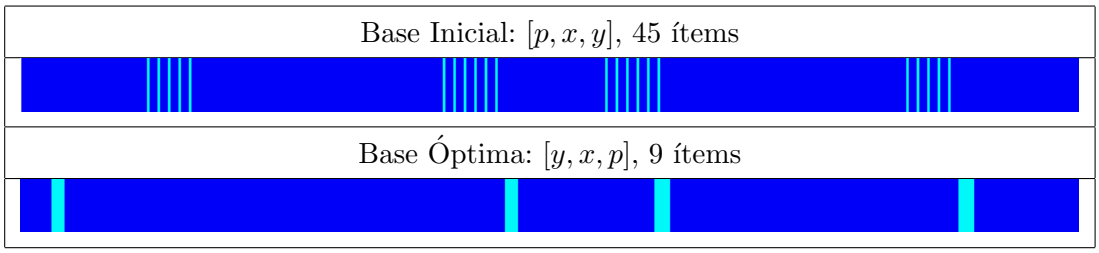

Figura 5.1: Espectros de las listas *KBM2L*

Debido a que $|A| = 5$ y $|B| = 6$, de forma análoga a la representación mostrada anteriormente, esta estructura representa $5 \cdot 6 = 30$ TPC como la que tiene a =0.14, b =0.40, o a =0.13, b =0.44. Así, por ejemplo, almacenamos implícitamente 0.83< $P(x_1|y_0)$ ≤0.88 en $\langle 0(1)[88], 1|$ y $P(x_0|y_0)$ >0.11 en $\langle 0(0)[11], 0|^{12}_{[y,x,p]}$. Podríamos incrementar la resolución fácilmente porque p tiene peso igual a 1 y la longitud de la lista no cambia.

Podemos ver también, por ejemplo, que el tercer ítem $\langle 0(1)[83], 0|^{168}_{[y,x,p]}$ representa con *falso*, 84 casos con $Y = y_0$, $X = x_0$ y $17 \leq p \leq 100$ y 84 casos con $Y = y_0$, $X = x_1$ y $0 \leq p \leq 83$. Luego, la $P(X = x_0|Y = y_0)$ es menor que 0.17 y la $P(X = x_1|Y = y_0)$ es mayor que 0.83. En este ejemplo, la *Base* es muy sencilla y no se aprecia la ventaja de la lista óptima que muestra una información similar a la TPC. La *Base* $[x, y, p]$ es óptima también y sin embargo, las *Bases* $[y, p, x]$, $[p, y, x]$ y $[x, p, y]$ tienen configuraciones de igual tamaño que la *Base* inicial.

La *Base* óptima ofrece la representación más eficiente, con la longitud menor, de la TPC. Los ítems de respuesta *verdadero* aportan conocimiento relativo a la imprecisión de la TPC y en general, aunque no necesariamente, tienen a x, a todo el contexto $\bar{y}$ y a p como atributos fijos y ADR. La respuesta binaria nos permite explotar la explicación de los ítems de respuesta *falso* de forma dual o complementaria. La explicación de dichos ítems es más sencilla y por tanto las proposiciones formuladas en sentido negativo tienen menos atributos. □

Las dos secciones siguientes muestran la aplicación de las representaciones propuestas en este capítulo a los SAD PGNHL e IctNeo.

5.4. Aplicación al Sistema PGNHL

Esta sección muestra, con ayuda del SAD PGNHL, (Lucas et al., 1998) la extracción de conocimiento y validación de una de sus TPC. En este modelo tenemos dos TPC grandes para las variables *Resultados Inmediatos* (RI, 1920 parámetros de probabilidad) y *Resultados a los Cinco Años* (R5A, 3840 parámetros). Hemos escogido sin embargo la TPC de *Enfermedad Abultada* (240 parámetros) que nos permite mostrar la técnica sin complicarnos en un ejemplo excesivamente grande.

La variable Enfermedad Abultada (EA) toma dos valores (SI, NO) y está condicionada por las variables:

1. *EDAD* (E) que toma 12 valores (10a19, 20a29, 30a39, 40a44, 45a49, 50a54, 55a59, 60a64, 65a69, 70a79, 80a89, masde90);

2. *Clasificación Histológica* (CH) que toma 2 valores (GradoBajo, GradoAlto) y

3. Fase Clínica (FC) que toma 5 valores (I, II1, II2, III, IV).

La TPC de la variable EA se muestra en los Cuadros 5.5 y 5.6. Cada elemento de la TPC es de la forma $P(EA, E, CH, FC) = p$, donde se han codificado con números enteros las modalidades de los atributos conforme al orden mostrado anteriormente. No es una tabla excesivamente grande, pero sí se observa cierta dificultad en la descripción de su contenido del que pretendemos extraer conocimiento en forma de reglas y restricciones.

Vamos a utilizar la dimensión de probabilidad como respuesta, véase la sección 5.2.1. La lista vacía en la *Base* $[EA, E, CH, FC]$ es: $\langle((1), 11, 1, 4), -1|$. La TPC tiene 182 ítems para representar los 240 casos cuya respuesta es p y está discretizada con $K = 101$, véanse el Cuadro 5.7 y la Figura 5.2.

Figura 5.2: Espectro de la lista *KBM2L* inicial para la TPC de EA con probabilidad como respuesta

En la *Base* $[EA, CH, FC, E]$ la lista tiene 130 ítems, véanse el Cuadro 5.8 y la Figura 5.3.

Vamos a analizar algunos ítems y sus explicaciones basadas en la parte fija y ADR. Los ítems 3 ($p = 10$), 77 ($p = 50$) y 68 ($p = 90$) constan de 4 casos cada uno. Significa que las declaraciones asociadas a cada ítem presentan un nivel de incertidumbre similar. Examinamos el ítem 3 y observamos que EA=SI presenta una probabilidad de 0.10 para edades comprendidas entre 40

Cuadro 5.5: TPC de la variable EA condicionada a E, CH y FC (I)

EA=SI	EA=NO	EA=SI	EA=NO
P(1,1,1,1) =0.40	P(2,1,1,1) =0.60	P(1,1,1,2) =0.30	P(2,1,1,2) =0.70
P(1,1,1,3) =0.35	P(2,1,1,3) =0.65	P(1,1,1,4) =0.25	P(2,1,1,4) =0.75
P(1,1,1,5) =0.25	P(2,1,1,5) =0.75	P(1,1,2,1) =0.70	P(2,1,2,1) =0.30
P(1,1,2,2) =0.70	P(2,1,2,2) =0.30	P(1,1,2,3) =0.75	P(2,1,2,3) =0.25
P(1,1,2,4) =0.60	P(2,1,2,4) =0.40	P(1,1,2,5) =0.60	P(2,1,2,5) =0.40
P(1,2,1,1) =0.30	P(2,2,1,1) =0.70	P(1,2,1,2) =0.30	P(2,2,1,2) =0.70
P(1,2,1,3) =0.35	P(2,2,1,3) =0.65	P(1,2,1,4) =0.15	P(2,2,1,4) =0.85
P(1,2,1,5) =0.15	P(2,2,1,5) =0.85	P(1,2,2,1) =0.50	P(2,2,2,1) =0.50
P(1,2,2,2) =0.55	P(2,2,2,2) =0.45	P(1,2,2,3) =0.75	P(2,2,2,3) =0.25
P(1,2,2,4) =0.60	P(2,2,2,4) =0.40	P(1,2,2,5) =0.60	P(2,2,2,5) =0.40
P(1,3,1,1) =0.20	P(2,3,1,1) =0.80	P(1,3,1,2) =0.25	P(2,3,1,2) =0.75
P(1,3,1,3) =0.50	P(2,3,1,3) =0.50	P(1,3,1,4) =0.05	P(2,3,1,4) =0.95
P(1,3,1,5) =0.05	P(2,3,1,5) =0.95	P(1,3,2,1) =0.35	P(2,3,2,1) =0.65
P(1,3,2,2) =0.35	P(2,3,2,2) =0.65	P(1,3,2,3) =0.60	P(2,3,2,3) =0.40
P(1,3,2,4) =0.50	P(2,3,2,4) =0.50	P(1,3,2,5) =0.50	P(2,3,2,5) =0.50
P(1,4,1,1) =0.10	P(2,4,1,1) =0.90	P(1,4,1,2) =0.20	P(2,4,1,2) =0.80
P(1,4,1,3) =0.50	P(2,4,1,3) =0.50	P(1,4,1,4) =0.05	P(2,4,1,4) =0.95
P(1,4,1,5) =0.05	P(2,4,1,5) =0.95	P(1,4,2,1) =0.25	P(2,4,2,1) =0.75
P(1,4,2,2) =0.30	P(2,4,2,2) =0.70	P(1,4,2,3) =0.55	P(2,4,2,3) =0.45
P(1,4,2,4) =0.40	P(2,4,2,4) =0.60	P(1,4,2,5) =0.40	P(2,4,2,5) =0.60
P(1,5,1,1) =0.10	P(2,5,1,1) =0.90	P(1,5,1,2) =0.20	P(2,5,1,2) =0.80
P(1,5,1,3) =0.50	P(2,5,1,3) =0.50	P(1,5,1,4) =0.05	P(2,5,1,4) =0.95
P(1,5,1,5) =0.05	P(2,5,1,5) =0.95	P(1,5,2,1) =0.20	P(2,5,2,1) =0.80
P(1,5,2,2) =0.25	P(2,5,2,2) =0.75	P(1,5,2,3) =0.55	P(2,5,2,3) =0.45
P(1,5,2,4) =0.40	P(2,5,2,4) =0.60	P(1,5,2,5) =0.40	P(2,5,2,5) =0.60
P(1,6,1,1) =0.10	P(2,6,1,1) =0.90	P(1,6,1,2) =0.20	P(2,6,1,2) =0.80
P(1,6,1,3) =0.50	P(2,6,1,3) =0.50	P(1,6,1,4) =0.05	P(2,6,1,4) =0.95
P(1,6,1,5) =0.05	P(2,6,1,5) =0.95	P(1,6,2,1) =0.20	P(2,6,2,1) =0.80
P(1,6,2,2) =0.25	P(2,6,2,2) =0.75	P(1,6,2,3) =0.50	P(2,6,2,3) =0.50
P(1,6,2,4) =0.40	P(2,6,2,4) =0.60	P(1,6,2,5) =0.40	P(2,6,2,5) =0.60

Figura 5.3: Espectro de la lista *KBM2L* optimizada para la TPC de EA con probabilidad como respuesta

y 59 años, que se incrementa en pacientes más jovenes y más mayores que están representados en los ítems adyacentes, respectivamente:

Item: 2	Indice: [0, 0, 0, 2] Respuesta: $p = 20$
	EA: SI, CH: GradoBajo, FC: I, E: 30a39
Item: 3	Indice: [0, 0, 0, 6], 3 Atributos fijos, 4 Casos
	EA: SI, CH: GradoBajo, FC: I, E: 55a59
Item: 4	Indice: [0, 0, 0, 7] Respuesta: $p = 20$
	EA: SI, CH: GradoBajo, FC: I, E: 60a64

En el ítem 77 observamos que EA = NO presenta una probabilidad de 0.50 para CH = GradoBajo, FC = II2 y edades comprendidas entre 30 y 54 años, que se incrementa en pacientes

Cuadro 5.6: TPC de la variable EA condicionada a E, CH y FC (II)

EA=SI	EA=NO	EA=SI	EA=NO
P(1,7,1,1) =0.10	P(2,7,1,1) =0.90	P(1,7,1,2) =0.20	P(2,7,1,2) =0.80
P(1,7,1,3) =0.55	P(2,7,1,3) =0.45	P(1,7,1,4) =0.05	P(2,7,1,4) =0.95
P(1,7,1,5) =0.05	P(2,7,1,5) =0.95	P(1,7,2,1) =0.20	P(2,7,2,1) =0.80
P(1,7,2,2) =0.25	P(2,7,2,2) =0.75	P(1,7,2,3) =0.50	P(2,7,2,3) =0.50
P(1,7,2,4) =0.40	P(2,7,2,4) =0.60	P(1,7,2,5) =0.40	P(2,7,2,5) =0.60
P(1,8,1,1) =0.20	P(2,8,1,1) =0.80	P(1,8,1,2) =0.25	P(2,8,1,2) =0.75
P(1,8,1,3) =0.55	P(2,8,1,3) =0.45	P(1,8,1,4) =0.10	P(2,8,1,4) =0.90
P(1,8,1,5) =0.10	P(2,8,1,5) =0.90	P(1,8,2,1) =0.25	P(2,8,2,1) =0.75
P(1,8,2,2) =0.30	P(2,8,2,2) =0.70	P(1,8,2,3) =0.55	P(2,8,2,3) =0.45
P(1,8,2,4) =0.50	P(2,8,2,4) =0.50	P(1,8,2,5) =0.50	P(2,8,2,5) =0.50
P(1,9,1,1) =0.25	P(2,9,1,1) =0.75	P(1,9,1,2) =0.30	P(2,9,1,2) =0.70
P(1,9,1,3) =0.60	P(2,9,1,3) =0.40	P(1,9,1,4) =0.20	P(2,9,1,4) =0.80
P(1,9,1,5) =0.20	P(2,9,1,5) =0.80	P(1,9,2,1) =0.25	P(2,9,2,1) =0.75
P(1,9,2,2) =0.35	P(2,9,2,2) =0.65	P(1,9,2,3) =0.55	P(2,9,2,3) =0.45
P(1,9,2,4) =0.60	P(2,9,2,4) =0.40	P(1,9,2,5) =0.60	P(2,9,2,5) =0.40
P(1,10,1,1) =0.30	P(2,10,1,1) =0.70	P(1,10,1,2) =0.35	P(2,10,1,2) =0.65
P(1,10,1,3) =0.65	P(2,10,1,3) =0.35	P(1,10,1,4) =0.30	P(2,10,1,4) =0.70
P(1,10,1,5) =0.30	P(2,10,1,5) =0.70	P(1,10,2,1) =0.25	P(2,10,2,1) =0.75
P(1,10,2,2) =0.35	P(2,10,2,2) =0.65	P(1,10,2,3) =0.65	P(2,10,2,3) =0.35
P(1,10,2,4) =0.70	P(2,10,2,4) =0.30	P(1,10,2,5) =0.70	P(2,10,2,5) =0.30
P(1,11,1,1) =0.30	P(2,11,1,1) =0.70	P(1,11,1,2) =0.35	P(2,11,1,2) =0.65
P(1,11,1,3) =0.70	P(2,11,1,3) =0.30	P(1,11,1,4) =0.40	P(2,11,1,4) =0.60
P(1,11,1,5) =0.40	P(2,11,1,5) =0.60	P(1,11,2,1) =0.30	P(2,11,2,1) =0.70
P(1,11,2,2) =0.40	P(2,11,2,2) =0.60	P(1,11,2,3) =0.70	P(2,11,2,3) =0.30
P(1,11,2,4) =0.70	P(2,11,2,4) =0.30	P(1,11,2,5) =0.70	P(2,11,2,5) =0.30
P(1,12,1,1) =0.30	P(2,12,1,1) =0.70	P(1,12,1,2) =0.35	P(2,12,1,2) =0.65
P(1,12,1,3) =0.70	P(2,12,1,3) =0.30	P(1,12,1,4) =0.40	P(2,12,1,4) =0.60
P(1,12,1,5) =0.40	P(2,12,1,5) =0.60	P(1,12,2,1) =0.30	P(2,12,2,1) =0.70
P(1,12,2,2) =0.40	P(2,12,2,2) =0.60	P(1,12,2,3) =0.70	P(2,12,2,3) =0.30
P(1,12,2,4) =0.70	P(2,12,2,4) =0.30	P(1,12,2,5) =0.70	P(2,12,2,5) =0.30

Cuadro 5.7: *KBM2L* inicial de TPC de dimensionalidad media, ($K = 101$), $\bar{Y} = E$, CH, FC

Item	EA	$\bar{Y}$	p	Item	EA	$\bar{Y}$	$\{0,1\}$
0	$\langle$ (0)	000,	$40\|^1$	10	$\langle$ (0)	110,	$50\|^1$
1	$\langle$ (0)	001,	$30\|^1$	11	$\langle$ (0)	111,	$55\|^1$
2	$\langle$ (0)	002,	$35\|^1$	12	$\langle$ (0)	112,	$75\|^1$
3	$\langle$ (0)	004,	$25\|^2$	13	$\langle$ (0)	114,	$60\|^2$
4	$\langle$ (0)	011,	$70\|^2$	14	$\langle$ (0)	200,	$20\|^1$
5	$\langle$ (0)	012,	$75\|^1$	15	$\langle$ (0)	201,	$25\|^1$
6	$\langle$ (0)	014,	$60\|^2$	16	$\langle$ (0)	202,	$50\|^1$
7	$\langle$ (0)	101,	$30\|^2$	...	...	...	...
8	$\langle$ (0)	102,	$35\|^1$	...	...	...	...
9	$\langle$ (0)	104,	$15\|^2$	181	$\langle$ (1)	1114,	$30\|^3$

Cuadro 5.8: *KBM2L* optimizada de TPC de dimensionalidad media, ($K = 101$), $\bar{Y} = CH$, FC, E

Item	EA	$\bar{Y}$	p	Item	EA	$\bar{Y}$	$\{0,1\}$
0	$\langle$ (0)	000,	$40\|^1$	10	$\langle$ (0)	018,	$30\|^1$
1	$\langle$ (0)	001,	$30\|^1$	11	$\langle$ (0)	021,	$35\|^5$
2	$\langle$ (0)	002,	$20\|^1$	12	$\langle$ (0)	025,	$50\|^4$
3	$\langle$ (0)	006,	$10\|^4$	13	$\langle$ (0)	027,	$55\|^2$
4	$\langle$ (0)	007,	$20\|^1$	14	$\langle$ (0)	028,	$60\|^1$
5	$\langle$ (0)	008,	$25\|^1$	15	$\langle$ (0)	029,	$64\|^1$
6	$\langle$ (0)	011,	$30\|^5$	16	$\langle$ (0)	0011,	$70\|^2$
7	$\langle$ (0)	012,	$25\|^1$	...	...	...	...
8	$\langle$ (0)	016,	$20\|^4$	...	...	...	...
9	$\langle$ (0)	017,	$25\|^1$	129	$\langle$ (1)	1411,	$0\|^0$

más jovenes y decrementa en los más mayores.

Item: 76	Indice: [1, 0, 2, 1] Respuesta: $p = 65$
	EA: NO, CH: GradoBajo, FC: II2, E: 20a29
Item: 77	Indice: [1, 0, 2, 5], 3 Atributos fijos, 4 Casos
	EA: NO, CH: GradoBajo, FC: II2, E: 50a54
Item: 78	Indice: [1, 0, 2, 7] Respuesta: $p = 45$
	EA: NO, CH: GradoBajo, FC: II2, E: 60a64

En el ítem 68 observamos que EA = NO presenta una probabilidad de 0.90 para CH = Gradobajo, FC = I y edades comprendidas entre 40 y 59 años, que se decrementa en pacientes más jovenes y más mayores.

Item: 67	Indice: [1, 0, 0, 2] Respuesta: $p = 80$
	EA: NO, CH: GradoBajo, FC: I, E: 30a39
Item: 68	Indice: [1, 0, 0, 6], Atributos fijos 3, 4 Casos
	EA: NO, CH: GradoBajo, FC: I, E: 55a59
Item: 69	Indice: [1, 0, 0, 7] Respuesta: $p = 80$
	EA: NO, CH: GradoBajo, FC: I, E: 60a64

La descripción de los ítems anteriores sugiere que la lista se puede analizar respecto de ciertos umbrales de la respuesta [0,1]. Por ejemplo, entre 0 y 0.10 es poco probable, entre 0.11

y 0.40 la probablidad es moderada, entre 0.41 y 0.60 es probable, entre 0.61 y 0.85 es bastante probable y entre 0.86 y 1 es muy probable. Pensamos que esta escala depende del problema y la deben ajustar los expertos de acuerdo a las expectativas del análisis que se pretenda realizar. De este modo, podemos agrupar mediante la lista los factores, dimensiones de la tabla, según su importancia en la determinación del nivel de probabilidad. La Figura 5.4 muestra el espectro de la TPC de EA con los 5 niveles indicados para p en la *Base* óptima $[FC, CH, EA, E]$ y 53 ítems, frente a los 130 ítems de la lista en la *Base* inicial $[EA, CH, FC, E]$, con 101 niveles para p.

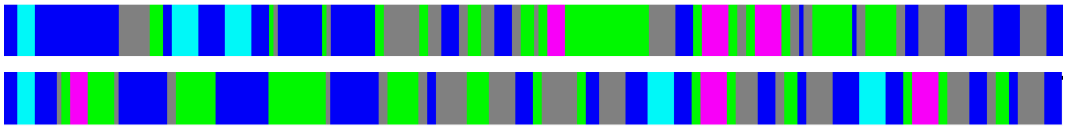

Figura 5.4: Espectro de las listas *KBM2L* inicial (arriba) y optimizada (abajo) para la TPC de EA con probabilidad como respuesta (5 niveles)

Observamos los ítems 1, 30 y 42 que reúnen los 16 casos de nivel p más bajo, entre 0 y 0.10. Todos los pacientes son mayores de 40 y menores de 65 años, EA=SI, CH=GradoBajo y no presentan la FC=II1 ni FC=II2. Los ítems 5, 33, 45 son el complemento (EA=NO) de los anteriores, ahora con niveles más altos (más de 0.85) y presentan un patrón en el espectro similar al de 1, 30 y 42 desplazado 12 casos a la izquierda.

A continuación, vamos a utilizar la declaración de la tabla como respuesta, véase la sección 5.2.2. Aunque no tenemos imprecisión en la TPC, esta vista de la tabla puede ser interesante y además mostraremos los detalles de la representación. Hemos considerado dos resoluciones con 101 (0, 1, 2,. . . , 99, 100) y 21 (0, 1, 2,. . . , 19, 20) valores para el dominio discreto de la probabilidad y la representación lógica de la *KBM2L* para las 240 declaraciones de elementos de probabilidad. Ambas discretizaciones son uniformes en el intervalo real [0,1], salvo el extremo superior. Así, el valor 1 se corresponde en la primera con p=0.01 y en la segunda con p=0.05.

La lista vacía en la *Base* $[p, EA, E, CH, FC]$ es: $\langle([100], (1), 11, 1, 4), -1|$ con $K = 101$ y la lista con el contenido de la TPC tiene 365 ítems para representar los 240 casos en un espacio de 24240 celdas, véanse el Cuadro 5.9 y la Figura 5.5.

En la *Base* $[EA, p, CH, FC, E]$ la lista tiene 261 ítems, véanse el Cuadro 5.10 y la Figura 5.6.

La interpretación puede ser la siguiente. En la TPC de EA, el atributo más relevante es CH y a continuación FC. La E es el atributo menos relevante en la tabla. Hay tramos de la E donde se mantiene constante la probabilidad, fijados los valores del resto de variables.

La lista se puede analizar respecto de la respuesta 0 o la 1. 0 es equivalente a considerar

Cuadro 5.9: *KBM2L* inicial de TPC de dimensionalidad media, ($K = 101$), $\bar{Y} = E, CH, FC$

Item	p	EA	$\bar{Y}$	$\{0,1\}$	Item	p	EA	$\bar{Y}$	$\{0,1\}$
0	⟨[5]	(0)	202,	$0\|^{1223}$	10	⟨[10]	(0)	214,	$0\|^{1165}$
1	⟨[5]	(0)	204,	$1\|^{2}$	11	⟨[10]	(0)	300,	$1\|^{1}$
2	⟨[5]	(0)	302,	$0\|^{8}$	12	⟨[10]	(0)	134,	$0\|^{9}$
3	⟨[5]	(0)	304,	$1\|^{2}$	13	⟨[10]	(0)	400,	$1\|^{1}$
4	⟨[5]	(0)	402,	$0\|^{8}$	14	⟨[10]	(0)	414,	$0\|^{9}$
5	⟨[5]	(0)	404,	$1\|^{2}$	15	⟨[10]	(0)	500,	$1\|^{1}$
6	⟨[5]	(0)	502,	$0\|^{8}$	16	⟨[10]	(0)	514,	$0\|^{9}$
7	⟨[5]	(0)	504,	$1\|^{2}$	...	...	...	...	...
8	⟨[5]	(0)	602,	$0\|^{8}$	...	...	...	...	...
9	⟨[5]	(0)	604,	$1\|^{2}$	364	⟨[100]	(1)	1114,	$0\|^{1255}$

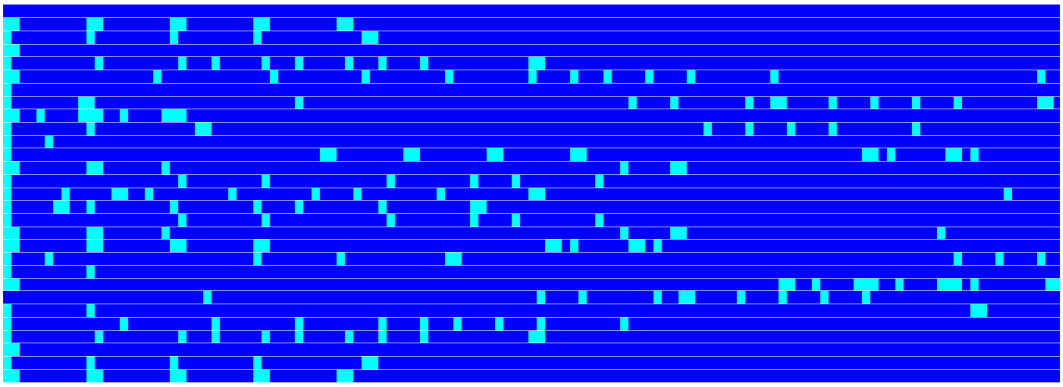

Figura 5.5: Espectro de la lista *KBM2L* inicial para la TPC de EA con la declaración como respuesta, $K = 101$

restricciones sobre la TPC. Por ejemplo, consideremos los ítems más grandes, que tienen explicación más simple. Observamos que dichos ítems en esta TPC corresponden a los valores de respuesta 0.

Vamos a analizar algunos ítems. Los ítems 78, 130 y 182 constan de 1111, 6118 y 1084 casos de respuesta 0, respectivamente. Significa que la probabilidad de las declaraciones asociadas no es cierta. Examinamos el ítem 78 y observamos que EA=SI no presenta una probabilidad entre 0,41 y 0,49 para todas las combinaciones de valores de los atributos condicionantes.

Cuadro 5.10: *KBM2L* optimizada de TPC de dimensionalidad media, ($K = 101$), $\bar{Y} = CH$, FC, E

Item	EA	p	$\bar{Y}$	$\{0,1\}$	Item	EA	p	$\bar{Y}$	$\{0,1\}$
0	⟨(0)	[5]	031,	$0\|^{638}$	10	⟨(0)	[15]	030,	$0\|^{581}$
1	⟨(0)	[5]	036,	$1\|^{5}$	11	⟨(0)	[15]	031,	$1\|^{1}$
2	⟨(0)	[5]	041,	$0\|^{7}$	12	⟨(0)	[15]	040,	$0\|^{11}$
3	⟨(0)	[5]	046,	$1\|^{5}$	13	⟨(0)	[15]	041,	$1\|^{1}$
4	⟨(0)	[10]	002,	$0\|^{548}$	14	⟨(0)	[20]	001,	$0\|^{552}$
5	⟨(0)	[10]	006,	$1\|^{4}$	15	⟨(0)	[20]	002,	$1\|^{1}$
6	⟨(0)	[10]	036,	$0\|^{36}$	16	⟨(0)	[20]	006,	$0\|^{4}$
7	⟨(0)	[10]	037,	$1\|^{1}$	...	...	...	...	...
8	⟨(0)	[10]	046,	$0\|^{11}$	...	...	...	...	...
9	⟨(0)	[10]	047,	$1\|^{1}$	260	⟨(1)	[100]	1411,	$0\|^{665}$

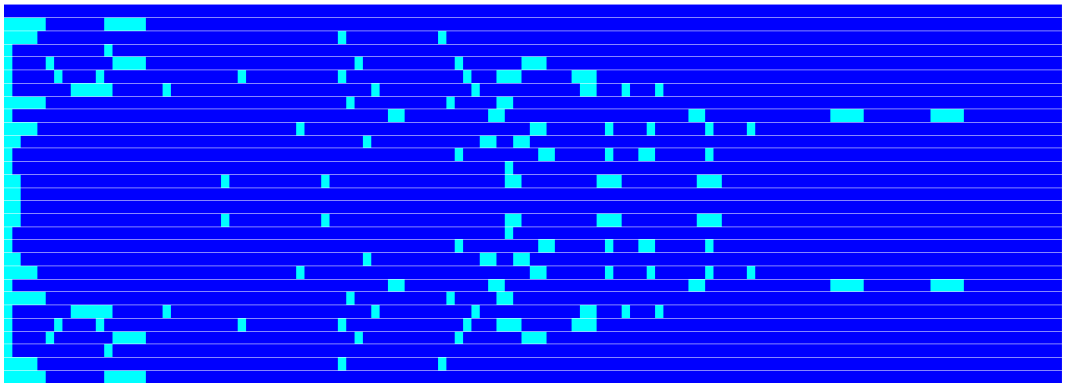

Figura 5.6: Espectro de la lista *KBM2L* optimizada para la TPC de EA con la declaración como respuesta, $K = 101$

Item: 77	Indice: [0, 40, 1, 4, 6]
Item: 78	Indice: [0, 50, 0, 2, 1], No hay atributos fijos, 1111 Casos EA: SI, p: p50, CH: GradoBajo, FC: II2, E: 20a29

Respecto del ítem 130, vemos que su parte fija es vacía. Consideramos las dos partes del ítem que se derivan de fijar el valor de EA. La primera supone que EA=SI no presenta una probabilidad entre 0.76 y 1.0 para todas las combinaciones de valores de los atributos condicionantes. La segunda supone que EA=NO no presenta una probabilidad entre 0.0 y 0.24 para todas las

combinaciones de valores de los atributos condicionantes.

Item: 129	Indice: [0, 75, 1, 2, 1]
Item: 130	Indice: [1, 25, 1, 1, 11], No hay atributos fijos, 6118 Casos
	EA: NO, p: p25, CH: GradoAlto, FC: II1, E: masde90
Item: 130/EA=0	Indice: [0, 100, 1, 4, 11]
Item: 130/EA=1	Indice: [1, 25, 1, 1, 11]

Por último, consideramos el ítem grande número 182. Observamos que la probabilidad de EA=NO nunca está en el intervalo [0.51, 0.59] para todas las combinaciones de valores de los atributos condicionantes.

Item: 181	Indice: [1, 50, 1, 4, 7]
Item: 182	Indice: [1, 59, 1, 4, 11], No hay atributos fijos, 1084 Casos
	EA: NO, p: p59, CH: GradoAlto, FC: IV, E: masde90

Analizamos ahora ítems pequeños. Comparamos los ítems 75 y 77 cuyas respuestas son iguales a 1. Ambos ítems representan 4 casos. La probabilidad de EA=SI condicionada a CH=GradoAlto es 0,40 y no cambia entre 40 y 59 años, siempre que el FC sea III o IV.

Item: 75	Indice: [0, 40, 1, 3, 6], 4 Atributos fijos, 4 Casos
	EA: SI, p: p40, CH: GradoAlto, FC: III, E: 40a44
	EA: SI, p: p40, CH: GradoAlto, FC: III, E: 45a49
	EA: SI, p: p40, CH: GradoAlto, FC: III, E: 50a54
	EA: SI, p: p40, CH: GradoAlto, FC: III, E: 55a59
Item: 77	Indice: [0, 40, 1, 4, 6], 4 Atributos fijos, 4 Casos
	EA: SI, p: p40, CH: GradoAlto, FC: IV, E: 40a44
	EA: SI, p: p40, CH: GradoAlto, FC: IV, E: 45a49
	EA: SI, p: p40, CH: GradoAlto, FC: IV, E: 50a59
	EA: SI, p: p40, CH: GradoAlto, FC: IV, E: 55a59

De los ítems 1 y 3 (EA=SI), podemos decir análogamente que entre 40 y 59 años siempre que el FC sea III o IV, la probabilidad es 0.05.

Item: 1	Indice: [0, 5, 0, 3, 6], 4 Atributos fijos, 4 Casos
	EA: SI, p: p05, CH: GradoBajo, FC: III, E: 30a39
	EA: SI, p: p05, CH: GradoBajo, FC: III, E: 40a44
	EA: SI, p: p05, CH: GradoBajo, FC: III, E: 45a49
	EA: SI, p: p05, CH: GradoBajo, FC: III, E: 50a54
	EA: SI, p: p05, CH: GradoBajo, FC: III, E: 55a59
Item: 3	Indice: [0, 5, 0, 4, 6], 4 Atributos fijos, 4 Casos
	EA: SI, p: p05, CH: GradoBajo, FC: IV, E: 30a39
	EA: SI, p: p05, CH: GradoBajo, FC: IV, E: 40a44
	EA: SI, p: p05, CH: GradoBajo, FC: IV, E: 45a49
	EA: SI, p: p05, CH: GradoBajo, FC: IV, E: 50a59
	EA: SI, p: p05, CH: GradoBajo, FC: IV, E: 55a59

Comparamos los ítems 69 y 71 cuyas respuestas son iguales a 1. Los tres ítems representan dos casos cada uno. La probabilidad de EA = SI condicionada a CH = GradoBajo es 0.40 y no cambia a partir de 80 años, siempre que el FC sea II1, III o IV.

Item: 69	Indice: [0, 40, 0, 3, 11], 4 Atributos fijos, 2 Casos
	EA: SI, p: p40, CH: GradoBajo, FC: III, E: 80a89
	EA: SI, p: p40, CH: GradoBajo, FC: III, E: masde90
Item: 71	Indice: [0, 40, 0, 4, 11], 4 Atributos fijos, 2 Casos
	EA: SI, p: p40, CH: GradoBajo, FC: IV, E: 80a89
	EA: SI, p: p40, CH: GradoBajo, FC: IV, E: masde90

La lista óptima que hemos obtenido en la TPC de EA es casi un 30 % más reducida que la original, lo cual está muy bien en una tabla tan dispersa. Creemos que la representación mediante los 261 ítems es más clara que la representación convencional en una tabla, donde el orden no es relevante. La mitad de los ítems, los más grandes con respuesta 0, constituyen una representación dual del conocimiento que recoge la distribución de probabilidad. Los ítems de respuesta 1 son 130 y representan las 240 declaraciones de la TPC original organizadas por el peso (importancia) de los atributos.

Veamos ahora los resultados de la otra discretización. La lista vacía en la $Base$ [p, EA, E, CH, FC] es: $\langle([20],(2),11,1,4),-1|$ con $K = 21$ y tiene 365 ítems para representar los 240 casos en un espacio de 5040 celdas. Tiene un esquema de menor resolución para p, que se adapta, en principio, más a la TPC cuyo contenido, véanse los Cuadros 5.5 y 5.6, son múltiplos de 5 divididos por 100 (que son 21 valores distintos). Pero observamos que tiene peor lista óptima, con 319 ítems (en vez de los 261 que tenía con la otra resolución).

La Figura 5.7 muestra el espectro de las listas inicial y optimizada.

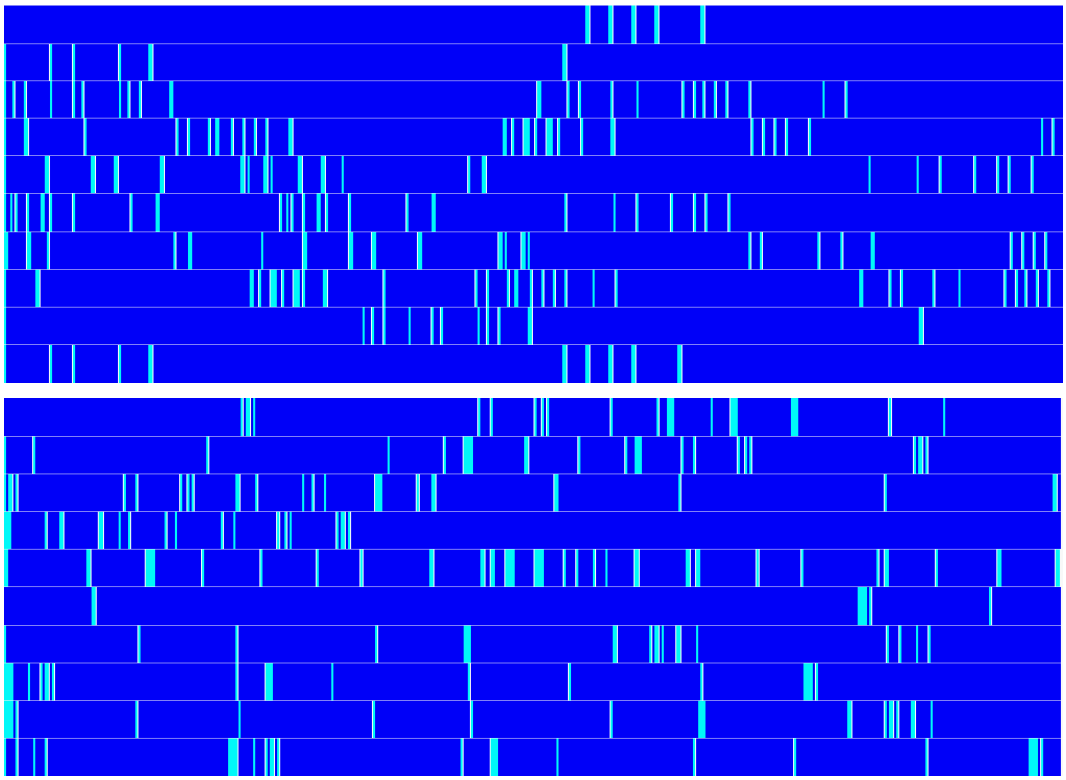

Figura 5.7: Espectros de las listas *KBM2L* inicial (arriba) y optimizada (abajo) para la TPC de EA, $K = 21$

Por último, el espectro muestra la organización, la proximidad de las declaraciones y la simetría (en esta tabla es muy clara debido a que la variable condicionada es binaria). Esta representación gráfica puede ser útil para planificar con los expertos el análisis de sensibilidad de la TPC marcando las probabilidades de interés, los intervalos de variación y las regiones de sensibilidad. En el capítulo próximo proponemos utilizar las listas *KBM2L* para realizar AS de las probabilidades y utilidades.

5.5. Aplicación al Sistema IctNeo

En esta sección volvemos a hacer uso del SAD IctNeo para mostrar un ejemplo real de alta dimensionalidad adaptado para la representación de una TPC con valores lógicos de respuesta. En este modelo necesitamos asignar muchas TPC grandes sobre la ictericia neonatal.

Nuestro ejemplo es una TPC para la concentración de hemoglobina (X=CHgb: baja(0)/-normal(1)/alta(2)) de un paciente, condicionada a varias (8) patologías (ausente(0), presente(1))

y la edad crítica del paciente (No(0)/Sí(1)). Por consiguiente, sea $\bar{Y}$ el vector de contexto con 9 componentes binarias. Tomemos una discretización de tamaño 101 para la resolución de p. La lista vacía es $\langle [100](1)\bar{1}, unKB|$ con 155136 casos. La TPC tiene $3 * 2^9 = 1536$ parámetros, pero sólo 1024 parámetros son libres, por el hecho de ser un conjunto de distribuciones de probabilidad (el paso al complementario de la unidad). En (Bielza et al., 2000) el problema de educción fue parcialmente resuelto por medio de una *puerta OR* (noisy OR–gate) (Pearl, 1988) *graduada* (Díez, 1994) y todos los parámetros de probabilidad fueron precisos. Pero frecuentemente es difícil que las hipótesis que permiten utilizar correctamente la puerta OR se satisfagan y tratamos aquí de asignar directamente la TPC: $P(CHgb|Edad, Patolog_1, \ldots, Patolog_8)$.

Supongamos que la información inicial es nula (la lista está vacía) o que hay alguna información a priori como: $P(0|000000000) = 0$, $P(1|000000000) = 1$, $P(2|000000000) = 0$, $P(0|111111111) = 1$, $P(1|111111111) = 0$, $P(2|111111111) = 0$. Se trata de seis declaraciones relativas a la distribución, que indican, respectivamente:

1. Si no hay ninguna patología presente, seguro que la CHgb no es patológicamente baja; tiene probabilidad 0.

2. Si no hay ninguna patología presente, seguro que la CHgb es patológicamente normal; tiene probabilidad 1.

3. Si no hay ninguna patología presente, seguro que la CHgb no es patológicamente alta; tiene probabilidad 0.

4. Si todas las patologías están presentes, seguro que la CHgb es patológicamente baja; tiene probabilidad igual a 1.

5. Si todas las patologías están presentes, seguro que la CHgb no es patológicamente normal; tiene probabilidad igual a 0.

6. Y por último, si todas las patologías están presentes, seguro que la CHgb no es patológicamente alta; tiene probabilidad igual a 0.

De las tres últimas declaraciones cabe decir que el contexto es poco probable, pues las 8 patologías no se han presentado simultáneamente en un paciente, aún cuando puede ser posible. El experto podría suponer ciertas restricciones en este sentido y utilizar la etiqueta *coKB*.

La lista inicial en la *Base* $[p, X, \bar{Y}]$ se muestra en el Cuadro 5.11.

Con la lista inicial mostrada, el analista pregunta al experto (o realiza consultas a la base de datos) por el conjunto de proposiciones con probabilidad igual a 1.0, 0.90, ..., 0.0. El experto

Cuadro 5.11: *KBM2L* inicial de TPC de dimensionalidad alta

Item	p	X	$\bar{Y}$	$\{0,1\}$
0	$\langle[0]$	(0)	000000000,	$1\|^{1}$
1	$\langle[0]$	(1)	111111110,	$-1\|^{1022}$
2	$\langle[0]$	(2)	000000000,	$1\|^{2}$
3	$\langle[0]$	(2)	111111110,	$-1\|^{510}$
4	$\langle[0]$	(2)	111111111,	$1\|^{1}$
5	$\langle[100]$	(0)	111111110,	$-1\|^{152575}$
6	$\langle[100]$	(1)	000000000,	$1\|^{2}$
7	$\langle[100]$	(2)	111111111,	$-1\|^{1023}$

declara cuáles son los contextos con $p = 100, 90, \ldots, 10, 0$. Este proceso puede repetirse varias veces sobre toda la lista o sobre los ítems desconocidos, en contextos restringidos. La idea principal es llenar de conocimiento la lista que inicialmente es un contenedor vacío. También, el experto puede declarar reglas del dominio, restricciones (como la indicada anteriormente) o atributos que modulan la probabilidad de otros. Por ejemplo, puede dar las reglas siguientes:

1. *Cuando tres o más patologías están presentes, CHgb es patológicamente baja con probabilidad 1,*

2. *Cuando una patología está presente, la edad crítica del paciente incrementa la probabilidad de CHgb patológicamente baja alrededor de un 10 %.*

3. *No más de cuatro patologías están presentes en un paciente, y la patología 1 y la patología 8 nunca se presentan conjuntamente,...*

La primera es un regla del dominio que establece una relación determinista en el contexto señalado. La segunda revela el papel de un atributo (la edad crítica del paciente) que en cierto contexto es un factor que puede modular (incrementar o disminuir) la probabilidad. La tercera es un conjunto de restricciones sobre el conjunto de atributos condicionantes.

Debemos considerar ciertas metarreglas para que el proceso de educción sea consistente. Debe considerarse un orden de aplicación de mayor a menor generalidad, es decir, primero aquellas reglas que afectan a mayor número de casos en la TPC. Las reglas que soportan restricciones prevalecen sobre las restantes y si hay conflictos entre las reglas deben ser detectadas y resueltas mediante prioridades entre las reglas o refinando el conjunto. La reglas que modulan la proba-

bilidad deben aplicarse de nuevo siempre que se modifiquen las probabilidades a las que hacen referencia en la TPC durante el proceso. Por último, si hay reglas redundantes también deben ser identificadas por el procedimiento.

La lista *KBM2L* debe ser consistente, véase la sección 2.3.1, mientras la información (parámetros, reglas o restricciones) se introduce en la estructura. Observamos que al considerar la representación de la TPC con la declaración como respuesta no realizamos la asignación de los parámetros de probabilidad en el sentido usual. La lista *KBM2L* representa todas las posibles asignaciones (con p discretizada) y el procedimiento, en realidad, da soporte a la identificación de la distribución de probabilidad requerida por el experto.

En la Figura 5.8 mostramos el proceso de optimización de la lista y de incorporación de las 3 reglas. Las dimensiones p y X se denotan con 0 y 1, respectivamente, en las *Bases* inicial y final. Aquí los espectros muestran el comienzo de construcción de la lista porque hay ítems desconocidos muy grandes. Los procesos de construcción y optimización pueden realizarse concurrentemente. El resultado final es la TPC optimizada, véanse la Figura 5.8 y el Cuadro 5.12. Los ítems extremadamente grandes, de respuesta desconocida, muestran su espectro con longitud no proporcional a su tamaño en la fase inicial de educción. Mostramos un fragmento del principio de los espectros de las listas que tienen ítems extremadamente grandes, de respuesta desconocida, la fase de educción que alcanza el ejemplo. El sistema ayuda al analista a formular preguntas al experto. Ambos disponen del espectro de la TPC que muestra las interrelaciones entre los atributos que van apareciendo conforme avanza la sesión de educción.

En el Cuadro 5.12 mostramos un fragmento de la lista óptima.

La síntesis de TPC (por ejemplo de distribuciones a posteriori) podría servirnos para realizar diagnósticos en problemas de decisiones clínicas.

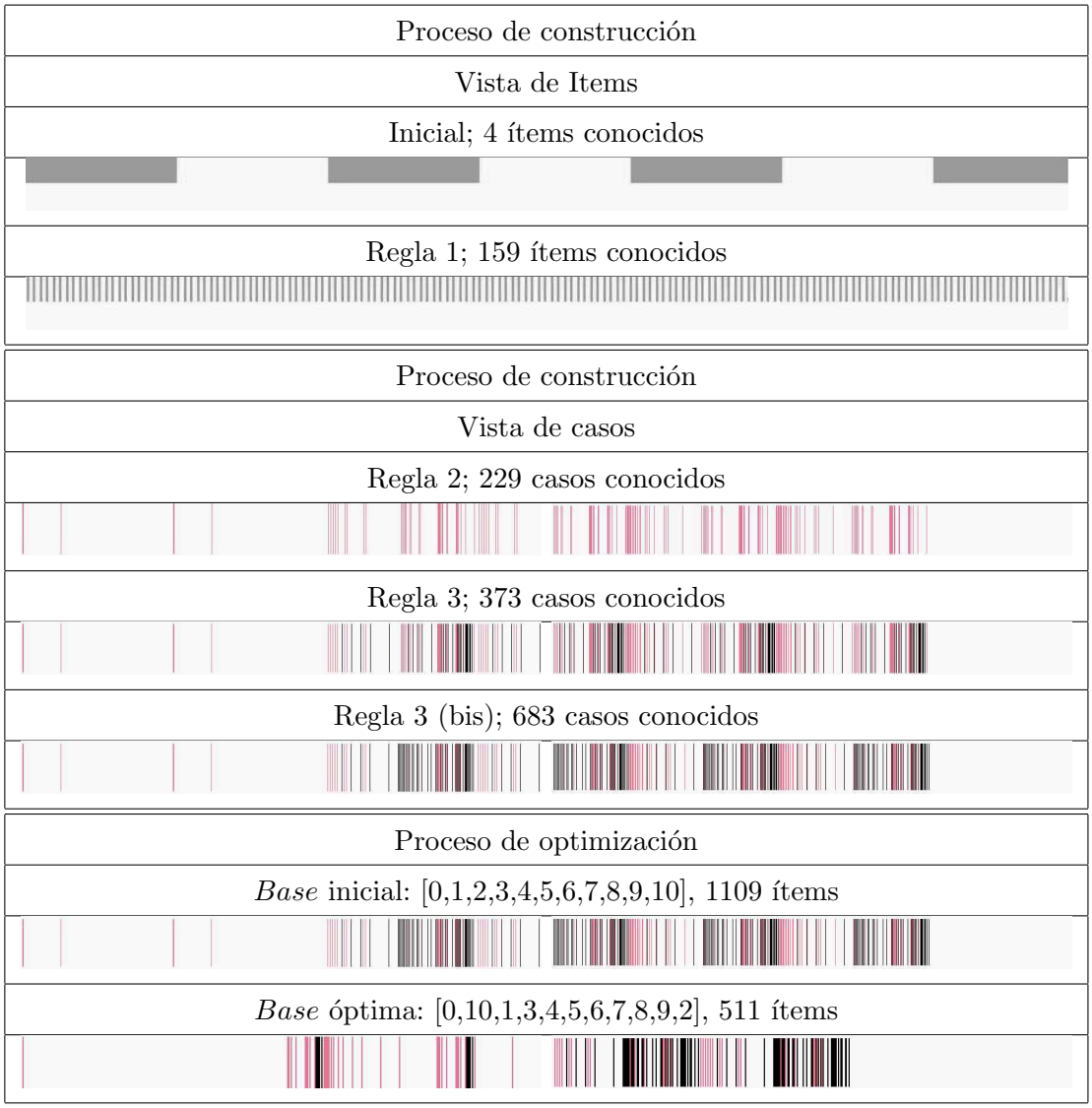

Figura 5.8: TPC de alta dimensión

Cuadro 5.12: *KBM2L* óptima de TPC de alta dimensión

Item	p	Y_{10}	X	$\{Y_3, Y_4, Y_5, Y_6$ $Y_7, Y_8, Y_9, Y_2\}$	$\{0, 1\}$
0	⟨[0]	0	(0)	00000001,	$1\|^2$
1	⟨[0]	0	(1)	11000001,	$-1\|^{448}$
2	⟨[0]	0	(1)	11000101,	$1\|^4$
3	⟨[0]	0	(1)	11000111,	$-1\|^2$
4	⟨[0]	0	(1)	11001001,	$1\|^2$
5	⟨[0]	0	(1)	11001111,	$-1\|^6$
6	⟨[0]	0	(1)	11010001,	$1\|^2$
7	⟨[0]	0	(1)	11011111,	$-1\|^{14}$
8	⟨[0]	0	(1)	11100101,	$1\|^6$
9	⟨[0]	0	(1)	11100111,	$-1\|^2$
10	⟨[0]	0	(1)	11101001,	$1\|^2$
11	⟨[0]	0	(1)	11101111,	$-1\|^6$
12	⟨[0]	0	(1)	11110001,	$1\|^2$
13	⟨[0]	0	(1)	11111001,	$-1\|^8$
14	⟨[0]	0	(1)	11111011,	$1\|^2$
...	...	...	...	...	...
184	⟨[0]	1	(2)	11111010,	$0\|^3$
185	⟨[0]	1	(2)	11111101,	$-1\|^3$
186	⟨[0]	1	(2)	11111111,	$0\|^2$
187	⟨[1]	0	(1)	10000000,	$-1\|^{385}$
188	⟨[1]	0	(1)	10000001,	$1\|^1$
189	⟨[1]	0	(1)	11100000,	$-1\|^{63}$
190	⟨[1]	0	(1)	11100001,	$1\|^1$

Item	p	Y_{10}	X	$\{Y_3, Y_4, Y_5, Y_6$ $Y_7, Y_8, Y_9, Y_2\}$	$\{0, 1\}$
191	⟨[3]	0	(1)	01000000,	$-1\|^{2943}$
192	⟨[3]	0	(1)	01000001,	$1\|^1$
193	⟨[3]	0	(1)	01100000,	$-1\|^{31}$
194	⟨[3]	0	(1)	01100001,	$1\|^1$
195	⟨[10]	0	(1)	01111111,	$-1\|^{10782}$
196	⟨[10]	0	(1)	11000000,	$1\|^1$
197	⟨[10]	0	(1)	11011111,	$-1\|^{63}$
198	⟨[10]	0	(1)	11100000,	$1\|^1$
...	...	...	...	...	...
253	⟨[30]	1	(1)	00000001,	$-1\|^{763}$
254	⟨[30]	1	(1)	00000010,	$1\|^1$
255	⟨[50]	0	(0)	11110001,	$-1\|^{29935}$
256	⟨[50]	0	(0)	11110011,	$0\|^2$
257	⟨[50]	0	(0)	11110101,	$-1\|^2$
258	⟨[50]	0	(0)	11110111,	$0\|^2$
259	⟨[50]	0	(0)	11111011,	$-1\|^4$
...	...	...	...	...	...
506	⟨[100]	1	(0)	11110111,	$-1\|^4$
507	⟨[100]	1	(0)	11111001,	$0\|^2$
508	⟨[100]	1	(0)	11111101,	$-1\|^4$
509	⟨[100]	1	(0)	11111111,	$0\|^2$
510	⟨[100]	1	(2)	11111111,	$-1\|^{512}$

Capítulo 6

KBM2L para Análisis de Sensibilidad

El capítulo 6 explora las posibilidades de representación de las listas definidas en esta tesis para el Análisis de Sensibilidad del modelo (Fernández del Pozo y Bielza, 2004).

La sección 6.1 motiva el análisis de sensibilidad en el ámbito de los modelos gráficos probabilísticos y muestra las interconexiones del análisis de sensibilidad con la explicación de los resultados o inferencias que proporciona el sistema de ayuda a la decisión.

En las secciones 6.2 y 6.3 desarrollamos la teoría que permite realizar análisis de sensibilidad utilizando listas *KBM2L*. Proponemos una estrategia que parte del modelo de referencia, con los valores de los parámetros que se consideran más ajustados. Posteriormente extendemos el esquema de las listas que representan las tablas de decisiones óptimas para gestionar los distintos modelos (con diferentes valores de los parámetros) en una misma lista. A continuación generamos una muestra aleatoria en el espacio de parámetros. En la lista extendida y con los modelos representados en la muestra, identificamos los casos cuya respuesta discrepa respecto del modelo de referencia. Finalmente, explicamos las discrepancias (respuesta sensible) mediante el subesquema de parámetros, y la sensibilidad de los casos en el intervalo de variación de los parámetros mediante el subesquema de variables del modelo.

En la sección 6.4 las dificultades computacionales y la interpretación de los resultados que se presentan al abordar el análisis de sensibilidad se tratan de ilustrar con un ejemplo sencillo.

6.1. Análisis de Sensibilidad de Modelos

Los métodos de Análisis de Sensibilidad (AS) estudian la estabilidad de las alternativas óptimas identificadas por el modelo de decisión respecto de la variación de los parámetros numéricos

y estructurales. El AS es esencial para el refinamiento y la validación de los modelos y hace más robusta la educción de creencias y preferencias por el experto. (French, 2003) muestra que el AS puede desempeñar diferentes papeles: modelización, inferencia y toma de decisiones, según el propósito del estudio. Adoptando el punto de vista bayesiano, sugiere diversas e importantes tareas del Análisis de Decisiones que progresan apoyándose en los resultados del AS:

- Construcción y exploración de modelos de las consecuencias, es decir, el efecto de las decisiones tomadas en los distintos estados del problema;
- Exploración de las relaciones entre modelos de probabilidad y modelos de consecuencias;
- Ayuda a la educción de entradas subjetivas y aprendizaje de los datos en un análisis;
- Desarrollo de algoritmos computacionalmente eficientes como mecanismos de inferencia (teorema de Bayes);
- Diseño de experimentos que nos suministran datos observados;
- Guía en la realización de inferencias, predicciones y toma de decisiones;
- Exploración y obtención de consensos, con respecto a las probabilidades y las utilidades;
- Comprensión de los problemas: los datos, el modelo y el mecanismo de inferencia.

En este capítulo nos centramos en el punto segundo y en los dos últimos. Nos interesa conocer el grado de sensibilidad que muestran los resultados que propone el SAD cuando varía ligeramente el modelo de probabilidad y utilidad. Además hemos intentado mejorar la comprensión del modelo mediante la explicación de las TPC (capítulo 5). Todo ello redundará en valores de probabilidades y utilidades sobre los que tendremos más confianza, estando más consensuados.

Respecto al tipo de AS, reflexionemos acerca de las dos características indicadas en (French, 2003) relativas al alcance del AS. Primeramente nuestro AS podría calificarse como *global* pero pensamos que dicho concepto es mejor referirlo a la naturaleza heterogénea de los parámetros (probabilidades de cualquiera de las TPC y utilidades) que a la amplitud de la medida del entorno de variación considerado. Un entorno multiparamétrico continuo muy amplio supone una comparación entre modelos demasiado divergentes. En consecuencia, consideramos un AS *local* cuando nos centramos en una TPC o en utilidades de un contexto de atributos que muestra la variación de uno. En segundo término, consideramos la variación combinada de muchos parámetros, más de 10, de forma independiente e idénticamente distribuida (uniforme) en intervalos centrados en valores base o de referencia.

Durante la construcción y validación de un SAD, el AS y la explicación de los resultados del modelo aparecen como tareas esenciales. Por una parte, el AS considera que tenemos ya un modelo construido para el problema de toma de decisiones, el Modelo de Referencia (MR), y analizamos y comparamos sus resultados con modelos similares, es decir con información cuantitativa (y/o cualitativa) parecida.

En general, suponemos que los parámetros tienen unos valores base o de referencia en el MR y en un entorno "pequeño" de dicha referencia podemos tolerar cierta imprecisión. Los decisores o expertos reconocen la posible imprecisión que rodea a sus estimaciones puntuales para los parámetros de entrada, y se preguntan por su influencia en la salida o resultado del modelo. Por consiguiente, los decisores están interesados en las interrelaciones entre los cambios en estas estimaciones y los consiguientes cambios inducidos en el resultado que aporta la evaluación del modelo. Como resultado básico del AS se obtiene el subconjunto del entorno de variación de los parámetros que muestra el mismo comportamiento que el MR. El comportamiento del MR está definido por las decisiones óptimas que propone en cada caso, el valor de utilidad esperada en cada decisión, y la relevancia de las variables en el proceso de decisión. El MR puede ser sensible a la variación de los parámetros manifestando cambios en las decisiones, utilidades y explicaciones.

Por otra parte, los decisores pueden encontrar difícil aceptar los resultados del SAD, por ejemplo, las propuestas o recomendaciones acerca de las decisiones óptimas, si no comprenden las razones que hay tras ellas y su robustez respecto del modelo. Es decir, necesitan explicaciones de los resultados del SAD que los justifiquen, suministrando al experto nuevas intuiciones sobre el problema, y, como una síntesis del conocimiento, pueden servir también para validar el SAD.

Así, el AS y la explicación están estrechamente relacionados. La diferencia entre ambos reside en los elementos que utilizan. Las explicaciones se refieren a los resultados del modelo, es decir, a las estrategias óptimas, como función de las variables del modelo: "ésta es la alternativa preferida porque la variable A toma el valor a". El AS se refiere a los resultados del modelo y quizás a algún indicador complementario (como la utilidad máxima esperada), como función de algunos parámetros del modelo. Estos parámetros normalmente definen las relaciones cuantitativas entre las variables del modelo. Podemos obtener hallazgos como: "ésta es la alternativa preferida y lo sigue siendo en tanto que el parámetro K pertenezca al intervalo I_K. La utilidad máxima esperada recorre el intervalo $[U_K^1, U_K^2]$ asociado a la mencionada variación del paramétro."

Nuestros modelos serán DI, para los que sabemos que el AS multiparamétrico se muestra como una tarea muy difícil. Las listas *KBM2L* nos han servido en capítulos precedentes como estructuras de soporte de la explicación de alto nivel de las políticas óptimas de las TDO,

así como para la construcción y análisis de grandes TPC.

Ahora damos un paso más, pues extendemos el marco anterior y proponemos una metodología para realizar AS del modelo que origina las TDO. Como consecuencia, se revela la estrecha relación mencionada entre las tareas de AS y explicación. Mostramos que el AS y encontrar explicaciones son en cierto sentido un mismo problema. El AS clasifica cada posible escenario de atributos en estable/inestable (insensible/sensible) de acuerdo al valor de la respuesta, es decir, la decisión óptima, según la variación de los parámetros considerada. Más aún, nuestra metodología nos permite clasificar a los parámetros y ordenarlos como explicación de dicha estabilidad/inestabilidad. Un problema de índole clínica nos servirá, como en los capítulos precedentes de esta tesis, para ilustrar la teoría y obtener resultados de los algoritmos sobre los que extraer conclusiones.

A continuación, mostramos nuestra técnica de AS. Se trata de una aproximación computacional simbólica que se apoya en el uso de las listas *KBM2L* para explorar y describir el conjunto de modelos asociado al entorno del modelo MR.

6.2. Análisis de Sensibilidad Usando *KBM2L*

Vamos a extender las listas *KBM2L* para abordar el AS. Como hemos visto en esta tesis, una lista *KBM2L* procedente de la evaluación de un DI provee las políticas óptimas y explicaciones que utiliza el SAD. Antes de aceptar definitivamente el DI, los expertos involucrados en la construcción del SAD quieren estudiar si las salidas de sus modelos están determinadas fuertemente por los parámetros de entrada. Ésta es básicamente la idea del AS. Los parámetros de entrada son normalmente parámetros de probabilidad y utilidad, estimados subjetivamente bajo incertidumbre por los expertos. Así, la robustez o insensibilidad de un modelo que muestra sólo ligeras variaciones en sus resultados para substanciales perturbaciones en el valor de algún parámetro, reforzará la asignación subjetiva del experto relativa a estos parámetros como parte del modelo.

Los métodos estándar de AS incluyen métodos de proximidad a un umbral, AS probabilístico, AS basado en la entropía, identificación de alternativas no dominadas,... muchos de ellos originales de la literatura Bayesiana robusta (Ríos-Insua y F., 2000; Felli y Hazen, 1998). Pero el AS en grandes DI es muy complejo, debido a los múltiples parámetros a examinar. De hecho, la mayoría de los artículos en la literatura tratan problemas bastante simples, con pocos nodos, funciones de utilidad no muy complicadas y realizan un AS que involucra a lo sumo dos o tres parámetros simultáneamente.

Nuestro objetivo concierne a la llamada sensibilidad de la *decisión*, que se refiere al impacto de la variación paramétrica sobre las políticas óptimas identificadas por el modelo –*alternativas*

óptimas de referencia. Nuestro AS se centrará en la estabilidad de estas alternativas frente a la variación de un subconjunto de parámetros de interés. Como cada variación paramétrica o instanciación significa un DI con diferente información cuantitativa, todavía con idéntica estructura, tendremos un conjunto de modelos más o menos similares según el alcance del AS (local/global).

Nuestro punto de partida es el MR, con valores de los parámetros que los expertos juzgan como los más apropiados. La variación paramétrica se define mediante un intervalo, a veces llamado intervalo plausible (Coupé et al., 2000) que será discretizado para enumerar los modelos en la lista *KBM2L* (y calcular los desplazamientos). Típicamente, la variación se referirá a varios parámetros al mismo tiempo, es decir, un AS multiparamétrico, descubriendo posibles relaciones entre los parámetros. Cada modelo con su instanciación correspondiente de los parámetros es evaluado por el SAD. Las TDO resultantes (con s filas) forman una sola tabla (con $s \times m$ filas, si hay m modelos) haciendo simplemente que los parámetros jueguen el mismo papel que las variables. Así, la TDO final extendida, conducirá a una lista *KBM2L* extendida, *EKBM2L*, constituida por índices que contienen tanto parámetros como variables, y las alternativas óptimas de la evaluación de todos los DI. Observamos que la distinción entre parámetro y variable en el índice es de naturaleza semántica y la lista *EKBM2L* es formalmente igual a una lista *KBM2L*.

Consideremos a continuación dos proyecciones de la MM. La **proyección de la MM sobre las dimensiones de los parámetros** da lugar a una lista *KBM2L* de modelos, denotados con X. En dicha proyección podemos distinguir entre modelos iguales al MR (R) en su respuesta y distintos ($\neg R$) o podemos distinguir todos los posibles modelos ($\{X_1, X_2, \ldots\}$). La primera opción es más sencilla pues las listas presentan mayor granularidad y la adoptamos en nuestro AS. Nótese que si en la *EKBM2L* los parámetros tienen todos más peso que las variables de la *EKBM2L*, la lista es una secuencia de modelos, véase el Ejemplo 6.1.

Por otra parte, si las variables tienen todas mayor peso que los parámetros, entonces la lista *EKBM2L* es una secuencia de casos donde se considera la variación de todos los parámetros, véase el Ejemplo 6.1. La **proyección de la MM sobre las dimensiones de las variables** es una lista de casos denotados con x. En dicha proyección podemos distinguir entre casos de igual respuesta (caso estable α) a sus homólogos en el MR y de respuesta distinta (caso inestable β) o podemos distinguir todos los posibles valores de la respuesta ($\beta_{\mathbf{1}}, \beta_{\mathbf{2}}, \ldots$). Igualmente, la primera opción es más sencilla y la adoptamos en nuestro AS.

Dada una lista *EKBM2L*, a la lista proyección sobre las dimensiones de los parámetros la llamamos $E^{\downarrow PAR}$ y a la lista proyección sobre las dimensiones de las variables la llamamos $E^{\downarrow VAR}$.

Ejemplo 6.1 Sea la lista *EKBM2L* con dos parámetros binarios, p_1 y p_2, y dos variables binarias, A_1 y A_2. Cada parámetro toma dos valores numéricos que codificamos en la lista como 0 y 1. Por ejemplo, p_1 es 0.7 ó 0.8 y p_2 es 100 ó 120.

p_1	p_2	A_1	A_2			$E^{\downarrow PAR}$	A_1	A_2	p_1	p_2		$E^{\downarrow VAR}$
0	0	0	0	x_1	X	R	0	0	0	0	x_1	β
		0	1	x_2					0	1	x_1	
		1	0	x_3					1	0	x_2	
		1	1	x_4					1	1	x_1	
0	1	0	0	x_1	X	R	0	1	0	0	x_2	β
		0	1	x_2					0	1	x_2	
		1	0	x_3					1	0	x_1	
		1	1	x_4					1	1	x_3	
1	0	0	0	x_2	Y	$\neg R$	1	0	0	0	x_3	β
		0	1	x_1					0	1	x_3	
		1	0	x_3					1	0	x_3	
		1	1	x_4					1	1	x_2	
1	1	0	0	x_1	Z	$\neg R$	1	1	0	0	x_4	α
		0	1	x_3					0	1	x_4	
		1	0	x_2					1	0	x_4	
		1	1	x_4					1	1	x_4	

Las secuencias $X = \{x_1, x_2, x_3, x_4\}$, $Y = \{x_2, x_1, x_3, x_4\}$ y $Z = \{x_1, x_3, x_2, x_4\}$ representan modelos mientras que x_i representan respuestas de casos donde los parámetros varían. Supongamos que el MR es $p_1 = 0$ y $p_2 = 0$ con respuesta X. La proyección $E^{\downarrow PAR}$ es $\langle (0,1), R| \langle (1,1), \neg R|$, o bien $\langle (0,1), X| \langle (1,0), Y| \langle (1,1), Z|$ si distinguimos los modelos que se manifiestan al variar los parámetros. La proyección $E^{\downarrow VAR}$ es $\langle (1,0), \beta| \langle (1,1), \alpha|$ pues todos los casos cambian su valor de respuesta en los modelos que se derivan de la variación de los parámetros excepto si $A_1 = 1$ y $A_2 = 1$. □

El elevado número de parámetros en los problemas de tamaño real nos pone las cosas muy difíciles. Por ejemplo, 100 parámetros con 7 valores cada uno, conduce a 7^{100} modelos que deben ser evaluados, y cae sobre la tarea del AS la maldición de la dimensionalidad. La estrategia consiste en elegir un subconjunto de parámetros. Esto podría basarse en el consejo del experto acerca de las distribuciones de probabilidad más difíciles de asignar, o acerca de los parámetros más determinantes en el sistema. También, las restricciones en la variación de los parámetros, por ejemplo, combinaciones específicas de parámetros no permitidas, ayudan a disminuir el número de modelos a examinar. Otro criterio para restringir la variación de los parámetros puede ser

el $XBase$ óptimo en el conjunto de variables pues indica que los modelos son muy divergentes del MR. Nos interesa estudiar el entorno del MR, no cualquier modelo cuya evaluación sea completamente diferente de la obtenida del MR.

Si fuéramos capaces de evaluar el espacio de los parámetros completamente, la TDO extendida podría ser realmente dispersa (en términos de X) debido a la similitud entre algunos modelos; se puede esperar que muchos modelos produzcan el mismo conjunto de políticas óptimas, especialmente si se diferencian en parámetros robustos. Por tanto, deseamos encontrar la *Base* extendida (ahora con parámetros y variables) que muestre la mejor organización de la tabla bajo el criterio de *CRMS* mínimo en términos de los modelos X, porque una lista muy dispersa tiene la misma longitud, es decir, es óptima, en casi todas las *Bases* posibles (la parte fija de casi todos los ítems de respuesta conocida está formada por casi todos los parámetros). Es decir, la organización que buscamos se refiere a los modelos. Las relaciones entre los valores de los parámetros (presentes en el índice o el desplazamiento) y el comportamiento del SAD puede entonces analizarse de forma análoga a las relaciones entre variables y los casos del modelo.

Para hacer mucho más fácil la interpretación de los resultados del AS y el algoritmo que mostramos seguidamente, las *Bases* extendidas tendrán todos los parámetros a la izquierda y todas las variables a la derecha. La búsqueda de la *Base* óptima se realizará en el espacio de las posibles permutaciones entre parámetros a la izquierda y entre variables a la derecha, es decir, los parámetros nunca se mezclarán con las variables. Estudiaremos la lista *EKBM2L* y las proyecciones definidas $E^{\downarrow PAR}$ y $E^{\downarrow VAR}$. La interpretación del contenido de la lista *EKBM2L* es más sencilla si los parámetros tienen más peso que las variables pues los casos de un mismo modelo son adyacentes en todas las *Bases* mientras que de la otra forma, si las variables tienen más peso que los parámetros, vemos el comportamiento de los casos en el intervalo de variación de los parámetros y perdemos la visión de conjunto del modelo.

La idea es que los parámetros son relevantes (mucho peso, véase la ecuación (2.1)) porque cuantifican las relaciones estructurales del modelo. La lista *EKBM2L* óptima, con sus explicaciones, nos indicará esto y ordenará los parámetros con respecto a esta relevancia. Si algunos parámetros pierden peso entonces irán hacia la parte variable y saldrán de la explicación, y podríamos concluir que hay insensibilidad respecto de dichos parámetros. Si el parámetro es una probabilidad que define una distribución, podríamos juzgar tal vez que la variable aleatoria debe considerarse como determinística, y por consiguiente simplificaríamos el modelo. Podemos mejorar ciertas partes del modelo (DI) basándonos en estas observaciones.

Desafortunadamente, la evaluación del espacio completo de los parámetros se vuelve intratable en la práctica y la TDO extendida contiene alternativas óptimas desconocidas –o huecos–

para algunos índices. Si el modelo es muy complejo consideramos la *misma* instancia de variables en todo el análisis y un conjunto de instancias de los parámetros que da lugar a un conjunto de modelos. Hemos llamado X a la lista *KBM2L* resultado de un modelo. Podemos encontrar huecos entre dos modelos diferentes, que denominamos hueco de tipo XY, como $(\ldots, \langle i, X|\langle j, -1|\langle k, Y|, \ldots)$, donde -1 significa desconocido y X e Y discrepan en la respuesta de algún caso. Por otro lado, podemos identificar huecos de tipo XX, entre dos modelos iguales, desde el punto de vista del resultado de su evaluación.

Entonces proponemos el siguiente procedimiento para AS usando *EKBM2L* y análisis de los resultados.

0 Inicializamos la lista *EKBM2L* con la lista *KBM2L* óptima del MR.

1 Dado el espacio paramétrico discretizado de tamaño l, tomamos una muestra aleatoria de instanciaciones de tamaño $log(l)$ para ser evaluadas.

2 Añadimos los resultados a la lista *EKBM2L*. La optimizamos permutando parámetros (minimiza el número de huecos XY) y variables (minimiza la longitud de la lista) pero sin mezclar.

3 Mientras el tamaño de la lista *EKBM2L* no exceda un cota prefijada (por ejemplo $10000 * log(l)$ ítems), exploramos la lista para completarla, mediante la evaluación selectiva de nuevos modelos. También podemos limitar el tiempo de exploración y selección de nuevos modelos, especialmente si la evaluación del DI es muy costosa.

3.1 Para los huecos de tipo XY, evaluamos más modelos, tantos como $log(t)$, donde t es el tamaño del hueco. Generamos $log(t)$ modelos uniformente distribuidos en el hueco. Los evaluamos y actualizamos la lista *EKBM2L*.

3.2 Para los huecos de tipo XX, evaluamos sólo los modelos localizados en la mitad del hueco (el punto medio entre los desplazamientos extremos). Los evaluamos y actualizamos la lista *EKBM2L*.

3.3 Igual que el paso 2.

4 En huecos de tipo XX podemos inferir la respuesta de un ítem de la lista $E^{\downarrow PAR}$ en el que se unen los tres ítems $(\ldots, \langle i, X|\langle j, -1|\langle k, X|, \ldots) \rightarrow \ldots, \langle k, X|, \ldots$ si se mantiene el contexto del hueco, su naturaleza XX, en un número siginificativo de *Bases* próximas a la óptima, en el sentido de la pseudodistancia G_{H2}. En huecos de tipo XY podemos inferir

un ítem de la lista $E^{\downarrow PAR}$ en el que se unen los tres ítems $(\ldots, \langle i, X|\langle j, -1|\langle k, Y|, \ldots) \rightarrow \ldots, \langle k, X|\langle k, Y|, \ldots$ Las explicaciones de las listas *EKBM2L*, $E^{\downarrow PAR}$ y $E^{\downarrow VAR}$ son el AS.

Como vemos, los huecos XY requieren una exploración más exhaustiva. Sin embargo, los huecos XX son más fácilmente completados con el conocimiento inferido. La inferencia de la respuesta de los casos de un ítem hueco puede tener diferentes grados de acierto dependiendo del número de atributos que se pongan en juego para generar el entorno de la *Base* óptima: dos atributos son un entorno de 2 *Bases*, tres son 6 *Bases*, cuatro son 24 *Bases*,... Nótese que estos atributos son parámetros y siempre de máximo peso en la base óptima. El mecanismo de inferencia de valores desconocidos del paso **4** tiene un error que depende del número de modelos que recoge la lista en el bucle del paso **3** y de la *Base* obtenida en el paso **3.3**. El error es menor cuantos más modelos hay en la lista y éstos están optimizados en la *Base* óptima del MR.

Es importante darse cuenta de que no se deben inferir modelos en la lista *EKBM2L*, pues (salvo que sean vacíos, con respuesta constante) en un hueco puede dar lugar a un número excesivo de ítems. La razón es que el MR puede estar en las *Bases* óptimas, y tener pocos ítems, pero los otros modelos, en general no.

Por otra parte, se pueden inferir ítems de la lista $E^{\downarrow VAR}$ si no se han realizado evaluaciones completas del DI pero al proyectar eliminamos los parámetros que son el objeto del análisis. Nuestra inferencia no pretende resolver la evaluación del diagrama. Sin embargo, en la sección 6.3 se propone etiquetar la lista *EKBM2L* para mostrar las diferencias entre los modelos y el MR y la lista $E^{\downarrow VAR}$ sí es sometida a inferencia en sus huecos. En este modo se trata de reflejar la relación entre los parámetros y las variables mostrando los casos que no cambian nunca.

La explicación con más probabilidad de que la inferencia sea correcta es la que tiene mayor número de atributos, más larga. Por eso, los huecos XY son diseccionados más y los ítems correspondientes a los nuevos huecos tienen menos casos y explicaciones más largas. En cierto modo los pasos 3.1 y 3.2 son similares a la copia parcial, véase el capítulo 3. La idea es considerar que tenemos toda la lista *EKBM2L* y la copiamos en la misma *Base*. Pero en lugar de tomar la información de la lista que desconocemos, la tomamos de la evaluación del modelo.

Si analizamos la *Base* de la lista *EKBM2L*, distinguimos el orden de los parámetros y el orden de las variables. El primero pretende minimizar los huecos XY facilitando la inferencia de modelos y la explicación de la sensibilidad. El segundo trata de minimizar la longitud de las secuencias de casos, agrupados en ítems, que constituyen los modelos, ofreciendo explicaciones eficientes del comportamiento de cada modelo. Pensamos que lo mejor para realizar el análisis de la lista *EKBM2L* es tomar como orden de las variables el de la *Base* óptima del MR. Como

en general, los demás modelos tienen otra *Base* óptima diferente, la lista *EKBM2L* puede ser muy compleja (larga) pues muchos modelos no están óptimizados. Aquí radica la inoportunidad de inferir modelos en la lista *EKBM2L*. Pero precisamente la diversidad de las *Bases* óptimas de los modelos es un indicador de sensibilidad propio de las listas que introduce esta tesis. Por consiguiente, podemos considerar una distancia entre modelos a partir de la pseudodistancia G, véase la sección 3.4, entre las *Bases* óptimas correspondientes.

Desde este punto de vista, la sensibilidad del modelo respecto de los parámetros se manifiesta en cambios en la decisión y en la *Base* óptima de cada modelo comparada con la del MR. El último aspecto implica que la variación paramétrica redunda en posibles cambios en la explicación de la respuesta y en la relevancia/importancia de las variables. Ambas manifestaciones de sensibilidad esán relacionadas. Si hay cambios de decisión entonces también hay cambios en las explicaciones. Pero si los cambios provocan un $XBase$ con G "grande" respecto de la óptima del MR y la variación paramétrica nos parece razonable, tendremos que pensar que el modelo MR tiene un comportamiento muy inestable y crítico respecto de los parámetros. En la línea de la propuesta de (Nielsen y Jensen, 2003) podemos obtener una esfera (o elipsoide) multidimensional de seguridad en el espacio de parámetros donde no cambia el MR y otras de mayor diámetro donde además cambia la *Base* con valores crecientes de G. Un diámetro de gran tamaño corresponde a un parámetro estable mientras que el diámetro pequeño corresponde a un parámetro más sensible. Observamos que podemos caracterizar la variación paramétrica en el sentido de indicar que en ciertos modelos, fuera de la esfera de G moderado, los nuevos parámetros alteran la semántica del DI y las variables pierden su significado inicial.

6.3. Estrategias de Análisis de Sensibilidad

Incluso con la clase de muestreo e inferencia indicados en el algoritmo, es probable que no podamos llenar suficientemente la lista *EKBM2L* a causa del coste computacional extremadamente alto que se induce. Por consiguiente, los ítems obtenidos incluirán unos pocos casos en relación a la enorme *EKBM2L*, dando como resultado que no dispogamos de explicaciones útiles: casi todos los parámetros y variables constituirán la explicación de las políticas óptimas pues la parte variable del índice será casi siempre vacía.

Así, en la práctica, el paso 4 anterior se llevará a cabo como sigue. Nos centramos en subproblemas del MR que podemos muestrear con un esfuerzo razonable. En primer lugar, exploramos la sensibilidad de los modelos de la lista *EKBM2L* mediante indicadores del alcance del cambio de decisión, es decir, el número de casos sensibles a la variación paramétrica. En segundo lugar, estudiamos la lista óptima respecto del mínimo *CRMS* observando la *Base*. A continuación

detallamos los dos pasos.

1. El análisis de la lista *EKBM2L* consiste en explorar las diferencias entre cada modelo y el MR. Podemos plantear básicamente la distancia entre pares de modelos utilizando el valor numérico de los parámetros asociados. Consideramos una cota superior D_{max} de la distancia usual L_1 (suma de valores absolutos diferencias entre componentes del vector de parámetros) entre modelos en el espacio de los parámetros, en el entorno de variación paramétrica. Así, obtenemos una lista con el esquema de la lista original donde hemos etiquetado con ***RSKB*** (Respuesta Sensible conocida) los casos cuya respuesta es diferente de la mostrada en el MR. De este modo, si hay r parámetros y analizamos $\mathcal{P}$ ($\mathcal{P} \leq r$), éstos toman todos los valores de su intervalo de variación libremente. Al resto solamente se les permite tomar valores que no excedan la D_{max} de interés. Así, inicialmente con la $D_{max} = 0$ obtenemos un indicador de sensibilidad para el conjunto de los $\mathcal{P}$ parámetros en exclusiva. Incrementando D_{max} introducimos el efecto de la variación del resto de parámetros que pueden estar más o menos acoplados con el conjunto. Seleccionando subconjuntos del conjunto de parámetros podemos obtener información del papel que desempeñan en la variación del resultado del modelo y las interrelaciones con otros subconjuntos.

   Un aspecto destacable del análisis es la **normalización** del conjunto de parámetros. Es importante señalar que la distancia entre modelos debe estar normalizada pues puede haber grandes diferencias en los órdenes de magnitud de los parámetros. Hemos tomado como coordenadas normalizadas para calcular la distancia L_1 entre dos modelos el valor numérico, veáse la sección 6.4, dividido por la amplitud del intervalo de variación del parámetro. Sin normalizar la escala del parámetro, se produce un sesgo en el indicador de sensibilidad. En la sección 6.4 el modelo MR es $p_1 = 7$ con el valor 0.60 y $u_2 = 7$ con el valor 100 y otro modelo es $p_1 = 14$ con el valor 0.84 y $u_2 = 11$ con el valor 120. La distancia sin normalizar es 0.84-0.60 + 120-100 = 20.24. La distancia normalizada es $\frac{0,84-0,60}{0,1} + \frac{120-100}{20} = 2.4 + 1.0 = 3.4$. La última, al estar normalizada, no se ve afectada por las escalas de los valores.

   Interpretamos la sensibilidad en los entornos de radio D_{max} del modo siguiente:

   a) Si D_{max} es "pequeña" muestra la sensibilidad específica de la combinación o grupo de parámetros $\mathcal{P}$.

   b) Si D_{max} toma un valor medio o central muestra la sensibilidad de las combinaciones en interacción con el resto de parámetros.

c) Por último, si D_{max} es "grande" muestra la sensibilidad del conjunto completo de parámetros del esquema.

Sea $[KB]$ el número de casos conocidos en la lista. La proporción $[RSKB]/[KB]$ es un indicador de sensibilidad de la decisión (en términos del número de cambios) al conjunto de parámetros.

La **proporción absoluta** $[RSKB]/[KB]$ puede ser comparada con la **proporción relativa** $[RSKB]/[KB(D_{max})]$ que tiene en cuenta el número de casos conocidos y sus cambios pero sólo en los modelos a distancia menor o igual a D_{max} del MR. Al variar D_{max}, los puntos (D_{max}, $[RSKB]/[KB(D_{max})]$) determinan la que hemos llamado **curva de sensibilidad relativa** del subconjunto de parámetros $\mathcal{P}$. La forma de la curva caracteriza la sensibilidad de los posibles subconjuntos.

Un aspecto importante del AS multiparamétrico es simplificar el conjunto en la medida de lo posible. Una estrategia a seguir es estudiar la sensibilidad explorando la lista primero con todos los parámetros $\mathcal{P}$ de interés variando en todo su rango. Después dejaremos un parámetro cada vez en su valor de referencia y el resto variará conjuntamente. Nótese que un análisis uniparamétrico hace lo contrario, es decir, varía un parámetro y el resto queda fijo en su valor de referencia. Si $[RSKB]/[KB(D_{max})]$ es similar cuando varían los $\mathcal{P}$ y los $\mathcal{P}$ menos el parámetro, el resultado no es sensible a este parámetro y podemos eliminarlo o tal vez plantear ampliar su intervalo de variación.

El mismo análisis se puede realizar sobre las listas $E^{\downarrow PAR}$ y $E^{\downarrow VAR}$, que son mucho más simples que la lista *EKBM2L*. Con la lista $E^{\downarrow PAR}$ analizamos los modelos y con la lista $E^{\downarrow VAR}$ analizamos prototipos de los casos.

2. A partir de la lista *EKBM2L* etiquetada, optimizamos las proyecciones respecto del *CMRS*, que es igual al número de huecos XY en la lista $E^{\downarrow PAR}$ y al número de huecos xy en la lista $E^{\downarrow VAR}$. Ahora podemos inferir ítems de respuesta X (x) que unen los modelos (casos) de hueco XX (xx) y los modelos (casos) adyacentes e inferir ítems de respuesta Y (y) que unen los modelos (casos) de hueco XY (xy) y los modelos (casos) adyacentes. Las explicaciones de los ítems en $E^{\downarrow PAR}$ con respuesta *RSKB* es el AS del modelo respecto del conjunto de parámetros de interés. Las explicaciones de los ítems en $E^{\downarrow VAR}$ con respuesta *RSKB* constituyen el AS del modelo respecto de los atributos e indica qué casos son sensibles a la variación de los parámetros y sugiere las instancias de interés para el análisis de subproblemas.

6.4. Ejemplo: un problema médico

El siguiente diagrama de la Figura 6.1 nos sirve para ilustrar las anteriores secciones del capítulo. Muestra el DI acerca de la práctica de cirugía para el injerto de un "by–pass" de la arteria coronaria, adaptado de (Horvitz et al., 1988).

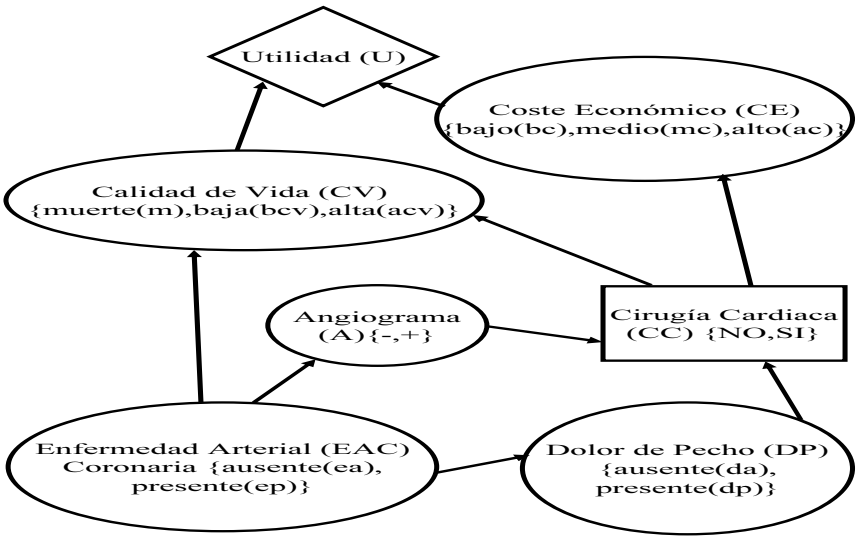

Figura 6.1: Diagrama de influencia para enfermedad arterial coronaria

Antes de tomar la decisión de *operar*, el médico explora al paciente para comprobar los posibles síntomas como *dolor en el pecho* causado por un esfuerzo particular. También lleva a cabo un *angiograma*, una técnica de diagnóstico por imagen de las arterias, para disminuir la incertidumbre acerca de la presencia de una enfermedad arterial coronaria en el paciente. Hay un riesgo de infarto de miocardio que mide la *calidad de vida* del paciente, la cual depende de la presencia de la *enfermedad* y de la decisión de operar (porque hay riesgo de muerte asociado a la propia cirugía). La preferencias del decisor se definen conjuntamente mediante la calidad de vida y *el coste económico.*

Además mostramos los dominios de cada variable en la Figura 6.1. El DI (o MR) tiene 28 entradas de probabilidad y 9 valores de utilidad, un tamaño adecuado para ilustrar nuestras ideas. Después de resolver el DI, las alternativas óptimas de referencia son función de $\{A, DP\}$. La *Base* óptima de la TDO (asociada a CC) es $[A, DP]$ y su *KBM2L* en notación de índice es $\langle(negativo, presente), NO|\langle(positivo, presente), SI|$, donde hemos enumerado los diferentes dominios de valores en el orden dado en la Figura 6.1. La *Base* es muy sencilla y G no permite discriminar más que 0 y 1, es decir, $G([A, DP], [A, DP]) = 0$ y $G([A, DP], [DP, A]) = 1$. Por

tanto, según las explicaciones del segundo ítem, vemos que la recomendación es practicar cirugía sólo si el test del angiograma es positivo.

Para el AS elegimos el conjunto de parámetros $\mathcal{P}$ mostrado en el Cuadro 6.1, donde delimitamos sus variaciones plausibles.

Cuadro 6.1: Parámetros de interés para el AS

Parám. en $\mathcal{P}$	Valor–Ref.	Intervalo Plaus.
$p_1 = P(CV = acv \mid CC = SI, EAC = ep)$	0.60	[0.46,0.84]
$p_2 = P(CV = bcv \mid CC = NO, EAC = ep)$	0.30	[0.16,0.44]
$u_1 = U(CV = acv, CE = ac)$	180	[145,215]
$u_2 = U(CV = m, CE = bc)$	100	[65,135]
$u_3 = U(CV = m, CE = ac)$	70	[35,105]
Parám. en $\mathcal{P}$	Valor–Ref–Normaliz.	Intervalo Plaus.Normaliz.
$p_1 = P(CV = acv \mid CC = SI, EAC = ep)$	1.58	[1.21,2.21]
$p_2 = P(CV = bcv \mid CC = NO, EAC = ep)$	1.07	[0,57,1.57]
$u_1 = U(CV = acv, CE = ac)$	2.57	[2.07,3.07]
$u_2 = U(CV = m, CE = bc)$	1.43	[0.93,1.93]
$u_3 = U(CV = m, CE = ac)$	1.0	[0.50,1.50]

Estos parámetros involucran probabilidades y utilidades (nótese la escala diferente de ambos) y se refieren a la calidad de vida, una importante consecuencia del problema sobre la que existe mucha incertidumbre. Aquí, consideramos más interesante estudiar estos parámetros en lugar de otros cuya asignación es más estable o duradera como la prevalencia de la enfermedad o el resultado del test de angiograma. En efecto, inicialmente tomamos otros parámetros y los descartamos debido a su escaso efecto.

La variación de los parámetros de probabilidad implica la variación del resto de probabilidades dentro de la misma distribución. El reparto de la masa de probabilidad se lleva a cabo proporcionalmente. Como mencionamos anteriormente, el espacio paramétrico no puede ser evaluado completamente. Tenemos 5 parámetros con 15 valores cada uno (debido a la discretización) que conducen a 15^5 que deben ser evaluados, y nos encontramos con el problema de la dimensionalidad. Siguiendo el algoritmo propuesto, evaluamos 4588 modelos tras el MR en 10 minutos en un ordenador Pentium^{TM} IV a 3.21-GHz y 2 GB. Se produce una lista *EKBM2L* con 18352 casos conocidos (de un total de 3037500), 4045 de ellos etiquetados como $RSBK$, es decir, el

22.04 % de los casos cambian alguna decisión dada en el MR.

La Figura 6.2 muestra el gráfico del espectro de la lista. Cada modelo es una banda de cuatro casos. Podemos observar fácilmente si hay (color azul) o no cambio respecto al MR. La alternativa de $CC = SI$ se muestra en amarillo y $CC = NO$ en cian. Los modelos más inestables son los primeros.

La Figura 6.3 muestra las curvas de sensibilidad relativa cuando $\mathcal{P}$ es $\{p_1\}$, $\{p_2\}$, $\{u_1\}$, $\{u_2\}$ y $\{u_3\}$, respectivamente, donde $\{p_1\}$ y $\{p_2\}$ son parámetros de probabilidad en $P(CV|CC, EAC)$ y $\{u_1\}$, $\{u_2\}$ y $\{u_3\}$ son parámetros de utilidad.

Observamos que el parámetro u_1 es el que mayor sensibilidad (AS uniparamétrico) manifiesta y u_3 el que menos. Los primeros puntos de la curva de $\{p_1\}$ muestran una alta sensibilidad (AS uniparamétrico), es decir, cuando $\{p_1\}$ varía desde el mínimo valor de su rango hasta el máximo, con todos los demás parámetros manteniendo sus valores de referencia. Esto también se observa para la curva de $\{u_1\}$.

Las curvas de $\{u_1\}$, $\{p_1\}$, y $\{p_2\}$ muestran una interacción grande en el intervalo [0.1,1.0] de D_{max}, donde el movimiento del resto de respectivos parámetros causa un descenso del índice de sensibilidad $[RSKB]/[KB(D_{max})]$. Nuestro problema médico es sensible a la mayoría de las variaciones conjuntas de ciertos parámetros. Esto revela relaciones entre los parámetros. Pero es poco sensible a la variación individual de parámetros como $\{u_2\}$ y $\{u_3\}$. La curva $\{u_2\}$ (y la $\{u_3\}$ también) tiene más pendiente e indica que es su interacción con los demás parámetros la que causa cambios en las decisiones óptimas de referencia.

De este modo, vemos que las interacciones entre parámetros son importantes. Conforme D_{max} aumenta, los otros parámetros también comienzan a cambiar gradual y simultáneamente, y podemos observar su influencia conjunta sobre la sensibilidad.

Claramente, los últimos puntos, cuando todos los parámetros cambian libremente al tiempo, coinciden en todas las curvas. La sensibilidad relativa conjunta converge hacia la distancia normalizada 1.5 en $[RSKB]/[KB(D_{max})] = 0.2204$. A través de D_{max} podemos medir la máxima perturbación requerida por un parámetro para mostrar una sensibilidad dada de interés.

De acuerdo con estos resultados, algunas partes del DI pueden ser mejoradas, como herramientas de validación y verificación. Por ejemplo, podríamos hacer un esfuerzo en la asignación de $\{u_1\}$, $\{p_1\}$ y $\{p_2\}$ debido a su sensibilidad.

La Figura 6.4 muestra las curvas de sensibilidad relativa cuando $\mathcal{P}$ es $\{p_1, p_2\}$ y $\{u_1, u_2, u_3\}$, respectivamente. Nótese que la variación simultánea de los parámetros de probabilidad con los parámetros de utilidad en sus valores de referencia muestra una sensibilidad alta. Por el contrario, la variación simultanea de los parámetros de utilidad con los parámetros de probabilidad en sus

valores de referencia muestra una sensibilidad baja.

La lista $E^{\downarrow PAR}$, contiene 4532 modelos: 3995 etiquetados con [RSKB] y 537 iguales al modelo de referencia. Por tanto hay un 88.15 % de sensibilidad a la variación paramétrica del experimento. En la *Base* inicial $[p_1, p_2, u_1, u_2, u_3]$ y se presentan 3757 huecos XX y 774 huecos XY. Optimizamos en la *Base* $[p_2, p_1, u_1, u_3, u_2]$ obteniendo 3717 huecos XX y 610 huecos XY. La *Base* que sugiere la Figura 6.3 es $[p_2, p_1, u_3, u_1, u_2]$, donde se obtienen 3713 huecos XX y 614 huecos XY. Ambas son similares y reflejan la importancia de los parámetros en la lista $E^{\downarrow PAR}$.

Los parámetros se ordenan respecto de su relevancia y acoplamiento en términos de sensibilidad. El espectro correspondiente lo muestra la Figura 6.5.

No mostramos la lista $E^{\downarrow VAR}$ porque todas las alternativas óptimas de sus casos discrepan de las del MR, es decir, no hay perfiles que mantengan las alternativas óptimas de referencia. Deberíamos considerar un intervalo de variación de los parámetros más pequeño que mostrase la frontera de sensibilidad desde el punto de vista de la lista $E^{\downarrow VAR}$.

El AS basado en las listas *KBM2L* que hemos introducido puede ser juzgado como una técnica complementaria a los actuales métodos. Ésta desarrolla un metamodelo de exploración, donde los parámetros de probabilidad y utilidad desempeñan un papel de variables convencionales. Cada variación de los parámetros puede interpretarse como un modelo diferente, el cual tras ser evaluado permite conocer parcialmente la *EKBM2L*. En teoría, un algoritmo basado en un procedimiento de muestreo e inferencia se enfenta al AS con un elevado número de parámetros. En la práctica, estudiamos proyecciones de la *EKBM2L* que ayudan a obtener conclusiones centradas en AS específicos, perfiles de interés y clasificación de los parámetros con respecto a su relevancia en términos de sensibilidad.

Finalmente, de cara al AS puede ser interesante almacenar como respuesta el vector de las alternativas ordenado de forma decreciente según la utilidad esperada y el propio vector de utilidades esperadas. La ventaja es clara, pues ahora disponemos de mucha información respecto de la decisión óptima, pero el coste computacional se incrementa en varios órdenes de magnitud.

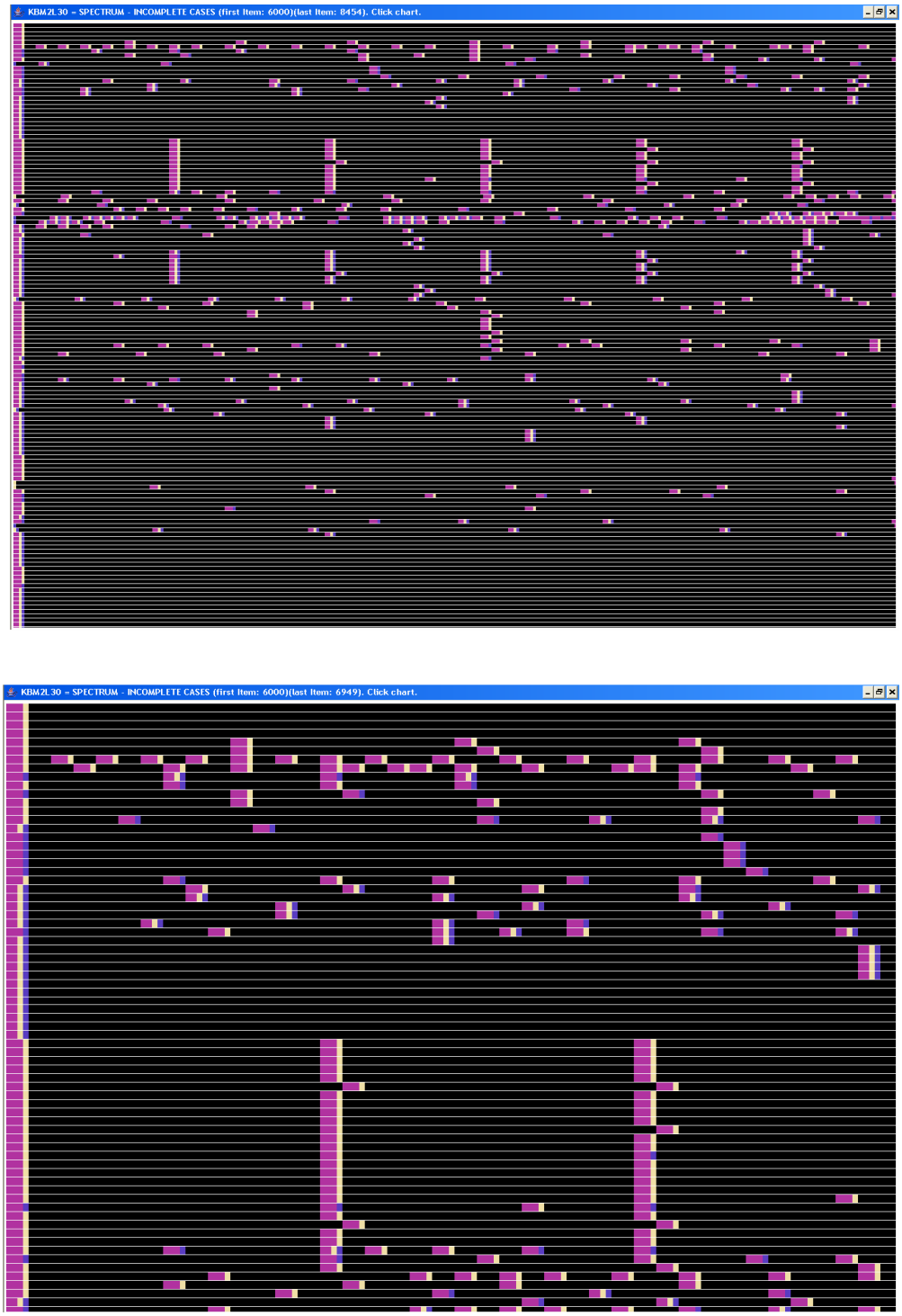

Figura 6.2: Espectro de la lista *EKBM2L*

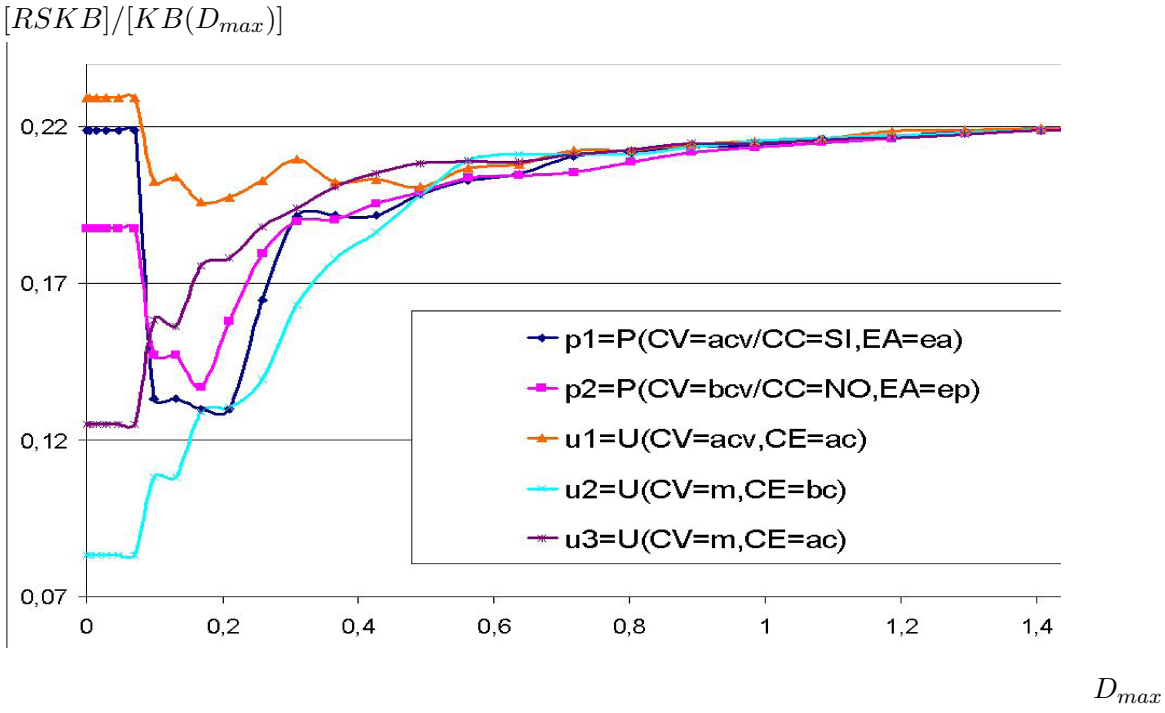

Figura 6.3: Cuvas de sensibilidad relativa para $\{p_i\}, i = 1, 2$ y $\{u_i\}, i = 1, 2, 3$

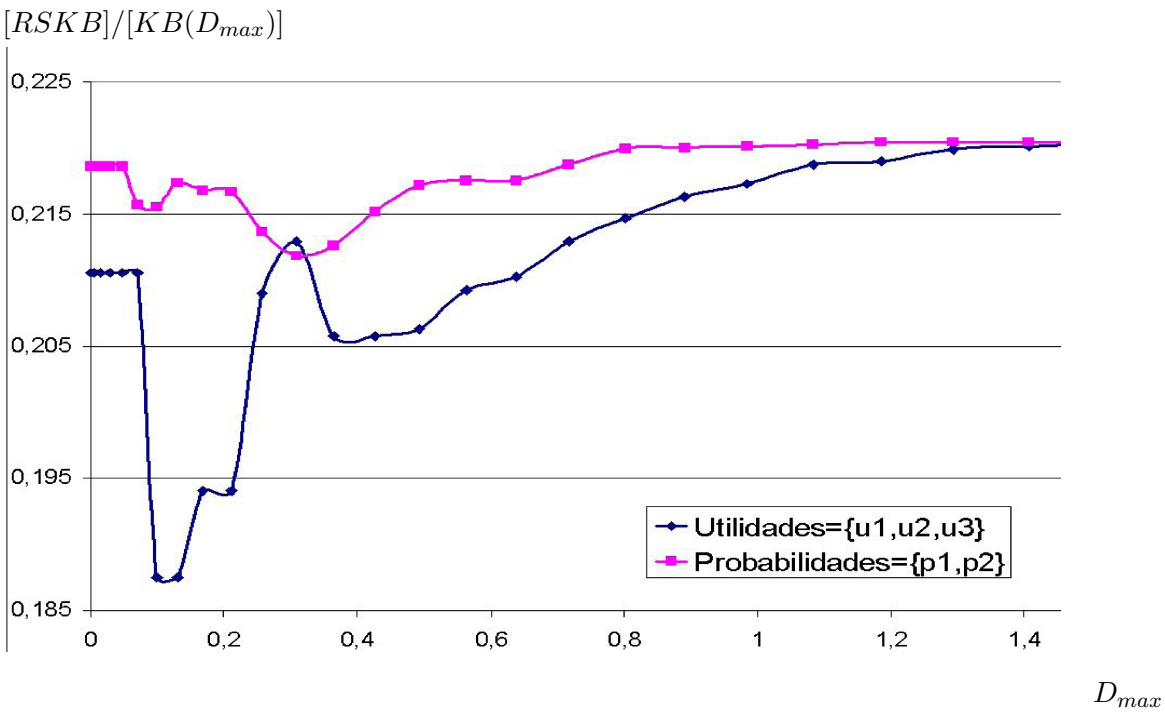

Figura 6.4: Curvas de sensibilidad relativa para $\{p_1, p_2\}$ y $\{u_1, u_2, u_3\}$

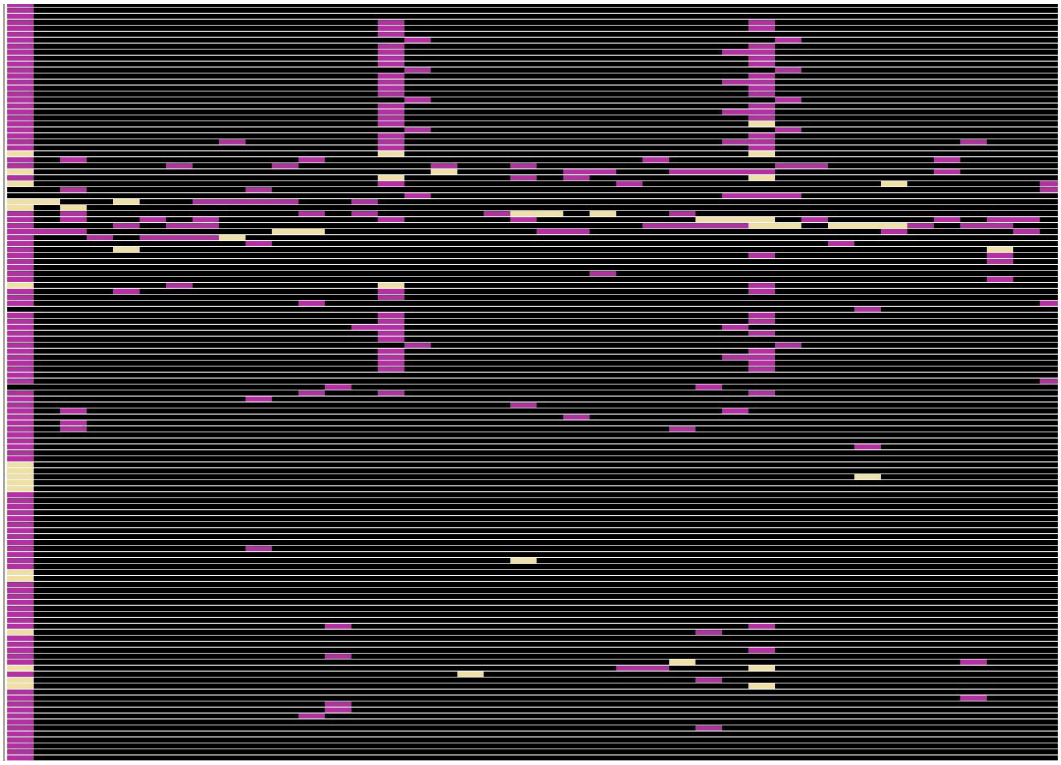

Figura 6.5: Espectro de la lista $E^{\downarrow PAR}$

Capítulo 7

Conclusiones y Futuras Líneas de Investigación

El capítulo 7 expone las conclusiones de esta tesis y describe algunas de las posibles líneas futuras de investigación por desarrollar.

Los fundamentos de la representación del conocimiento mediante listas *KBM2L* fueron expuestos en el capítulo 2. Los tres temas clave fueron: el almacenamiento eficiente de grandes tablas, la extracción de explicaciones y la representación gráfica de la lista *KBM2L*.

La optimización de la estructura *KBM2L* fue abordada en el capítulo 3. La búsqueda de la solución se sirvió de diversas heurísticas y de dos algoritmos de optimización combinatoria. A lo largo de la tesis hemos propuesto tres posibles criterios de optimización: global respecto del número de ítems de la lista, sobre los ítems de una familia (una modalidad de la respuesta) y sobre el número de *Cambios de Modalidad de la Respuesta en la Secuencia* de ítems. Dependiendo del objetivo y de las características de la lista procede centrarse en uno u otro.

El capítulo 4 recogió el problema de la combinatoria del almacenamiento en las *Tablas de Decisiones Óptimas* y propuso las listas *KBM2L* como solución eficiente para su gestión y para la extracción de explicaciones. La formulación y resolución de consultas a las *Bases de Conocimiento* de los *Sistemas de Ayuda a la Decisión* fueron modelizadas mediante un diálogo de agentes apoyado en las listas *KBM2L*.

La representación y construcción de grandes *Tablas de Probabilidad Condicionada* fue planteada en el capítulo 5. La propuesta presentaba dos alternativas y un protocolo de educción, validación y extracción de conocimiento.

El *Análisis de Sensibilidad* del modelo es una línea que de forma natural surge al extender el esquema de los resultados con atributos que son parámetros del propio modelo. Podemos

describir la tabla del resultado del modelo que corresponde a la evaluación de la solución y simultáneamente a la variación de ciertos parámetros. El capítulo 6 mostró el planteamiento general.

Terminamos esta tesis con las futuras líneas y temas abiertos. Distinguimos dos ámbitos: los fundamentos y las aplicaciones de las listas *KBM2L*. Los fundamentos tratan de generalizar la representación de conocimiento para hacerla más expresiva, amplia y eficiente. Las aplicaciones están relacionadas con el estado de la cuestión en áreas como: Sistemas de Ayuda a la Decisión, Aprendizaje Automático, Análisis Multidimensional de Datos y Minería de Datos.

7.1. Conclusiones

En esta tesis hemos propuesto una estructura de información equivalente a un producto cartesiano discreto que optimiza el almacenamiento de Matrices Multidimensionales y tablas de datos que encontramos como Bases de Conocimiento en los Sistemas de Ayuda la Decisión. Sin la posibilidad de resumir y/o agregar los resultados, dichos Sistemas de Ayuda a la Decisión son prácticos para el análisis de decisiones y la modelización asociada, pero el modelo es computacionalmente imposible de resolver de forma exacta.

Bajo las hipótesis de representatividad discreta y granularidad de la información, el planteamiento propuesto presenta grandes ventajas. En conjunción con la evaluación parcial de modelos (instanciación de casos), permite trabajar con Sistemas de Ayuda a la Decisión complejos. Los sistemas aprenden de forma incremental y escalable y sintetizan, generalizan, explican e infieren en el contexto de un modelo evaluado. El punto de partida es la observación del conocimiento relativo a Sistemas de Ayuda a la Decisión en las Tablas de Decisiones Óptimas y en las Tablas de Probabilidad Condicionada. Éstas son accedidas y gestionadas como Matrices Multidimensionales, cuyas estructuras de datos son memorizadas secuencialmente por los sistemas informáticos convencionales como listas.

Hagamos un recorrido por cada capítulo de la tesis:

1. La introducción muestra el contexto de la investigación desarrollada en la tesis centrándose en las líneas de interés relativas a los Sistemas de Ayuda a la Decisión. Por otra parte, trata de situar los problemas que pretenden abordar el Aprendizaje Automático, la Estadística, el Descubrimiento en Bases de Datos, la Minería de Datos, la Lógica, los Conjuntos Aproximados, las Redes de Neuronas Artificiales,... destacando su relación con esta tesis.

2. En el capítulo 2, en primer lugar, presentamos una descripción completa de la lista *KBM2L* en las secciones 2.1 y 2.2. En la sección 2.3 desarrollamos el proceso de su construcción.

En la sección 2.4 mostramos el índice, la unión de ítems de la lista mediante *cambios de Base* y la explicación. En la sección 2.5 visualizamos los elementos de la lista mediante el espectro. Finalmente, en la sección 2.6 revelamos cómo implementar la lista *KBM2L*.

3. La búsqueda de listas óptimas la recoge el capítulo 3. En la sección 3.1 definimos el problema de optimización asociado a la obtención de la lista *KBM2L* que sintetiza mejor el conocimiento. En la sección 3.2 establecemos la conexión entre la complejidad de la lista *KBM2L* y la *Base* del desplazamiento. En la sección 3.3 caracterizamos el espacio y el proceso de búsqueda de la *Base* óptima. Las secciones 3.4 y 3.5 desarrollan dos estrategias para realizar la búsqueda de la solución. Primeramente, un algoritmo de entorno variable en la métrica definida entre las *Bases*. En segundo lugar, un algoritmo genético que utiliza una estimación de *Cambios de Modalidad de la Respuesta en la Secuencia* de ítems como función de evaluación de la bondad de las *Bases* y un operador de cruce basado en la votación ponderada de las posiciones de los atributos en la *Base* de la siguiente generación. En la sección 3.6 ilustramos todo el capítulo con un conjunto de ejemplos. Finalmente, en la sección 3.7 ilustramos otras estructuras afines en el área del Apredizaje Automático estableciendo una comparación con nuestra propuesta.

4. En el capítulo 4 aplicamos la síntesis de conocimiento mediante listas *KBM2L* a las Tablas de Decisiones Óptimas. El uso de nuestra metodología está completamente justificado a causa de las innumerables dificultades computacionales (Bielza et al., 2000; Cooper, 1990), asociadas a la evaluación de los Modelos Gráficos Probabilistas. En efecto, en la literatura reciente de Diagramas de Influencia, hay un creciente interés en determinar las variables que son relevantes para cada decisión, pues el algoritmo estándar para resolver Diagramas de Influencia tiende a construir funciones de la política óptima con dominios demasiado extensos (Vomlelová y Jensen, 2004). Nuestra aproximación se realiza después de la evaluación del diagrama, a diferencia de otras aproximaciones que determinan estas variables antes, mediante el análisis de la estructura del diagrama, véanse (Fagiouli y Zaffalon, 1998; Nielsen y Jensen, 1999; Lauritzen y Nilsson, 2001), entre otros.

   La granularidad de las Tablas de Decisiones Óptimas no sólo depende del problema, sino también de la organización interna de la tablas. Como hemos explicado en teoría y mostramos en los sistemas *IctNeo* y *PGNHL*, una reorganización puede conducir a la agrupación de casos en contextos de idéntica propuesta. Una buena organización reduce los requisitos de memoria para almacenar las Tablas de Decisiones Óptimas y extrae información cualitativa de las variables: sólo algunas variables son realmente relevantes en ciertos contextos.

Por consiguiente, descubrimos relaciones entre atributos que pueden explicar las propuestas del Sistema de Ayuda a la Decisión. En la sección 4.2 usamos el Sistema de Ayuda a la Decisión *PGNHL* para mostrar resultados y detalles del procedimiento.

Una de las más importantes características de un Sistema de Ayuda a la Decisión es responder a las consultas del usuario. En la sección 4.3 proponemos un sistema de consultas basado en el marco de las listas *KBM2L*. La información es eficientemente accedida y recuperada para construir la respuesta. Las *Bases* óptima y operativa permiten que los registros involucrados en la consulta puedan organizarse bajo diferentes perspectivas. Las consultas generales o abiertas conducen a respuestas imprecisas que son manejadas vía un proceso de aprendizaje de las relaciones entre atributos/política, donde la interacción con el experto es necesaria para alcanzar una respuesta satisfactoria, con baja incertidumbre. Nuestra aproximación proporciona una explotación clara de la Base de Conocimiento para el Sistema de Ayuda la Decisión, puesto que conjuga las consultas, respuestas y explicaciones de forma flexible y simplicando el proceso de ayuda.

En la sección 4.4 desarrollamos un ejemplo completo de consulta abierta para el Sistema de Ayuda a la Decisión *IctNeo*. En el caso de *IctNeo*, la carga computacional no es excesiva en tanto que usamos algunas estrategias (Gómez, 2002). Por ejemplo, dividimos el proceso de evaluación mediante la instanciación de la evidencia (Ezawa, 1998), eligiendo algunos atributos para conseguir un tamaño manejable de las Tablas de Decisiones Óptimas y producir un impacto pequeño en la propagación de la incertidumbre.

5. El aspecto central del capítulo 5 es el intento de desarrollar una extensión de la representación de la lista *KBM2L* sobre las Tablas de Probabilidad Condicionada. Una Tabla de Probabilidad Condicionada representa relaciones probabilistas en Modelos Gráficos Probabilistas y es difícil de construir cuando tenemos alta dimensionalidad. Proponemos dos aproximaciones diferentes –probabilidad y valores lógicos– según el papel que desempeña la respuesta de la lista en la Tabla de Probabilidad Condicionada asociada. Describimos la sintaxis y semántica o interpretación, el almacenamiento computacional y las operaciones de coalescencia en Tablas de Probabilidad Condicionada grandes. También usamos la lista *KBM2L* para ayudar a la educción de probabilidades imprecisas mediante intervalos. Realizamos algún esfuerzo en el sentido de capturar ciertas regularidades, es decir, independencias que se manifiestan tan sólo en ciertos contextos de dichas tablas de probabilidad, las llamadas independencias específicas del contexto (Boutilier et al., 1996).

   Planteamos algunas aplicaciones: para realizar la educción de probabilidades, para validar

el conocimiento y para extraer el conocimiento. En concreto, proponemos un protocolo de asignación de probabilidades, ilustrado mediante la construcción de la lista *KBM2L* de dos Tablas de Probabilidad Condicionada de los Sistemas de Ayuda a la Decisión *IctNeo* y *PGNHL*. La propuesta tiene en cuenta restricciones del dominio y reglas expertas. Permite trabajar mezclando conocimiento experto y bases de datos. La técnica mantiene la consistencia.

6. En el capítulo 6 planteamos una interesante aplicación de nuestra técnica al Análisis de Sensibilidad de modelos de decisión. La obtención de las reglas explícitas de toma de decisiones y la observación de la influencia de la variación de los parámetros en dichas reglas implica casualmente un procedimiento práctico para la realización de Análisis de Sensibilidad. Podemos considerar algunos parámetros (probabilidades, preferencias, dependencias,...) como atributos y evaluar los modelos resultantes usando diferentes valores. Los datos de la evaluación pueden organizarse en una lista *KBM2L* extendida (*EKBM2L*) y las relaciones entre los valores de los parámetros pueden analizarse desde el punto de vista del comportamiento del Sisitema de Ayuda a la Decisión. La lista *KBM2L* resultante revela si cada decisión es robusta o no cuando varía cada parámetro, dependiendo de su peso, en cierto rango. Un peso alto supone una gran sensibilidad.

   Las listas *KBM2L* extendidas dan una visión muy general de la evaluación del modelo, donde los parámetros de probabilidad y utilidad funcionan como variables en el sentido usual. Cada variación de un parámetro significa un modelo diferente, el cual es posteriormente evaluado permitiéndonos conocer las piezas de la lista *EKBM2L*. Un algoritmo basado en los procedimientos de muestreo e inferencia se enfrenta, en teoría, a un Análisis de Sensibilidad sobre un gran número de parámetros. En la práctica, el estudio de las proyecciones de la lista *EKBM2L* ayuda a reducirla, por una parte, al esquema de parámetros $E^{\downarrow PAR}$ y, por otra, al Análisis de Sensibilidad del correspondiente submodelo $E^{\downarrow VAR}$.

   El Análisis de Sensibilidad basado en las listas *KBM2L* que hemos introducido en la tesis puede considerarse una técnica complementaria o auxiliar de los actuales métodos de Análisis de Sensibilidad. Suministra una clasificación de los parámetros con respecto a la relevancia en términos de sensibilidad, y estudia la estabilidad de las alternativas óptimas dentro de la variación de los parámetros. Un inconveniente es que no tiene en cuenta un valor de sensibilidad, es decir, se refiere al cambio en la alternativa preferida, pero no a la magnitud del valor óptimo que determina el cambio (la utilidad esperada asociada al cambio).

En definitiva, hemos conseguido mostrar qué sensibilidad presenta un modelo frente a la variación de un grupo de parámetros a pesar del esfuerzo computacional que implica la elevada dimensionalidad.

Nuestra aproximación puede verse como una teoría de representación del conocimiento tabulado. Ésta será análoga a la teoría de las aplicaciones lineales donde conceptos como base/cambio de base, diagonalización/equivalencia,..., juegan papeles similares y la diferencia de fondo está en la estructura algebraica de las tablas–matrices y las operaciones. Por ejemplo, la *Base* óptima da una especie de forma canónica de Jordan a la matriz multidimensional, el $XBase$ es una especie de transformación ortogonal por semejanza y existen varias matrices que representan la misma tabla (información).

7.2. Futuras Líneas de Investigación y Temas Abiertos

Algunas posibles generalizaciones o extensiones de lo expuesto en la tesis son:

1. Respecto a los fundamentos de las listas *KBM2L*:

   a) Las **reglas de gestión** de los ítems están siendo revisadas actualmente para mejorar su definición y eficiencia. Tiene interés práctico disponer de otras que unan listas cualesquiera, ítems, bloques de ítems,...

   b) Es básico investigar en la mejora de la búsqueda de una buena *Base* **anticipando el rechazo de** *Bases* **no óptimas**, y mejorar los **operadores de cruce** del algoritmo genético, utilizando algunos diseñados expresamente para permutaciones. También podemos estudiar **la codificación** alternativa de los individuos.

   También creemos que hay posibles **conexiones con el Análisis Multidimensional de Datos** (componentes principales, conglomerados,...) que pueden ser interesantes para la optimización de las listas y por otra parte, las listas *KBM2L* pueden aportar a su vez resultados al análisis estadístico.

   c) Podríamos considerar restricciones sobre contextos que permitan **manejar la asimetría** que por naturaleza hay en los problemas de decisión. Algo ya se ha hecho en esta dirección. La marca *coKB* expresa que el ítem y el contexto asociado están restringidos o prohibidos. Las razones de la restricción son: asimetrías en la respuesta, contradicciones en el índice, asignaciones de probabilidad imposibles o incorrectas, asignaciones de utilidad imposibles o incorrectas,...

d) La representación puede, en principio, ser una ayuda para realizar tareas muy complejas de implementar en **ingeniería inversa** y obtener un modelo (DI o RB) nuevo o mejorado a partir de la lista *KBM2L*.

Dado el problema para el cual hemos construido un modelo, la evaluación nos proporciona unos resultados. La organización de los resultados en la lista *KBM2L* facilita un análisis que conduce a la revisión de partes concretas del modelo.

Un paso más es considerar junto a las optimizaciones de la *Base* en un esquema dado, las optimizaciones del propio esquema. Pensamos que el esquema de un modelo puede mejorarse formalizando mediante listas *KBM2L* los resultados o algunos de sus componentes. Estamos hablando de modificar el marco conceptual para que se ajuste al problema (que puede cambiar o ser análogo a otro modelo) y a los requerimientos de los agentes interesados en la toma de decisiones (que pueden ser varios y heterogéneos, (French, 2003)).

e) Extender la representación a **dominios continuos** (intervalos). En parte esto se ha realizado en el capítulo 5, pues se ha considerado tanto una respuesta continua como una dimensión continua respecto de la probabilidad. Quedan, no obstante, muchos temas abiertos como la **discretización** de los atributos continuos.

f) Añadir **medidas de preferencia e incertidumbre a los ítems** en las listas obtenidas a partir de las tablas de decisiones óptimas, que permitan realizar Análisis de Sensibilidad, que es uno de los aspectos esenciales de un Sistema de Ayuda a la Decisión. Tratamos de ver a varios niveles la bondad del modelo que subyace al sistema. Nos preguntamos si es correcta la asignación de probabilidades, la construcción de la función de utilidad, la resolución de dominios, la oportunidad de la relaciones de influencia y las redundancias.

En el marco descrito en la tesis, analizamos las tablas de decisiones óptimas de modelos de problemas de decisión. De algún modo es el primer paso en el Análisis de Sensibilidad y está directamente relacionado con la explicación. Los pasos siguientes se pueden plantear como una generalización de lo expuesto mediante la ampliación del esquema o introduciendo la utilidad en la respuesta vectorial de la tabla de decisiones óptimas.

La primera técnica es lo que hacemos en el capítulo 6. La segunda técnica supone que la respuesta contiene una medida derivada de la evaluación que se usa como indicador de sensibilidad. Es menos general pero permite abordar más fácilmente el Análisis de Sensibilidad en decisión (promedio de utilidades y probabilidades en ítems

de las listas *KBM2L*). Si las entradas están valoradas, se almacena en la medida de la respuesta. El problema abierto que surge es la selección del valor de utilidad que representa en un ítem la utilidad (medida) de los casos. Hay diferentes opciones (ni excluyentes ni exhaustivas) para recoger esta información. Por ejemplo: almacenar en el ítem la valoración del último caso insertado, la media, el máximo o el mínimo. En otros tipos de tabla esto puede ser distinto según el conocimiento que se pretenda representar.

Es interesante combinar ambas porque se enriquecen los procedimientos de explicación y el Análisis de Sensibilidad.

g) Tal vez, la lista unidimensional de una matriz multidimensional sea demasiado restrictiva para la representación en ciertos problemas.

El **desplazamiento multidimensional** lo consideramos una línea de desarrollo futuro. Múltiples índices, ortogonales o disjuntos (descomposición del índice), y dimensiones mayores (descomposición del esquema). Esta consideración ya mencionada, por ejemplo, en la sección 2.2.2, nos conduciría a abandonar el marco de las listas *KBM2L* generalizando la representación de las tablas mediante tensores sobre una métrica de adyacencia entre puntos o celdas de memoria. Ahora el $XBase$ es mucho más complejo y consiste en un verdadero cambio de sistema de referencia (formal y conceptual) para la matriz multidimensional inicial. Ésta puede ser una línea esencial de continuación del trabajo.

Si conservamos el esquema (el marco conceptual de la matriz multidimensional), la generalización de la estructura de la lista *KBM2L* con desplazamiento y espectro multidimensional implica definir la aplicación $f : R^{n+1} \rightarrow R^k$ que usa varios índices y desplazamientos. Esta *KBM2L* multidimensional implica una definición de los ítems más compleja como generalización de (2.1), la definición de adyacencia de casos. Esta idea ha surgido con especial interés en la construcción y análisis de una *EKBM2L* para realizar análisis de sensibilidad de diagramas de influencia. Para una investigación futura, en lugar de la alineación unidimensional de la información de las tablas de decisiones óptimas respecto de parámetros y variables, pensamos definir una ordenación bidimensional respecto de dos índices o desplazamientos, asociados respectivamente a los parámetros y variables. Obtenemos, de este modo, un espectro conjunto de las listas proyección $E^{\downarrow PAR}$ y $E^{\downarrow VAR}$.

De este modo extendemos las representaciones gráficas basadas en la memoria física del ordenador (espectros) y proponemos generalizaciones de la percepción espacial del

sistema basado en el conocimiento: lineal y no lineal, en el espacio de representación del conocimiento.

h) Una investigación en curso considera *Bases* más generales, con **permutaciones de los órdenes de los dominios de los atributos discretos**, a costa de agrandar mucho el espacio de búsqueda. En el capítulo 5 hemos considerado esta posibilidad en tablas cuyo esquema tiene pocos atributos pero los dominios son extensos, obtenidos al discretizar el intervalo [0,1] de la respuesta/dimensión de probabilidad.

2. Respecto a las aplicaciones de las listas *KBM2L*:

a) El ámbito de nuestra metodología es bastante amplio en tanto que abarca el aprendizaje, explicación, muestreo del espacio de representación, inferencia a partir del conocimiento parcial obtenido por medio de instancias, construcción de BC, AS,... Concretamente alcanza áreas del aprendizaje automático como la minería de datos, y de la estadística como el analisis multidimensional de datos (reconocimiento de patrones) y la ingeniería inversa de Modelos Gráficos Probabilistas.

De este modo, **las aplicaciones no se restringen a las tablas de Modelos Gráficos Probabilistas**. El procedimiento presentado es también útil en situaciones donde los datos provienen de programas matemáticos, de bases de datos, de sistemas de adquisición de datos, de internet,... Del mismo modo que en las Tablas de Decisiones Óptimas, el conocimiento en cualquier dominio puede representarse mediante relaciones entre atributos relevantes y proposiciones de interés (alternativas de decisión, medidas de incertidumbre, estados de un sistema,...) (Mira et al., 2000).

b) En problemas de toma de decisiones, las preferencias y la estructura de dependencias e independencias presentada en un modelo produce patrones de regularidad en las propuestas del Sistema de Ayuda a la Decisión. **La construcción de funciones de utilidad complejas**, de manera similar a las Tablas de Probabilidad Condicionada, es un tema abierto que podría ser abordado mediante las listas *KBM2L* y para el cual hay algunas propuestas. (Bohanec y Zupan, 2004) consideran que se puede construir la jerarquía de objetivos y la función de utilidad a partir de un conjunto de datos. Una función de n atributos que evalúa la utilidad es expresada de manera jerárquica descomponiendo los n atributos en dos grupos de forma que por ejemplo es $F(A,B,C,D) = F(A,G(B,C,D))$.

c) A pesar de la potencia de nuestro esquema iterativo de educción progresiva del conocimiento, las futuras investigaciones pueden centrarse en permitir **consultas con**

atributos restringidos en lugar de instanciados o completamente libres, modelizando las creencias iniciales sobre ellos. Un aspecto concreto a investigar es plantear cualquier clase de consulta, incluyendo atributos indefinidos (Fernández del Pozo y Bielza, 2002a). Aquí, los atributos no son libres, están sujetos a restricciones. Por ejemplo, sabemos que no toman algún valor.

También podría emplearse mayor esfuerzo en **la determinación de una buena** *Base* **operativa** si hay más de una. Los criterios podrían ser: mínimo esfuerzo computacional para obtener la nueva lista *KBM2L* y mínima fragmentación de los ítems.

Finalmente, aunque se permite al experto elegir los criterios de decisión en el algoritmo $\mathcal{A}2$ (véase la sección 4.3.2, página 166), podríamos alternativamente implementar una **búsqueda previa dentro del árbol de posibles sesiones**, es decir, posibles secuencias de respuestas $r_a - y^k_{a,i} - r_{a,i,b} - y^l_{a,i,b,j} - \ldots$ ($r_{\ldots}$ es el valor de la respuesta que ofrece la lista e $y^-_{\ldots}$ el valor del atributo instanciado). Esto podría filtrar posibilidades que no satisfacen las expectativas del experto, facilitando sus elecciones.

d) En el capítulo 5, los temas de la evaluación del DI y la propagación (inferencia) quedan como temas abiertos a futuros trabajos. Pensamos que las listas *KBM2L* son útiles para la educción y representación del modelo de probabilidad de un MGP y también para la inferencia y propagación de la probabilidad, pero no hemos estudiado este aspecto en profundidad. Esto conduciría al desarrollo de algoritmos para inferencia probabilista en RB y para la evaluación de DI que exploten el almacenamiento óptimo.

Otro aspecto es refinar la resolución de la probabilidad al ser discretizada, como un parámetro más de AS. Podemos introducir cualquier atributo en la *EKBM2L* que determine alguna característica de interés en el AS; el problema es la complejidad de la construcción y el análisis de la lista. Por ejemplo, podemos codificar la amplitud del dominio de las variables, la existencia de arcos de influencia entre variables o la propia existencia de variables en el modelo.

e) Mejorar las tareas de educción y validación de MGP. Actualmente trabajamos en una tercera representación de la lista *KBM2L*, con tablas cuyos índices se componen del parámetro de probabilidad y el contexto. Así, la respuesta es el atributo condicional. En modelos muy complejos tiene sentido realizar consultas a las TPC, pues una tabla de varios miles de entradas es incomprensible, especialmente para un usuario que no esté muy familiarizado con el modelo. La *KBM2L* puede ser una ayuda en esta tarea.

f) (Mannila, 2000) destaca la existencia de lagunas en la teoría de la Minería de Datos. Teniendo en cuenta las coincidencias entre la Minería de Datos y nuestro contexto,

es posible modelizar tareas típicas como agrupamiento en conglomerados, clasificación, reglas de descubrimiento automático, compactación de datos, interpretación y comprensión del conocimiento descubierto, y pensamos que nuestra aproximación es prometedora de cara a resolver estas lagunas.

Naturalmente el registro de datos de un conjunto de observaciones presenta en general repeticiones. Si se quieren considerar repeticiones (en una tabla no hay repeticiones) se almacenaría la frecuencia en la respuesta o se añadiría un atributo–dimensión que numere los casos, es decir, la dimensión de observación. Esto lo hemos realizado de manera sencilla en el ejemplo 3.12 (véase la sección 3.7.2, página 133).

g) Respecto del AS, quedan varios aspectos por resolver satisfactoriamente. En primer lugar seguirá siendo una tarea muy pesada computacionalmente. Creemos que 10 o 20 parámetros en un modelo real es sólo la *punta del iceberg* y deberíamos mejorar la selección de parámetros. Hasta ahora sólo hemos considerado que los parámetros de interés tienen una distribución uniforme en el intervalo de estudio. Podemos mejorar este planteamiento suponiendo una distibución cualquiera para el conjunto de parámetros. Una primera aproximación realizaría el muestreo de modelos alrededor de la esperanza de los parámetros (los valores de referencia) siguiendo las distribuciones y debería marcar con *coKB* los modelos cuyos parámetros fueran poco probables. Por otra parte, con una muestra del espacio de parámetros podemos considerar la clasificación de los modelos no evaluados mediante técnicas como KNN (Mitchell, 1997), RB (Pearl, 1988), regresión logística (Fahrmeir et al., 1994),... La clasificación puede ser parte del AS.

h) Pensamos que la teoría expuesta en la tesis puede ser muy útil para la implementación de SAD distribuidos que nos permitan deslocalizar las BC y desacoplar las consultas de los usuarios y el aprendizaje del sistema. La distribución es factible, pues computacionalmente las listas admiten un procesamiento vectorial y paralelo directo. Además, la BC es fácil de replicar y/o fragmentar y la evaluación de los DI se puede realizar parcialmente (instanciando) a demanda de los usuarios.

Bibliografía

[Agrawal et al.1993] R. Agrawal, T. Imielinski, y A. Swami. 1993. Mining association rules between sets of items in large databases. En ACM SIGMOD, editor, *Proc. Conference on Management of Data*, páginas 207–216. ACM.

[Agrawal et al.1996] R. Agrawal, M. Mehta, J.C. Shafer, R. Srikant, A. Arning, y T. Bollinger. 1996. The quest data mining system. En E. Simoudis, J. Han, y U.M. Fayyad, editores, *Proc. 2nd Int. Conf. Knowledge Discovery and Data Mining, KDD*, páginas 244–249. AAAI Press, 2–4 . URL: citeseer.nj.nec.com/agrawal96quest.html.

[Agresti1996] A. Agresti. 1996. *An Introduction to Categorical Data Analysis.* John Wiley & Sons, Inc., primera edición.

[Alander1992] J.T. Alander. 1992. On optimal population size of genetic algorithms. En *Computer Systems and Software Engineering, 6th Annual European Computer Conference*, páginas 65–70. Proceedings CompEuro.

[Apte2003] C. Apte. 2003. The big (data) dig, data mining analytics for business intelligence and decision support. *OR/MS TODAY*, Febrero:24–28.

[Arnold et al.1994] A. Arnold, J. Beauquier, B. Bérard, y B. Rozoy. 1994. *Programas Paralelos: Modelos y Validación.* Editorial Paraninfo, primera edición.

[Ball y Lillis2001] M.J. Ball y J. Lillis. 2001. E-health: transforming the physician/patient relationship. *International Journal of Medical Informatics*, 61:1–10.

[Banzhaf1990] B. Banzhaf. 1990. The "molecular"travelling salesman. *Biological Cybernetics*, 64:7–14.

[Bielza et al.2000] C. Bielza, M. Gómez, S. Ríos-Insua, y J.A. Fernández del Pozo. 2000. Structural, elicitation and computational issues faced when solving complex decision making problems with influence diagrams. *Computers & Operations Research*, 27(7-8):725–740.

[Bielza et al.2003] C. Bielza, J.A. Fernández del Pozo, y P. Lucas. 2003. Finding and explaining optimal treatments. En M. Dojat, E. Keravnou, y P. Barahona, editores, *Artificial Intelligence in Medicine, 9th Conference on Artificial Intelligence in Medicine in Europe: Qualitative and Model-Based Reasoning in Biomedicine*, Volumen 2780 de *Lecture Notes in Artificial Intelligence*, páginas 299–303. Springer. ISBN: 3–540–20129–7.

[Bielza et al.2006] C. Bielza, J.A. Fernández del Pozo, y P. Lucas. 2006. Explanation of clinical decisions through the extraction of regularity patterns. *Decision Support Systems*, Aceptado:–.

[Bielza y Shenoy1999] C. Bielza y P.P. Shenoy. 1999. A comparison of graphical techniques for asymmetric decision problems. *Management Science*, 45(11):1552–1569.

[Bohanec y Zupan2004] M. Bohanec y B. Zupan. 2004. A function–decomposition method for development of hierarchical multi–attribute decision models. *Decision Support Systems*, 36:215–233.

[Bohem1981] B.W. Bohem. 1981. *Software Engineering Economics.* Prentice-Hall.

[Bøttcher y Dethlefsen2003] S.G. Bøttcher y C. Dethlefsen. 2003. deal: A Package for Learning Bayesian Networks. *Journal of Statistical Software*, 8(20):1—40.

[Bouckaert1995] R.R. Bouckaert. 1995. *Bayesian Belief Networks: from Construction to Inference*. Tesis doctoral, University of Utrecht, Holanda.

[Boutilier et al.1996] C. Boutilier, N. Friedman, M. Goldszmidt, y D. Koller. 1996. Context-Specific Independence in Bayesian Networks. En *12th Annual Conference on Uncertainty in Artificial Intelligence*, páginas 115–123. Morgan Kaufmann.

[Breiman et al.1993] L. Breiman, J.H. Friedman, R.A. Olshen, y C.J. Stone. 1993. *Classification and Regression Trees*. Chapman & Hall, primera edición.

[Brunk et al.1997] C. Brunk, J. Kelly, y R. Kohavi. 1997. MineSet: An Integrated System for Data Mining. En D. Heckerman, H. Mannila, D. Pregibon, y R. Uthurusamy, editores, *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining*, páginas 135–138. AAAI Press, Menlo Park, California. URL: citeseer.ist.psu.edu/brunk97mineset.html.

[Buntine1996] W. Buntine. 1996. A guide to the literature on learning probabilistic networks from data. *IEEE Trans. on Knowledge and Data Engineering*, 8(2):195–210.

[Burges1998] C.J.C. Burges. 1998. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167. URL: citeseer.ist.psu.edu/burges98tutorial.html.

[Burstein et al.2001] F. Burstein, T. Bui, y D. Arnott. 2001. Decision support in the new millennium. *Decision Support Systems*, 31:163–164.

[Cano et al.2000] A. Cano, S. Moral, y A. Salmerón. 2000. Penniless propagation in join trees. *International Journal of Intelligent Systems*, 15(11):1027–1059.

[Castillo et al.1997] E. Castillo, J. Gutierrez, y A. Hadi. 1997. *Expert Systems and Probabilistic Network Models*. Springer-Verlag, primera edición.

[Chen1976] P. Chen. 1976. The entity relationship model, toward a unified view of data. *ACM Trans. Data Base Systems*, 1(1):6–36.

[Chenoweth et al.2004] T. Chenoweth, K.L. Dowling, y R.D. St. Louis. 2004. Convincing DSS users that complex models are worth the effort. *Decision Support Systems*, 37:71–82.

[Cios et al.1998] K. Cios, W. Pedrycz, y R. Swiniarski. 1998. *Data Mining: Methods for Knowledge Discovery*. Kluwer Academic Publisher, primera edición.

[Clark y Niblett1989] P. Clark y T. Niblett. 1989. The CN2 induction algorithm. *Machine Learning*, 3:261–283. URL: citeseer.nj.nec.com/clark89cn.html.

[Clemen1996] R.T. Clemen. 1996. *Making Hard Decisions: An Introduction to Decision Analysis*. Duxbury, segunda edición.

[Cooper y Herskovits1992] G.F. Cooper y E. Herskovits. 1992. A Bayesian Method for the Induction of Probabilistic Networks from Data. *Machine Learning*, 9:309–347.

[Cooper1988] G.F. Cooper. 1988. A method for using belief networks as influence diagrams. En R.D. Shachter, T.S. Levitt, L.N. Kanal, y J.F. Lemmer, editores, *Proceedings of the 4th Workshop on Uncertainty in Artificial Intelligence*, páginas 55–63. Morgan Kaufmann.

[Cooper1990] G.F. Cooper. 1990. The Computational Complexity of Probabilistic Inference Using Bayesian Belief Networks. *Artificial Intelligence*, 42:393–405.

[Coupé et al.2000] V. Coupé, L. van der Gaag, y J. Habbema. 2000. Sensitivity analysis: An aid for belief-network quantification. *Knowledge Engineering Review*, 15:1–18. URL: citeseer.ist.psu.edu/article/coupe00sensitivity.html.

[Cristianini et al.2001] N. Cristianini, J. Shawe-Taylor, y R.C. Williamson. 2001. Introduction to a special issue on kernel methods. *Journal of Machine Learning Research*, 2:95–96.

[Cuena1985] J. Cuena. 1985. *Lógica Informática*. Alianza Editorial.

[Date1986] C.J. Date. 1986. *An Introduction to Database Systems*. Addison–Wesley, cuarta edición.

[Davies2002] S. Davies. 2002. *Fast Factored Density Estimation and Compression with Bayesian Networks*. Tesis doctoral, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA.

[Delahaye1988] J.P. Delahaye. 1988. *Outils Logiques pour L' Intelligence Artificielle*. Eyrrolles, tercera edición.

[Díez y Mira1995] F.J. Díez y J. Mira. 1995. Distributed reasoning and learning in Bayesian Experts Systems. En C.A. Ntuen, editor, *Advances in Fault–Diagnosis Problem Solving*. CRC Press.

[Díez1993] F.J. Díez. 1993. Parameter adjustment in Bayes networks. The generalizad noisy OR–gate. En D. Heckerman y A. Mamdani, editores, *Proceedings of the Ninth Conference on Uncertainty in Artificial Intelligence*, páginas 99–105. Morgan Kaufmann.

[Díez1994] F.J. Díez. 1994. *Sistema experto bayesiano para ecocardiografía*. Tesis doctoral, Facultad de Ciencias, Universidad Nacional de Educación a Distancia, Madrid.

[Domingos y Pazzani1996] P. Domingos y M. Pazzani. 1996. Beyond independence: Conditions for the optimality of the simple Bayesian classifier. En *Proceedings of the 13th International Conference on Machine Learning*, páginas 105–112. Morgan Kaufmann.

[Dorigo et al.1999] M. Dorigo, G. Di Caro, y L.M. Gambardella. 1999. Ant algorithms for discrete optimization. *Artificial Life*, 5(2):137–172.

[Druzdzel y Flyn2000] M. Druzdzel y R.R. Flyn. 2000. Decision support systems. En A. Kent, editor, *Encyclopedia of Library of Information Science*. Marcel Deker, Inc. URL: citeseer.nj.nec.com/.

[Druzdzel1996] M. Druzdzel. 1996. Qualitative verbal explanations in Bayesian belief networks. *Artificial Intelligence and Simulation of Behaviour Quarterly*, 94:43–54. URL: citeseer.nj.nec.com/druzdzel96qualitative.html.

[Duda et al.2001] R.O. Duda, P.E. Hart, y D.G. Stork. 2001. *Pattern Classification*. John Wiley & Sons, Inc.

[Eidt et al.1994] S. Eidt, M. Stolte, y R. Fischer. 1994. Helicobacter pylori gastritis and primary gastric non-hodgkin's lymphomas. *J. Clin. Pathol.*, 47:436–439.

[Elvira2002] Consortium Elvira. 2002. First european workshop on probabilistic graphical models, 6-8 november - 2002 - cuenca (spain), electronic proceedings. En J.A. Gámez y A. Salmerón, editores, *Probabilistic Graphical Models*, páginas 105–112.

[Everitt y Dunn1992] B.S. Everitt y G. Dunn. 1992. *Applied Multivariate Data Analysis*. Oxford University Press, primera edición.

[Ezawa1998] K. Ezawa. 1998. Evidence propagation and value of evidence on influence diagrams. *Operations Research*, 46(1):73–83.

[Fagiouli y Zaffalon1998] E. Fagiouli y M. Zaffalon. 1998. A note about redundancy in influence diagrams. *International Journal of Approximate Reasoning*, 19(34):231–246.

[Fahrmeir et al.1994] L. Fahrmeir, G. Tutz, y W. Hennevolg. 1994. *Multvariate Statistical Modelling Based on Generalizad Linear Models*. Springer, primera edición.

[Farquhar1984] P.H. Farquhar. 1984. Utility assessment methods. *Management Science*, 30(11):1283–1300.

[Fayyad et al.1996] U.M. Fayyad, G. Piatetsky-Shapiro, y P. Smyth. 1996. From data mining to knowledge discovery: An overview. En U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, y R. Uthurusami, editores, *Advances in Knowledge Discovery and Data Mining*, páginas 1–34. AAAI Press.

[Fayyad1997] U.M. Fayyad. 1997. Editorial. *Data Mining and Knowledge Discovery*, 1:5–10.

[Felli y Hazen1998] J.C. Felli y G.B. Hazen. 1998. Sensitivity analysis and the expected value of perfect information. *Medical Decision Making*, 18(1):95—109.

[Fernández del Pozo et al.2001] J.A. Fernández del Pozo, C. Bielza, y M. Gómez. 2001. Knowledge organisation in a neonatal jaundice decision support system. En J. Crespo, V. Maojo, y F. Martín, editores, *Medical Data Analysis*, Volumen 2199 de *Lecture Notes in Computer Science*, páginas 88–94. Springer.

[Fernández del Pozo et al.2005] J.A. Fernández del Pozo, C. Bielza, y M. Gómez. 2005. A list-based compact representation for large decision tables management. *European Journal of Operational Research*, 160, Special Issue on Decision Making and AI:638–662.

[Fernández del Pozo y Bielza2002a] J.A. Fernández del Pozo y C. Bielza. 2002a. An interactive framework for open queries in decision support systems. En F.J. Garijo, J.C. Riquelme, y M. Toro, editores, *Advances in Artificial Intelligence*, Volumen 2527 de *Lecture Notes in Artificial Intelligence*, páginas 254–264. Springer.

[Fernández del Pozo y Bielza2002b] J.A. Fernández del Pozo y C. Bielza. 2002b. New structures for conditional probablity tables. En J.A. Gámez y A. Salmerón, editores, *Proceedings of the 1st European Workshop on Probabilistic Graphical Models*, páginas 61–70. Universidad de Castilla–La Mancha.

[Fernández del Pozo y Bielza2004] J.A. Fernández del Pozo y C. Bielza. 2004. Sensitivity analysis and explanation of optimal decisions. En P. Lucas y L. van der Gaag, editores, *Proceedings of the 2nd European Workshop on Probabilistic Graphical Models*, páginas 81–88. Lorentz Center.

[Fernández1981] C. Fernández. 1981. *Caracterización Matemática de la Bases de Datos Relacionales a partir de los axiomas de Armstrong: Una aproximación desde el punto de vista analítico.* Tesis doctoral, Facultad de Informática. Universidad Politécnica de Madrid.

[Flach y Lachiche1999] P.A. Flach y N. Lachiche. 1999. Confirmation-guided discovery of first-order rules with tertius. *Machine Learning*, 42:61–95.

[Foley et al.1993] J. Foley, A. van Dam, S. Feiner, y J. Huges. 1993. *Computer Graphics.* Addison-Wesley Publising Company, Inc., segunda edición.

[Freitas2002] A.A. Freitas. 2002. *Data Mining and Knowledge Discovery with Evolutionary Algorithms.* Springer, primera edición.

[French1998] S. French. 1998. Design of a decision support system for use in the event of a radiation accident. En J.F. Girón y M.L. Martínez, editores, *Decision Analysis Applications*, páginas 3–18. Kluwer Academic Publishers.

[French2003] S. French. 2003. Modelling, making inferences and making decisions: The roles of sensitivity analysis. *Top*, 11(2):229–251.

[Frosh y Kaplan1999] D.L. Frosh y R.M. Kaplan. 1999. Shared decision making in clinical medicine: Past research and future directions. *American Journal of Preventive Medicine*, 17(4):285–294.

[Gallarie et al.1984] H. Gallarie, J. Minker, y J.M. Nicolas. 1984. Logic and databases: A deductive approach. *Computing Surveys*, 16:153–185.

[Glymour et al.1997] C. Glymour, D. Madigan, D. Pregibon, y P. Smyth. 1997. Statistical themes and lessons for data mining. *Data Mining and Knowledege Discovery*, 1:11–28.

[GeNIe2003] GeNIe. 2003. GeNIe: Graphical network interface, versión 2.0,. *Decision System Laboratory, U. Pittsburgh, School of Information Systems.* URL: http://www.sis.pitt.edu/~dsl.

[Gödel1976] K. Gödel. 1976. *Sobre proposiciones formalmente indecidibles de los Principia Mathematica y sistemas relacionados (Traducción castellana).* Cuadernos Teorema, primera edición.

[Goldberg1989] D.E. Goldberg. 1989. *Genetic algorithms in search, optimization and machine learning.* Addison-Wesley Publising Company, Inc., primera edición.

[Gómez et al.2000] M. Gómez, S. Ríos-Insua, C. Bielza, y J.A. Fernández del Pozo. 2000. Multiattribute utility analysis in the *IctNeo* system. En Y.Y. Haimes y R.E. Steuer, editores, *Research and Practice in Multiple Criteria Decision Making*, Volumen 487 de *Lecture Notes in Economics and Mathematical Systems*, páginas 81–92. Springer.

[Gómez et al.2006] M. Gómez, C. Bielza, J.A. Fernández del Pozo, y D. Ríos-Insua. 2006. A graphical decision-theoretic model for neonatal jaundice. *Medical Decision Making*, Aceptado:–.

[Gómez2002] M. Gómez. 2002. *IctNeo: Un sistema de ayuda a la decisión para el tratamiento de la ictericia en recién nacidos*. Tesis doctoral, Departamento de Inteligencia Artificial. Facultad de Informática. Universidad Politécnica de Madrid.

[Gómez2004] M. Gómez. 2004. Real–world applications of influence diagrams. En J.A. Gámez, Moral S., y A. Salmerón, editores, *Advances in Bayesian Networks*, Volumen 146 de *Studies in Fuzziness and Soft Computing*, páginas 161–173. Springer.

[Goodwin y Wright1998] P. Goodwin y G. Wright. 1998. *Decision Analysis for Management Judgement*. Jonh Wiley & Sons, Inc., segunda edición.

[Hamming1950] R.W. Hamming. 1950. Error detecting and error correcting codes. *Bell System Tech Journal*, 9(Abril):147–160.

[Hand et al.2001] D. Hand, M. Mannila, y P. Smyth. 2001. *Principles of Data Mining*. The MIT Press, primera edición.

[Hansen y Mladenović2001] P. Hansen y N. Mladenović. 2001. Variable neighbourhood search: Principles and applications. *European Journal of Operational Research*, 130:449–467.

[Hastie et al.2001] T. Hastie, R. Tibshirani, y J. Friedman. 2001. *The Elements of Statistical Learning*. Springer, primera edición.

[Haughton et al.2003] D. Haughton, J. Deichmann, A. Eshgi, S. Sayek, N. Teebagy, y H. Topi. 2003. A review of software packages for data mining. *The American Statistician*, 57(4):290–309.

[Heckerman1990] D. Heckerman. 1990. *Probabilistic Similarity Networks*. Tesis doctoral, Stanford University (CA).

[Henrion et al.1991] M. Henrion, J.S. Breese, y E.J. Horvitz. 1991. Decision analysis and expert systems. *Artificial Intelligence Magazine*, 12:64–91.

[Holland1975] J.H. Holland. 1975. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, primera edición.

[Holsapple2001] C.W. Holsapple. 2001. Knowledge management support of decision making. *Decision Support Systems*, 31:1–3.

[Holsheimer y Siebes1994] M. Holsheimer y A. Siebes. 1994. Data mining: The search for knowledge in databases. Informe Técnico CS-R9406, Centrum voor Wiskunde en Informatica (CWI), P.O. Box 94079, 1090 GB Amsterdam. URL: http://citeseer.ist.psu.edu/holsheimer91data.html.

[Horvitz et al.1988] E.J. Horvitz, J.S. Breese, y M. Henrion. 1988. Decision theory in expert systems and artificial intelligence. *International Journal of Approximate Reasoning*, 2:247–302. URL: citeseer.ist.psu.edu/horvitz88decision.html.

[Howard y Matheson2005] R.A. Howard y J.E. Matheson. 2005. Influence diagram. *Decision Analysis*, 2(3):127–143.

[Hu y Dill1993] A.J. Hu y D.L. Dill. 1993. Reducing BDD size by exploiting functional dependencies. En SIGDA EDAC, IEEE-CAS, editor, *Proceedings of the 30th Design Automation Conference*, páginas 266–271. ACM Press. URL: citeseer.ist.psu.edu/hu93reducing.html.

[Hugin2002] Hugin. 2002. URL: http://www.hugin.com. *Expert Software ©2001.*

[Hwang y Briggs1985] K. Hwang y F.A. Briggs. 1985. *Computer Architecture and Parallel Processing.* McGraw–Hill International Editions, primera edición.

[Imielinski y Mannila1996] T. Imielinski y H. Mannila. 1996. A database perspective on knowledge discovery. *Comm. of the ACM*, 39:58–64. URL: citeseer.nj.nec.com/imielinski96database.html.

[Ingargiola2003] G. Ingargiola. 2003. Building classification models: ID3 and C4.5. URL: http://www.cis.temple.edu/ ingargio/cis587/readings/id3-c45.html.

[Jaffar y Maher1994] J. Jaffar y M.J. Maher. 1994. Constraint logic programming: A survey. *Journal of Logic Programming*, 9/20:503–581. URL: citeseer.ist.psu.edu/article/jaffar94constraint.html.

[Jensen2001] F.V. Jensen. 2001. Graphical models as languages for computer assisted diagnosis and decision making. En S. Benferhat y P. Besnard, editores, *Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, Volumen 2143 de *Lecture Notes in Artificial Intelligence*, páginas 1–15. Springer.

[John et al.1994] G.H. John, R. Kohavi, y K. Pfleger. 1994. Irrelevant features and the subset selection problem. En H. Hirsh y W. Cohen, editores, *Machine Learning: Proceedings of the Eleventh International Conference*, páginas 121–129. Morgan Kaufmann.

[Jordan et al.1999] M.I. Jordan, Z. Ghahramani, T. Jaakkola, y L.K. Saul. 1999. An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233. URL: citeseer.nj.nec.com/article/jordan98introduction.html.

[Kadane y Wolfson1997] J.B. Kadane y L.J. Wolfson. 1997. Experiences in elicitation. *The Statistician*, 46(4):147–160.

[Karnaugh1953] M. Karnaugh. 1953. The map method for synthesis of combinational logic circuits, part i. *Trans. AIEE*, 72(9):593–599.

[Kautz y Selman1991] H. Kautz y B. Selman. 1991. A general framework for knowledge compilation. En H. Richter y M. Richter, editores, *Proceedings of the International Workshop on Processing Declarative Knowledge*, páginas 287–300. Springer. URL: citeseer.nj.nec.com/kautz91general.html.

[Keeney1982] R. Keeney. 1982. Decision analysis: An overview. *Operations Research*, 30:803–838.

[Kernighan y Ritchie1991] B.W. Kernighan y D.M. Ritchie. 1991. *El Lenguaje de programación C.* Prentice-Hall Hispanoamericana S.A., segunda edición.

[Kirkpatrick et al.1983] S. Kirkpatrick, C.D. Gelett, y M.P. Vecchi. 1983. Optimization by simulated annealing. *Science*, 220:621–630.

[Knuth1968] D.E. Knuth. 1968. *The Art of Computer Programming, Vol. 1: Fundamental Algorithms.* Addison-Wesley, Reading, primera edición.

[Kohavi y Sommerfield1998] R. Kohavi y D. Sommerfield. 1998. Targeting business users with decision table classifiers. En G. Piatetsky-Shapiro, R. Agrawal, y P. Stolorz, editores, *The Fourth International Conference on Knowledge Discovery and Data Mining*, páginas 249–253. AAAI Press, Menlo Park, California, EEUU. URL: citeseer.ist.psu.edu/kohavi98targeting.html.

[Kohavi1994] R. Kohavi. 1994. Bottom–up induction of oblivious, read–once decision graphs. En F. Bergadano y D. Raedts, editores, *European Conference on Machine Learning*, páginas 154–169.

[Kohavi1995a] R. Kohavi. 1995a. The power of decision tables. En N. Lavrac y S. Wroles, editores, *Proceedings of the European Conference on Machine Learning*, Volumen 914 de *Lecture Notes in Computer Science*, páginas 174–189. Springer.

[Kohavi1995b] R. Kohavi. 1995b. *Wrappers for Performance Enhancement and Oblivious Decision Graphs*. Tesis doctoral, Department of Computer Science, Stanford University (CA).

[Kozlov y Koller1997] A.V. Kozlov y D. Koller. 1997. Nonuniform dynamic discretization in hybrid networks. En D. Geiger, P. Shenoy, y P. Pundali, editores, *Thirteenth Conference on Uncertainty in Artificial Intelligence*, páginas 314–325. Morgan Kaufmann.

[Lacave2003] C. Lacave. 2003. *Explicación en redes Bayesianas causales. Aplicaciones Médicas*. Tesis doctoral, Departamento de Inteligencia Artificial. Facultad de Ciencias. Universidad Nacional de Educación a Distancia, Madrid.

[Larrañaga et al.1999] P. Larrañaga, C.M. Kuijpers, R.H. Murga, I. Inza, y S. Dizdarevic. 1999. Genetic algorithms for the travelling salesman problem: A review of representations and operators. *Artificial Intelligence Review*, 13:129–170.

[Lauritzen y Nilsson2001] S. Lauritzen y D. Nilsson. 2001. Representing and solving decision problems with limited information. *Management Science*, 47:1235–1251.

[Liebowitz2001] J. Liebowitz. 2001. Knowledge management and its link to artificial intelligence. *Experts Systems with Applications*, 20:1–6.

[Lohr2000] S.L. Lohr. 2000. *Sampling, Design and Analysis*. Duxbury Press, primera edición.

[Lozano et al.2006] J.A. Lozano, P. Larrañaga, I. Inza, y E. (editores) Bengoetxea. 2006. *Towards a New Evolutionary Computation. Advances on Estimation of Distribution Algorithms. Series: Studies in Fuzziness and Soft Computing, Vol. 192*. Springer, primera edición.

[Lucas et al.1998] P. Lucas, H. Boot, y B. Taal. 1998. Computer-based decision-support in the management of primary gastric non-hodgkin lymphoma. *Methods Inf. Med.*, 37:206–219.

[Lucas et al.2000] P. Lucas, N.C. Bruijn, K. Schurink, y A. Hoepelman. 2000. A probabilistic and decision–theoretic approach to the management of infectious disease at the ICU. *Artificial Intelligence in Medicine*, 19:251–279.

[Lucas1996] P. Lucas. 1996. Knowledge acquisition for decision–theoretic expert systems. *AISB Quarterly*, 94:23–33.

[Manly1994] F.J. Manly. 1994. *Multivariate Statistical Methods, A Primer*. Chapman & Hall, segunda edición.

[Mannila2000] H. Mannila. 2000. Theoretical frameworks for data mining. *SIGKDD Explorations*, 1(2):30–32.

[Martínez et al.2001] C. Martínez, A. Panholzer, y H. Prodinger. 2001. Partial match queries in relaxed multidimensional search trees. *Algorithmica*, 29:181–204.

[Mayfield2000] J. Mayfield. 2000. Evaluating plan recognition system: Three properties of a good explanation. *Artificial Intelligence Review*, 14:351–376.

[Michalewizc1992] Z. Michalewizc. 1992. *Genetic Algorithm + Data Structures = Evolution Programs*. Springer, primera edición.

[Michalski et al.1986] R.S. Michalski, I. Mozetic, J. Hong, y N. Lavrac. 1986. The multipurpose incremental learning system AQ15 and its testing application to three medical domains. En *Proceedings of the Fifth National Conference on Artificial Intelligence*, páginas 1041–1045. Morgan Kaufmann.

[Mira et al.2000] J. Mira, A.E. Delgado, G.J. Boticario, y F.J. Díez. 2000. *Aspectos Básicos de la Inteligencia Artificial*. Sanz y Torres, primera edición.

[Mitchell1997] T.M. Mitchell. 1997. *Machine Learning*. McGraw-Hill International Editions, primera edición.

[Mitchell1998] T.M. Mitchell. 1998. *An Introduction to Genetic Algorithms*. MIT Press, Cambridge, primera edición.

[Murthy1998] S.K. Murthy. 1998. Automatic construction of decision trees from data: A multidisciplinary survey. *Data Mining and Knowledge Discovery*, 2(4):345–389. URL: citeseer.nj.nec.com/article/murthy97automatic.html.

[Newman y Maisels1992] T.B. Newman y M.J. Maisels. 1992. Evaluation and treatment of jaundice in the term infant: A kinder, gentler approach. *Pediatrics*, 89(5):809–818.

[Nielsen y Jensen1999] T.D. Nielsen y F.V. Jensen. 1999. Well defined decision scenarios. En K.B. Laskey y H. Prade, editores, *15th Conference on Uncertainty in Artificial Intelligence*, páginas 502–511. Morgan Kaufmann.

[Nielsen y Jensen2003] T.D. Nielsen y F.V. Jensen. 2003. Sensitivity analysis in influence diagrams. *IEEE Transactions on Systems, Man, and Cybernetics–Part a: Sysyems and Humans*, 33(1):223–234.

[Niermann2005] S.Ñiermann. 2005. Optimizing the ordering of tables with evolutionary computation. *The American Statistician*, 59(1):41–46.

[O'Hagan1998] A. O'Hagan. 1998. Eliciting expert beliefs in substantial practical problems. *The Statistician*, 47:21–35.

[Olmsted1983] S.M. Olmsted. 1983. *On representing and solving decision problems*. Tesis doctoral, Ph.D dissertation, Dept. Eng. Economic Syst., Stanford Univ., Stanford, CA.

[Ooi et al.2000] B.C. Ooi, K.L. Tan, C. Yu, y S. Bressan. 2000. Indexing the Edges–A simple and yet efficient approach to high-dimensional indexing. En *Symposium on Principles of Database Systems*, páginas 166–174. URL: citeseer.nj.nec.com/chin00indexing.html.

[Pawlak1997] Z. Pawlak. 1997. Rough set approach to knowledge–based decision support. *European Journal of Operational Research*, 99:48–57.

[Pawlak2002] Z. Pawlak. 2002. Rough Set, decision algorithms and Bayes' theorem. *European Journal of Operational Research*, 136:181–189.

[Pearl1988] J. Pearl. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, primera edición.

[Piatetsky-Shapiro2000] G. Piatetsky-Shapiro. 2000. Knowledge discovery in databases: 10 years after. *SIGKDD Explorations*, 1(2):59–61.

[Poole y Zhang2003] D. Poole y N.L. Zhang. 2003. Exploiting contextual independence in probabilisitic inference. *Journal of Artificial Intelligence Research*, 18:263–313.

[Preece y Shinghal1994] A.D. Preece y R. Shinghal. 1994. Foundation and application of knowledge base verification. *International Journal of Intelligent Systems*, 9(8):683–701. URL: citeseer.nj.nec.com/preece94foundation.html.

[Pressman1993] R.S. Pressman. 1993. *Software Engineering. A Practitioner's Approach*. McGraw–Hill, Inc., tercera edición.

[Quinlan1986] J.R. Quinlan. 1986. Induction of decision trees. *Machine Learning*, 1:81–106.

[Quinlan1987] J.R. Quinlan. 1987. Simplifying decision trees. *International Journal of Man-Machine Studies*, 27:221–234.

[Quinlan1993] J.R. Quinlan. 1993. *C4.5: Programs for Machine Learning*. Morgan Kauffman.

[R Development Core Team2005] R Development Core Team. 2005. *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, ISBN 3-900051-07-0 edición. URL: http://www.R-project.org.

[Raiffa y Schlaiffer1961] H. Raiffa y R. Schlaiffer. 1961. *Applied Statistical Decision Theory*. Harvard University Press.

[Raiffa1968] H. Raiffa. 1968. *Decision Analysis: Introductory Lectures on Choices Under Uncertainty*. Addison–Wesley, Reading.

[Ríos-García1995] S. Ríos-García. 1995. *Modelización*. Alianza Universidad.

[Ríos-Insua et al.2000] S. Ríos-Insua, M. Gómez, C. Bielza, y J.A. Fernández del Pozo. 2000. Implementation of *IctNeo*: a decision support system for jaundice management. En K. Inderfurth, G. Schwödiauer, W. Domschke, F. Juhnke, P. Kleinschmidt, y G. Wäscher, editores, Operations Research Proceedings 1999*, Gesellschaft für Operations Research e.V. (GOR)*, páginas 554–559. Springer.

[Ríos-Insua et al.2002] S. Ríos-Insua, C. Bielza, y A. Mateos. 2002. *Fundamentos de los Sistemas de Ayuda a la Decisión*. RA–MA, primera edición.

[Ríos-Insua et al.2004] S. Ríos-Insua, A. Mateos, C. Bielza, y A. Jiménez. 2004. *Investigación Operativa: Modelos Determinísticos y Estocásticos*. Centro de Estudios Ramón Areces, S.A., primera edición.

[Ríos-Insua y F.2000] D. Ríos-Insua y Ruggeri. F. 2000. *Robust Bayesian Analysis*. Spriger.

[Rivest1987] R.L. Rivest. 1987. Learning decision lists. *Machine Learning*, 2(3):229–246. URL: citeseer.nj.nec.com/rivest87learning.html.

[Robison1965] J.A. Robison. 1965. A machine oriented logic based on the resolution principles. *Journal of ACM*, 12:23–41.

[S-PLUS2003] S-PLUS. 2003. S-plus ©2002. version 6.0. *Insightful Corporation.* URL: http://www.insightful.com/default.asp.

[Sanders et al.1999] G.D. Sanders, C.G. Hagerty, F.A. Sonnenberg, M.A. Hlarty, y D.K. Owens. 1999. Distributed decision support using a web–based interface. *Medical Decision Making*, 19:157–166.

[Schewefel1981] H.P. Schewefel. 1981. *Numerical Optimization of Computers Models*. John Wiley & Sons, Inc.

[Shachter1986] R.D. Shachter. 1986. Evaluating influence diagrams. *Operations Research*, 34(6):871–882.

[Shaw et al.2001] M.J. Shaw, C. Subramaniam, G.W. Tan, y M.E. Welge. 2001. Knowledge management and data mining for marketing. *Decision Support Systems*, 31:127–137.

[Shenoy2000] P.P. Shenoy. 2000. Valuation network representation and solution of asymmetric decision problems. *European Journal of Operational Research*, 121(3):579–608.

[Shim et al.2002] J.P. Shim, M. Warkentin, J. Courtney, D.J. Power, R. Sharda, y C. Carlson. 2002. Past, present, and future of decision support technology. *Decision Support Systems*, 33:111–126.

[SAS1990] SAS Institute Inc. SAS. 1990. *SAS/GRAPH Software: Reference, Version 6*. SAS Institute Inc., volúmenes 1 y 2, primera edición.

[Smith et al.1993] J.E. Smith, S. Holtzman, y J.E. Matheson. 1993. Structuring conditional relationships in influence diagrams. *Operations Research*, 41(2):280–297.

[Starr et al.1998] B. Starr, A. Farquhar, V.K. Chaudhri, y R. Waldinger. 1998. Knowledge-intensive query processing. En A. Borgida, V. Chaudhri, y M. Staudt, editores, *5th KRDB International Workshop on Knowledge Representation Meets Databases: Innovative applications programming and query interfaces*, páginas 18.1–18.6. CEUR Workshop Proceedings, Technical University of Aachen.

[Sterling y Shapiro1994] L. Sterling y E. Shapiro. 1994. *The Art of Prolog*. The MIT Press, segunda edición.

[Sun Microsistems J.D.K.2006] Sun Microsistems J.D.K. 2006. API Specification: Platform Standard Edition 5.0. *Sun Microsistems JAVA^{TM} 2*.

[Swayne et al.1998] D.F. Swayne, D. Cook, y A. Buja. 1998. Xgobi: Interactive dynamic data visualization in the X Window System. *Journal of Computational and Graphical Statistics*, 7(1):113–130.

[Taha2002] H.A. Taha. 2002. *Operations Research, an Introduction*. Prentice–Hall, Inc., séptima edición.

[Tanembaum1990] A. Tanembaum. 1990. *Distributed Operative Systems*. Prentice–Hall, Inc., tercera edición.

[Vomlelová y Jensen2004] M. Vomlelová y F.V. Jensen. 2004. An extension of lazy evaluation for influence diagrams avoiding redundant variables in the potentials. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 12(1):1–18.

[Wang et al.2001] S. Wang, C. Yu, y B.C. Ooi. 2001. Compressing the Index– A simple and yet efficient approach to high–dimensional indexing. En X.S. Wang, G. Yu, y H. Lu, editores, *Advances in Web-Age Information Management: : Second International Conference, WAIM 2001, Xián, China, July 9-11, 2001, Proceedings*, Volumen 2118 de *Lecture Notes in Computer Science*, páginas 291–304. Springer.

[Weihs et al.2005] C. Weihs, U. Ligges, K. Luebke, y N. Raabe. 2005. klaR: Analyzing German Business Cycles, A Package for Classification and Visualization. En D. Baier, R. Decker, y L. Schmidt-Thieme, editores, *Data Analysis and Decision Support*, páginas 335–343, Berlin. Springer.

[Winston1992] P.H. Winston. 1992. *Artificial Intelligence*. Addison-Wesley, tercera edición.

[Witten y Frank2005] I.H. Witten y E. Frank. 2005. *Practical Machine Learning Tools and Techniques with Java Implementations. Chapter 8: WEKA, Machine Learning Software in Java*. Morgan Kaufman Publishers. ©1999-2005, versión 3.4.4, URL: http://www.cs.waikato.ac.nz/ml/weka/index.html.

[Wong et al.1986] S.K. Wong, W. Ziarko, y R. Li Ye. 1986. Comparison of rough set and statistical methods in inductive learning. *International Journal of Man-Machine Studies*, 24:53–72.

[Zantema y Bodlaender2000] H. Zantema y H.L. Bodlaender. 2000. Finding small equivalent decision trees is hard. *International Journal of Foundations of Computer Science*, 11(2):343–354. URL: citeseer.nj.nec.com/zantema99finding.html.

[Zantema y Bodlaender2002] H. Zantema y H.L. Bodlaender. 2002. Sizes of ordered decision trees. *International Journal on Foundations of Computer Science*, 13(2):445–458.

[Zantema1998] H. Zantema. 1998. Decision trees: Equivalence and propositional operations. En H.L. Poutré y J. Herik, editores, *Proceedings 10th Netherlands/Belgium Conference on Artificial Intelligence (NAIC'98)*, páginas 157–166. URL: citeseer.nj.nec.com/zantema98decision.html.

[Zantema1999] H. Zantema. 1999. Sizes of decision tables and decision trees. Informe Técnico UU-CS-1999-31, Utrecht University. URL: http://www.cs.uu.nl/-docs/research/publication/TechRep.html.

[Ziv y Lempel1977] J. Ziv y A. Lempel. 1977. A universal algorithm for sequential data compression. *IEEE Transactions on Information Theory*, 23:337–343.

[Zopounidis2002] C. Zopounidis. 2002. MCDA methodologies for classification and sorting. *European Journal of Operational Research*, 138:227–228.