UNIVERSIDAD DE GRANADA
E.T.S. INGENIERÍA INFORMÁTICA

Departamento de
Ciencias de la Computación
e Inteligencia Artificial

TESIS DOCTORAL

Imprecisión e Incertidumbre en el Modelo Multidimensional: Aplicación a la Minería de Datos

Carlos Molina Fernández

Granada, Julio de 2005

Editor: Editorial de la Universidad de Granada
Autor: Carlos Molina Fernández
D.L.: Gr.1323 - 2005
ISBN: 84-338-3546-7

Imprecisión e Incertidumbre en el Modelo Multidimensional: Aplicación a la Minería de Datos

memoria que presenta

Carlos Molina Fernández

para optar al grado de

Doctor en Informática

Julio de 2005

DIRECTORA

María Amparo Vila Miranda

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN
E INTELIGENCIA ARTIFICIAL

E.T.S. INGENIERÍA INFORMÁTICA UNIVERSIDAD DE GRANADA

La memoria titulada “Imprecisión e Incertidumbre en el Modelo Multidimensional: Aplicación a la Minería de Datos”, que presenta D. Carlos Molina Fernández para optar al grado de Doctor, ha sido realizada en el Departamento de Ciencias de la Computación e Inteligencia Artificial de la Universidad de Granada bajo la dirección de la Doctora María Amparo Vila Miranda.

Granada, Julio de 2005.

El Doctorando

La Directora

Fdo. Carlos Molina Fernández

Fdo. María Amparo Vila Miranda

Agradecimientos

Esta memoria refleja los resultados de un periodo en el que han intervenido muchas personas sin las cuales no podría estar escribiendo estas líneas.

La primera de ellas mi directora Amparo, por su dedicación y entrega, más allá de la obligación, para guiarme en este mundo de la investigación. En ella he descubierto una gran investigadora y, ante todo, una persona de una gran valía.

A los miembros del grupo de investigación IDBIS les tengo que agradecer su cálida acogida y el haber participado con sus comentarios y consejos a avanzar en el camino. Y como olvidar a los sufridores de mis abundantes visitas a sus despachos de Mecenas en busca de un rato de descanso. Entre ellos a Carmen, María, Antonio, Jesús, Nico y Fernando en los que he encontrado buenos amigos.

En general a todos los miembros del departamento de Ciencias de la Computación e Inteligencia Artificial por mostrarme que detrás de un frío artículo siempre hay personas de las que se puede aprender en muchos aspectos de la vida.

No puedo olvidar a mis nuevos compañeros del departamento de Informática de Jaén que me han acogido ayudándome en el siempre complicado cambio.

A Miguel Prados, Carmen Peña, Mª Elena Gómez, y José Mª Torre les tengo que agradecer el facilitarme los datos para poder probar las ideas propuestas.

Desde un punto de vista profesional este camino comenzó hace unos años. Sin embargo hubiera sido imposible llegar a empezar sin las personas que me han apoyado durante toda la vida y siguen haciéndolo: mi familia. Gracias a mis padres y abuela, por su entrega y cariño sin límites, por darme todo lo necesario para mi formación y, ante todo, por ser el mejor ejemplo para mi desarrollo como persona. A ellos les debo la persona en que me he convertido.

Gracias a mi hermano Luis, por estar siempre a mi lado, por escucharme y entenderme y por sus consejos desde el punto de vista del que lleva unos años de ventaja en este complicado mundo de la investigación. A Sonia, una reciente sufridora de la tesis, le tengo que agradecer sus palabras de ánimo y su ejemplo para completar este trabajo.

Y no puedo olvidar a Belén, por su paciencia y comprensión. A su dulce sonrisa le debo el poder continuar en muchos momentos en que la fría pantalla del ordenador parecía un abismo insalvable. Gracias por avanzar a mi lado en este camino y en el de la vida ayudándome a superar los obstáculos.

Dicen que somos la suma de lo que hacemos. No creo que pudiera haber realizado nada sin las personas que me rodean. Por eso, gracias.

Índice general

1. Introducción **1**

2. Conceptos previos **13**

2.1. Teoría de Conjuntos Difusos 13

2.1.1. Conjuntos y Bolsas Difusas 14

2.1.2. Operaciones sobre conjuntos difusos 17

2.1.3. Principio de extensión 21

2.1.4. Número difuso 21

2.1.5. Variables lingüísticas 24

2.1.6. Agregación de información difusa 26

2.1.6.1. Familia de Operadores OWA 27

2.1.6.2. OWA Difuso 29

2.1.6.3. Operador A_β^{OM} 30

2.1.6.4. Operadores de Rundensteiner y Bic adaptados . 32

2.1.6.5. Resumen lingüístico 35

2.2. Data Warehousing y OLAP 37

2.2.1. Data warehousing 37

2.2.1.1. Arquitectura de un Data Warehouse 38

2.2.2. OLAP y OLTP 42

2.2.3. El modelo multidimensional 46

2.2.3.1. Razones para su creación 46

2.2.3.2. Estructura 47
2.2.3.3. Operaciones 49
2.2.3.4. Modelos de implementación multidimensional . 53
2.2.4. Modelos multidimensionales clásicos 61
2.2.4.1. Modelo de Agrawal *et al.* 61
2.2.4.2. Modelo de Li y Wang 63
2.2.4.3. Modelo de R. Kimball 64
2.2.4.4. Modelo de Gray *et al.* 64
2.2.4.5. Modelo de Cabibbo y Torlone 65
2.2.4.6. Modelo de Datta y Thomas 66
2.2.5. Modelos Multidimensionales con tratamiento de imprecisión 67
2.2.5.1. Modelo de Dyreson 67
2.2.5.2. Modelo de Pedersen *et al.* 68
2.2.5.3. Modelo de Laurent 71
2.2.5.4. Modelo de Kaya y Alhajj 72
2.2.5.5. Modelo de English *et al.* 73

3. Modelo Multidimensional Difuso 75
3.1. Introducción 75
3.2. Modelo Multidimensional Preciso 76
3.2.1. Estructura Multidimensional Precisa 77
3.2.1.1. Dimensiones 77
3.2.1.2. Hechos Precisos 80
3.2.1.3. DataCubo Preciso 82
3.2.1.4. Ejemplo 83
3.2.2. Operaciones 85
3.2.2.1. Roll-up 85
3.2.2.2. Drill-down 87
3.2.2.3. Dice 88
3.2.2.4. Slice 89

3.2.2.5. Pivot 91
3.2.3. DataCubo Válido 92
3.3. Modelo Multidimensional Difuso 93
3.3.1. Estructura Multidimensional Difusa 93
3.3.2. Dimensión Difusa 93
3.3.2.1. Hechos Difusos 97
3.3.2.2. DataCubo Difuso 100
3.3.3. Operaciones 100
3.3.3.1. Roll-up 101
3.3.3.2. Drill-down 103
3.3.3.3. Dice 104
3.3.3.4. Slice 105
3.3.3.5. Pivot 106
3.4. Estructura Multidimensional con Jerarquías Lingüísticas 106
3.4.1. Dimensiones con jerarquías lingüísticas 109
3.4.2. Operaciones 110
3.5. Propiedades de las operaciones 113
3.5.1. Roll-up y drill-down 114
3.5.2. Pivot 115
3.5.3. Dice 115
3.6. Vista de Usuario 116
3.7. Ejemplo 122
3.7.1. Consulta 1 133
3.7.2. Consulta 2 137
3.8. Conclusiones 138

4. Extracción de reglas de asociación sobre DataCubos difusos 145
4.1. Introducción 145
4.1.1. Minería de datos: reglas de asociación 146
4.1.1.1. Conceptos generales 146
4.1.1.2. Extracción de reglas a múltiples niveles 148

4.1.2. OLAP Mining 151
4.2. COGARE: Extracción de reglas guiada por complejidad 155
4.2.1. Medidas de complejidad 156
4.2.1.1. Número de reglas 156
4.2.1.2. Abstracción 158
4.2.1.3. Medida de complejidad global 163
4.2.2. Medidas de calidad de reglas 164
4.2.2.1. Medidas de calidad clásicas 164
4.2.2.2. Calidad de un conjunto de reglas 170
4.2.3. Algoritmo 170
4.2.3.1. Generación de reglas 171
4.2.3.2. Generalización 175
4.2.3.3. Proceso completo 179
4.3. Experimentos 181
4.3.1. Parámetros de los procesos 183
4.3.2. Cálculo del soporte 184
4.3.3. Resultados 185
4.3.4. Análisis de tiempos 206
4.4. Conclusiones 209

5. Ejemplo de Esquemas Multidimensionales Difusos 211
5.1. DataCubo sobre datos médicos 212
5.1.1. Estructura del DataCubo 212
5.1.1.1. Dimensión *Duración* 214
5.1.1.2. Dimensión *Tiempo* 215
5.1.1.3. Dimensión *Paciente* 216
5.1.1.4. Dimensión *Material* 217
5.1.1.5. Dimensión *Lugar* 218
5.1.1.6. Dimensión *Causa* 221
5.1.1.7. Medidas 221
5.1.1.8. DataCubo 222

5.1.2. Consultas 222
5.2. DataCubo sobre datos contables 224
5.2.1. Estructura del DataCubo 225
5.2.1.1. Dimension *Tiempo* 226
5.2.1.2. Dimensión *Fallo* 226
5.2.1.3. Dimensión *Compañía* 227
5.2.1.4. Dimensión *Antigüedad* 227
5.2.1.5. Dimensión *Rentabilidad económica* 228
5.2.1.6. Dimensión *Fondo de maniobra* 230
5.2.1.7. Dimensión *Coste de endeudamiento* 230
5.2.1.8. Dimensión *Tamaño* 230
5.2.1.9. Dimensión *Auditoría* 231
5.2.1.10. Medidas 231
5.2.1.11. DataCubo 232
5.2.2. Consultas 232
5.3. DataCubo sobre datos del censo 234
5.3.1. Estructura del DataCubo 235
5.3.1.1. Dimensión *Persona* 235
5.3.1.2. Dimensión *Estudios* 238
5.3.1.3. Dimensión *Estado civil* 239
5.3.1.4. Dimensión *Tipo de trabajo* 239
5.3.1.5. Dimensión *País* 240
5.3.1.6. Dimensión *Relación* 240
5.3.1.7. Dimensión *Capital ganado* 242
5.3.1.8. Dimensión *Capital perdido* 242
5.3.1.9. Dimensión *Horas de trabajo* 243
5.3.1.10. Medidas 243
5.3.1.11. DataCubo 244
5.3.2. Consultas 244

6. Conclusiones y trabajos futuros 247

A. F-Cube Factory **255**
A.1. Introducción 255
A.2. Arquitectura 256
A.2.1. Servidor 257
A.2.1.1. DataCubos 257
A.2.1.2. Funciones de agregación 260
A.2.1.3. OLAM 261
A.3. Cliente 263
A.4. Manual de usuario 264
A.4.1. Creación y edición de DataCubos 264
A.4.1.1. Creación de un DataCubo 264
A.4.1.2. Creación de un DataCubo difuso 266
A.4.1.3. Edición de dimensiones y niveles 266
A.4.1.4. Edición de niveles 267
A.4.1.5. Definición de *Vistas de Usuario* 267
A.4.2. Consultas sobre DataCubos 268
A.4.3. Mostrar los hechos 270
A.4.4. Procesos de minería de datos 271
A.4.4.1. Crear un proceso 271
A.4.4.2. Consultar un proceso 272

Índice de figuras

1.1. Flujo de información y decisiones entre niveles 3
1.2. Arquitectura del sistema 4

2.1. Ejemplos de números difusos 23
2.2. Número difuso trapezoidal 23
2.3. Ejemplo de variable lingüística sobre la altura 25
2.4. Arquitectura de un Data Warehouse 40
2.5. Arquitectura detallada de un Data Warehouse 40
2.6. Pirámide de decisiones empresarial 46
2.7. Ejemplo de estructura multidimensional 48
2.8. Ejemplo de aplicación de las operaciones roll-up y drill-down sobre el datacubo de la figura 2.7 51
2.9. Ejemplo de aplicación de slice sobre el datacubo de la figura 2.7 . 52
2.10. Ejemplo de aplicación de pivot sobre el datacubo de la figura 2.7 . 52
2.11. Implementación del esquema de la figura 2.7 utilizando a) modelo en estrella b) modelo en copo de nieve 54
2.12. Arquitectura de un data warehouse para sistemas ROLAP 55
2.13. Arquitectura de un data warehouse para sistemas MOLAP 58

3.1. Ejemplo de jerarquía sobre las edades 78
3.2. Ejemplo de esquema multidimensional 84

3.3. Ejemplo de la aplicación de las operaciones roll-up y drill-down . 87
3.4. Ejemplo de la aplicación de la operación Slice 91
3.5. Ejemplo de la aplicación de la operación pivot 92
3.6. Definición de la relación de parentesco entre los niveles Grupo y Edad 95
3.7. Ejemplo de cálculo de la relación de parentesco extendido. a) camino Todas – Mayor Edad – Edad b) camino Todas – Grupo – Edad 97
3.8. Ejemplo de aplicación del principio de extensión al *mínimo* como *t-norma* 108
3.9. Resultado de aplicar resumen lingüístico 117
3.10. Estructura en dos capas 118
3.11. Representación gráfica de número difusos 119
3.12. Ejemplo de gráfica utilizando ambos enfoques de representación gráfica 120
3.13. Ejemplo de gráficas con ambos ejes difusos según diversas t-normas. A) producto. B) mínimo. C) Lukasiewicz 121
3.14. Diagrama de barras para el caso difuso 122
3.15. Ejemplo de esquema multidimensional 123
3.16. Etiquetas lingüísticas 127
3.17. Estructura del sistema para el ejemplo 129
3.18. Resultado de la consulta 1 en el caso preciso 134
3.19. Resultado de la consulta 1 en el caso difuso 135
3.20. Resultado de la consulta 1 en el caso difuso con jerarquías lingüísticas 135
3.21. Grupos de edades en el modelo Preciso Vs. Difuso 138
3.22. Resultado de la consulta 2 en el caso preciso 139
3.23. Resultado de la consulta dos en el caso difuso 141
3.24. Resultado de la consulta dos en el caso difuso marcando la tendencia 141

4.1. Evolución de la complejidad para diferentes tamaños de problemas 158

4.2. Comportamiento de la combinación de consistencia y cobertura propuesta por Michalski 166
4.3. Comportamiento de la combinación de consistencia y cobertura propuesta por Brazdil y Torgo 167
4.4. Generalización de reglas utilizando las dimensiones 171
4.5. Generalización de conjuntos no aceptados 173
4.6. Proceso de reducción de complejidad 176
4.7. Evolución del proceso sobre C_{Medico} difuso utilizando la medida de calidad Q_{Cohen}. El valor para el número de reglas representa el porcentaje sobre el número inicial 201
4.8. Mejor evolución del proceso sobre $C_{\text{Contables}}$ difuso utilizando la medida de calidad Q_{cons}. El valor para el número de reglas representa el porcentaje sobre el número inicial 203
4.9. Evolución del proceso sobre C_{Censo} difuso utilizando la medida de calidad Q_{cons}. El valor para el número de reglas representa el porcentaje sobre el número inicial 204

5.1. DataCubo sobre datos médicos 213
5.2. Distribución de las intervenciones según la duración 214
5.3. Valores y agrupación del nivel *Rango* en la dimensión *Duración* . 215
5.4. Clasificación de meses según su temperatura 216
5.5. Grupos de edades 217
5.6. Área metropolitana de Granada 219
5.7. Intervenciones según temperatura de los meses 223
5.8. Intervenciones según grupos de edades y duración 223
5.9. DataCubo sobre los datos contables 225
5.10. Valores y agrupación del nivel *Grupo* en la dimensión *Antigüedad* 228
5.11. Problema de bordes precisos con valores muy cercanos 229
5.12. Valores y agrupación de los niveles *Rango* 230
5.13. Número de empresas según fondo de maniobra y sector 233
5.14. Número de empresas fallidas según fondo de maniobra y sector . . 233

5.15. Número de empresas no fallidas según fondo de maniobra y sector 234
5.16. DataCubo sobre los datos del censo 236
5.17. Etiquetas sobre las edades 237
5.18. Etiquetas lingüísticas para el grado de la relación 241
5.19. Etiquetas y grados de pertenencia para el nivel *Rango* en las dimensiones *Capital ganado* y *Capital perdido* 242
5.20. Etiquetas y grados de pertenencia para el nivel *Rango* en la dimensión *Horas de trabajo* 243
5.21. Número de personas que hay en cada rango de horas de trabajo según si el trabajo es en el sector publico o no 245
5.22. Número de personas que hay en cada rango de capital ganado según se consideren minoría o no 246

A.1. Arquitectura del sistema 256
A.2. Arquitectura del servidor 257
A.3. Esquema del cálculo distribuido y paralelo 262
A.4. Pantallas del cliente WEB 263
A.5. Pantallas para la creación de un DataCubo 265
A.6. Pantallas de edición de una dimensión 266
A.7. Pantallas de edición de niveles 267
A.8. Pantalla de definición de *vistas de usuario* 268
A.9. Pantallas para realizar una consulta 269
A.10. Pantalla de selección de *vistas de usuario* 270
A.11. Pantallas de opciones de las gráficas 270
A.12. Pantallas para lanzar un proceso de minería de datos 271
A.13. Pantallas para consultar un proceso de minería de datos 273

Índice de cuadros

2.1. Ejemplo de t-normas 18
2.2. Ejemplo de t-conormas 19
2.3. Ejemplo de negaciones 21
2.4. Principales diferencias entre sistemas OLTP y OLAP 44

3.1. $\mu_{Gravedad,Causa}$ 127
3.2. $\mu_{Calidad,Proveedor}$ 127
3.3. $\mu_{Gravedad,Causa}$ 128
3.4. Hechos ejemplo sobre el esquema multidimensional internos a la compañía 130
3.5. Hechos ejemplo sobre el esquema multidimensional procedentes de la fuente con incertidumbre 131
3.6. Resultados de la consulta 1 en el caso preciso 133
3.7. Resultados de la consulta 1 en el caso difuso 134
3.8. Resultados de la consulta 1 con jerarquías lingüísticas 136
3.9. Resultados de la consulta 2 en el caso preciso 139
3.10. Resultados de la consulta 2 en el caso difuso 140

4.1. Tabla de contingencias con valores relativos 164
4.2. Algoritmo de cálculo de conjuntos frecuentes 174
4.3. Algoritmo de generalización sin pérdida de calidad 178

4.4. Algoritmo de generalización con pérdida de calidad 180
4.5. Algoritmo COGARE 182
4.6. Resultados tras la generación de reglas en $C_{\text{Médico}}$ preciso 186
4.7. Resultados tras la generalización sin pérdida en $C_{\text{Médico}}$ preciso . . 187
4.8. Resultados tras la generalización con pérdida en $C_{\text{Médico}}$ preciso . 187
4.9. Mejora tras aplicar la generalización sobre $C_{\text{Médico}}$ preciso 188
4.10. Resultados tras la generación de reglas en $C_{\text{Médico}}$ difuso 188
4.11. Resultados tras la generalización sin pérdida en $C_{\text{Médico}}$ difuso . . 189
4.12. Resultados tras la generalización con pérdida en $C_{\text{Médico}}$ difuso . . 189
4.13. Mejora tras aplicar la generalización sobre $C_{\text{Médico}}$ difuso 190
4.14. Resultados tras la generación de reglas en $C_{\text{Contables}}$ preciso 191
4.15. Resultados tras la generalización sin pérdida en $C_{\text{Contables}}$ preciso . 191
4.16. Resultados tras la generalización con pérdida en $C_{\text{Contables}}$ preciso 192
4.17. Mejora tras aplicar la generalización sobre $C_{\text{Contables}}$ preciso . . . 192
4.18. Resultados tras la generación de reglas en $C_{\text{Contables}}$ difuso 193
4.19. Resultados tras la generalización sin pérdida en $C_{\text{Contables}}$ difuso . 193
4.20. Resultados tras la generalización con pérdida en $C_{\text{Contables}}$ difuso . 194
4.21. Mejora tras aplicar la generalización sobre $C_{\text{Contables}}$ difuso 194
4.22. Resultados tras la generación de reglas en C_{Censo} preciso 195
4.23. Resultados tras la generalización sin pérdida en C_{Censo} preciso . . 195
4.24. Resultados tras la generalización con pérdida en C_{Censo} preciso . . 196
4.25. Mejora tras aplicar la generalización sobre C_{Censo} preciso 196
4.26. Resultados tras la generación de reglas en C_{Censo} difuso 197
4.27. Resultados tras la generalización sin pérdida en C_{Censo} difuso . . . 197
4.28. Resultados tras la generalización con pérdida en C_{Censo} difuso . . 198
4.29. Mejora tras aplicar la generalización sobre C_{Censo} difuso 198
4.30. Comparativa de resultados 200
4.31. Reducción media obtenida según la medida de calidad utilizada . . 206
4.32. Tiempos expresado en horas para la generación de reglas sobre los DataCubos $C_{\text{Contables}}$ preciso y difuso 207

4.33. Tiempos expresado en horas para la generalización sobre los DataCubos $C_{\text{Contables}}$ preciso y difuso 208
4.34. Tiempos expresados en horas para el proceso completo sobre los DataCubos $C_{\text{Contables}}$ preciso y difuso 209

5.1. Pertenencia de las localidades a el área metropolitana de Granada . 220
5.2. Código y grupos según al O.M.S. 221
5.3. Intervenciones según la duración y la procedencia de los pacientes 224
5.4. Sectores según códigos CNAE 227
5.5. Pertenencia de las razas a valor *Sí* en el nivel *Minoría* 238
5.6. Agrupación de los estudios según su nivel 238
5.7. Relación directa 241

A.1. Ejemplo de definición de una función de agregación para las relaciones de parentesco extendido en la configuración del servidor . . 259
A.2. Definición de métodos de minería de datos 261

Capítulo 1

Introducción

Sólo Dios puede hacer una elección al azar

LEY DE LEVY
Ley de Murphy

Presentación del problema y antecedentes: Tradicionalmente dentro de una organización se han delimitado diferentes niveles de decisión ([Ant65]):

- Planificación estratégica: ”la planificación estratégica es el proceso de decisión de objetivos de la organización, de los cambios de estos objetivos, de los recursos utilizados para alcanzar estos objetivos, y de las políticas para la adquisición, uso y disposición de estos recursos”[1].

[1] ”*Strategic planning is the process of deciding on objetives of the organization, on changes in these objetives, on the resources used to attain these objetives, and on the policies that are to govern the acquisition, use, and disposition of these resources”*

- Control de gestión: ”el proceso por el cual los directivos aseguran que los recursos son obtenidos y usados de forma efectiva y eficientemente para la realización de los objetivos de la organización”[2].

- Control operativo: ”el proceso de asegurar que unas tareas específicas son llevadas a cabo efectiva y eficientemente”[3]. La principal diferencia entre el control operativo y el control de dirección es que este último está más relacionado con las personas, mientras que el control operativo lo está con las tareas. En este nivel, se han de realizar menos decisiones dado que las tareas, objetivos y recursos se han delimitado mediante el control de dirección.

La frontera entre niveles no siempre es clara. Sin embargo, las necesidades de información para cada uno de ellos son bien distintas. En el control operativo la información que se requiere es interna a la organización, está bien definida y muy detallada, necesitándose que sea actual y precisa, dado que se va a utilizar para decisiones de inmediata aplicación.

En el caso de las decisiones estratégica, esta información es radicalmente distinta. Ahora necesitamos información en muchas ocasiones externa a la organización, donde el detalle no es importante (suele ser información agregada o resumida), que no se precisa que sea muy actual, dado que lo que se decidirán serán políticas a largo plazo y los objetivos de la organización. Mucha de esta información se obtendrá a través de la interacción humana.

Entre los niveles existe un flujo de información. Normalmente este va desde el nivel de control operativo hacia los niveles superiores (Figura 1.1). Sin embargo, el flujo de decisiones sigue la dirección contraria.

Las TIC intentan ayudar en todos los niveles del proceso de decisión. La gestión de la información en el nivel de control operativo se ha abordado de for-

[2] *”The process by which managers assure that resources are obtained and used effectively and efficiently in the accomplishment of the organization’s objetives”*

[3] *”the process of assuring that specific tasks are carried out effectively and efficiently”*

Figura 1.1: Flujo de información y decisiones entre niveles

ma satisfactoria mediante técnicas de almacenamientos de información y consulta (sistemas de gestión de bases de datos) que permiten un control eficiente de las tareas. Para los niveles altos, Data Warehousing y OLAP han sido las respuestas de la industria para ayudar a la toma de decisiones. Mediante la construcción de Data Warehouses dentro de las organizaciones se pretende mantener una visión coherente e integrada de la información interna a la empresa (*repositorios de conocimiento* dentro de la *infraestructura de servicios*, [Chu04]) que permite que se utilice para la elección y control de los objetivos de la organización (OLAP).

Conforme más alto es el nivel, la información que tendremos que utilizar será más desestructurada y obtenida de fuentes muy diversas. Ya en el comienzo de los 70 ([GSM71]) se vio la necesidad de modelos y lenguajes flexibles para manejar esta naturaleza mal definida en los sistemas de ayuda a la decisión (DSS).

Si se desea dar cobertura por parte de las nuevas tecnologías en estos niveles, se ha de permitir trabajar con esa heterogeneidad y poca estructuración. Si la información proviene de diferentes fuentes (Figura 1.2), muy probablemente presenten estructuras distintas y esquemas que no son compatibles. Existirá una gran labor para integrar los datos provenientes de estas fuentes, pero no en todos los casos se podrá llegar a una traducción completa a un esquema común. Si se

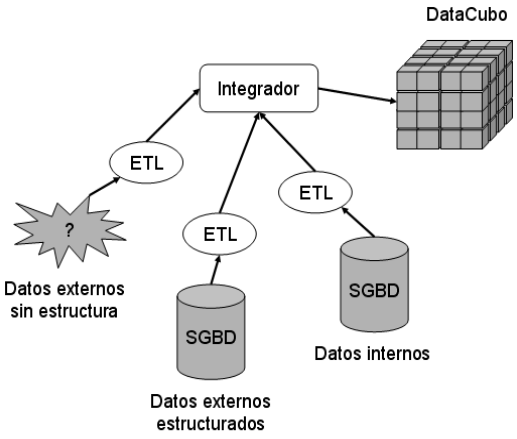


Figura 1.2: Arquitectura del sistema

integra dichos datos, el proceso tendrá que manejar esas incompatibilidades que se traducirán en una imprecisión y/o incertidumbre en los datos a considerar. Para tratar esta información, los sistemas deberán hacer frente a este hecho permitiendo modelarla y tratarla a los largo de los análisis que se requieran.

En el caso de la desestructuración de los datos, el problema que se plantea es el mismo. Los datos vendrán definidos en términos poco precisos. Si se quieren utilizar en el proceso de toma de decisiones, los sistemas que se desarrollen para ayudar a tal fin deberán permitir incluir estos datos y tratarlos de forma adecuada con el resto de los datos considerados.

Estos dos factores también se traducen en la contemplación de incertidumbre no sólo debida a los propios datos o su estructura, sino también asociada a las fuentes de procedencia. Estos datos serán aportados por fuentes externas a la empresa. Por esto, no se podrá controlar el proceso de adquisición de los mismos y, en la mayoría de los casos, ni siquiera se conocerá el proceso en sí. De esta forma, estos datos presentarán una calidad menor a los internos o, cuando

menos, desconocida como para considerarlos al mismo nivel. Los sistemas deberán hacer frente a este hecho permitiendo controlar estos casos e integrar los datos controlando la incertidumbre que tengan asociada.

En la actualidad, este tipo de sistemas se están aplicando a otros campos del conocimiento que presentan unos dominios más complejos. Uno de estos campos son los datos médicos. Este dominio no está tan fuertemente estructurado y su modelización dentro de sistemas de ayuda a la decisión es complicada, llegando a no poder representarse todos los detalles ([PJ98]). En estos casos de dominios más complejos se debe disponer de una gran flexibilidad a la hora de modelar las dimensiones de los datacubos. Las relaciones jerárquicas no son tan claras como en otros problemas y normalmente se definen de forma imprecisa.

Otra fuente importante de información para los procesos de toma de decisiones vendrá de expertos. Sería interesante poder modelar los datos aportados por estos al proceso de forma que se integre con el resto de los datos. Lo humanos tenemos una capacidad muy alta para trabajar con información imprecisa y, en la mayoría de los casos, la información que nos aporten los expertos vendrá sustentada en conceptos ambiguos que para nosotros es fácil de entender pero complicado de modelar utilizando modelos matemáticos clásicos. Por esto, su integración con el resto de las fuentes implicará la incorporación de conceptos vagamente definidos que de forma intuitiva se entienden pero cuya relación con el resto de los datos considerados será imprecisa.

En cualquier caso, el acceso a esta información ha de ser de la forma más flexible posible de cara al usuario (11ª regla de Codd para sistemas OLAP, [Cod93]). Este tipo de acceso normalmente no podrá ser previsto de tal forma que los sistemas deben de permitir una gran libertad para el usuario. Además, normalmente este no será una persona con grandes conocimientos informáticos o sobre los modelos matemáticos subyacentes. Esto hace que los sistemas deban dar un acceso intuitivo a la información y al moviendo por ella. En el caso de considerar la im-

precisión, los sistemas serán aún más complejos debido a los mecanismos para manejarla. Por esto mismo, en estos sistemas será incluso más importante dar un acceso intuitivo que oculte la complejidad subyacente.

Las primeras propuestas de modelos de datos para manejar esta información de cara al usuario fueron extensiones del modelo relacional utilizado en las bases de datos ([GCB+97]). Pero se vio que este modelo no se adaptaba a las necesidades de los tipos de análisis agrupados bajo la metodología OLAP ([Tho97]). El modelo multidimensional surgió como solución, proponiéndose múltiples propuestas ([AGS95, Kim96, CT97, CT98, DT99]). Estos modelos daban un acceso más intuitivo y más flexible a los datos que almacenaban. Sin embargo, las primeras propuestas no contemplaban la posibilidad del manejo de información imprecisa o la incertidumbre. Seguían proponiendo estructuras muy rígidas para la representación de los dominios.

Algunos modelos en este sentido han sido propuestas en los últimos años ([Dyr96, PJD01, JKPT04, AK03, Lau02, EGY04]). Los modelos propuestos incorporan el tratamiento de la imprecisión y/o incertidumbre en diferentes aspectos. Sin embargo, ninguno de los modelos propuestos propone mecanismos para la incorporación de información aportada por expertos. Un problema más importante que presentan es que la complejidad del manejo de imprecisión no queda oculta de cara al usuario, lo que hace los modelos difíciles de entender. Esto se debe a que la mayoría de ellos están pensados para utilizarse como soporte de datos para otros procesos automáticos.

Las técnicas de *extracción de conocimiento* o *data mining* también pueden ser herramientas útiles. Se trata de de métodos basados en inteligencia artificial y/o estadística que buscan patrones entre grandes cantidades de datos. De esta forma, su objetivo es extraer información no trivial que se encuentra almacenada en una base de datos. Una de las maneras más habituales de mostrarla al usuario es mediante reglas de la forma *si se cumple X entonces se también se da Y*, conocidas

como reglas de asociación.

Estas técnicas necesitan trabajar sobre datos integrados, consistentes y sobre los que previamente se ha aplicado una limpieza ([FPSSU96]). Por esto, en los últimos años se han propuesto nuevos métodos que trabajan sobre datacubos ([HCC$^+$97, NH94, EKS97, HCC98, KHC97, Zhu98]), dando origen a lo que se ha denominado OLAP Mining u OLAM. Integrar datos provenientes de múltiples fuentes, como pueden ser expertos, ayudarían a enriquecer los resultados obtenidos.

Algunas de estas técnicas se han adaptado para trabajar sobre los modelos con tratamiento de la imprecisión antes comentados ([AK03, Lau03b]). Cuando esta imprecisión se modela mediante la lógica difusa, estas técnicas se conocen como OLAM difuso o Fuzzy OLAM ([LBMD01]).

Las técnicas de extracción de reglas de asociación suelen tener el problema de obtener un conjunto muy numeroso de reglas. Esto hace que sea complicado por parte del usuario el interpretarlas. Las técnicas de *inducción orientada por atributos* utilizan taxonomías para reducir el número de reglas obtenidas. En el caso de basarlas en DataCubos, las jerarquías definidas en las dimensiones podrían cumplir la misma función. Además, utilizar conceptos más cercanos al usuario en la reglas ayudaría a su interpretabilidad. Sin embargo, como hemos comentado, estos conceptos pueden estar definidos de forma imprecisa. Para poder utilizarlos en el proceso necesitaríamos un modelo multidimensional capaz de representarlos.

Objetivos: Con estos planteamientos, lo que nosotros proponemos en esta memoria es un modelo multidimensional que ayude a modelar esta información mal definida. Para ello utilizaremos la lógica difusa, que ha sido ampliamente utilizada para trabajar con datos que presentan imprecisión e incertidumbre. Mediante la lógica difusa, desarrollaremos un modelo que permita una representación flexible que se adapte a las necesidades del decisor, incorporando información de múlti-

ples fuentes.

El conocimiento de los expertos puede ser una fuente muy interesante a utilizar en este tipo de análisis. Por ello, facilitaremos la integración de esta información dando mecanismos para incorporar la definición de las estructuras de una manera cercana al experto. A este modelo también lo dotaremos de mecanismos de respuesta a consultas que presenten la información de forma intuitiva para el usuario, aislándolo de la complejidad de los mecanismos internos. Sobre esta estructura, propondremos un método de extracción de reglas de asociación capaz de tratar con esta estructura imprecisa y orientado a reducir la complejidad del resultado final.

Más concretamente, los puntos a tratar serán:

- Desarrollar un modelo multidimensional que maneje imprecisión en toda la estructura y funcionamiento:
  - En los hechos, de tal manera que pueda representar y trabajar con la imprecisión propia de los valores así como la proveniente de las fuentes. Con ello pretendemos facilitar la integración de hechos provenientes de múltiples fuentes.
  - En las dimensiones, permitiendo la definición de múltiples jerarquías con la posibilidad de modelar imprecisión entre los elementos que las definen. En algunos caso esta imprecisión vendrá por la incertidumbre en las relaciones que unan los elementos. En otros, se deberá al modelado de conceptos vagamente definidos que introducen esta imprecisión al relacionarlos con el resto de los datos. Como en el caso anterior, cuanto más flexibles sean las estructuras de las jerarquías más fácil será integrar datos provenientes de esquemas con incompatibilidades.
  - El modelo debe soportar la integración de información proporcionado

por usuario y/o expertos de la manera más directa posible. La información puede venir en forma de conceptos definidos de forma imprecisa o en las mismas relaciones entre elementos precisos o que presenten imprecisión.

- Dar un conjunto de operaciones suficientemente amplio como para dar soporte a los análisis más habituales de los modelos multidimensionales. Estas operaciones deben trabajar con la imprecisión del modelo, dando como resultado un datacubo (el conjunto de operaciones debe ser cerrado).
- Toda la complejidad en el modelo para el manejo de la imprecisión e incertidumbre debe ser lo más transparente posible de cara al usuario final. Por esto, deben de desarrollarse mecanismos de representación de la información imprecisa para representar los resultados de consultas sobre los datacubos imprecisos. En la mayor parte de los casos, una representación gráfica de un resultado será más intuitiva que una numérica, sobre todo a los hora de comparar resultados. Por esto, el modelo debe de disponer de algún mecanismo de representación gráfica de los resultados.

- Sobre esta estructura, proponer una técnica de extracción de conocimiento mediante reglas de asociación con las siguientes características:

  - Trabajo sobre la estructura multidimensional comentada antes, de tal manera que trabaje sobre la estructura de datacubo difuso.
  - El método debe considerar los múltiples niveles de las dimensiones, permitiendo mezclar en una misma regla elementos definidos a diferentes niveles conceptuales.
  - Como en el caso del modelo multidimensional, el proceso de extracción debe estar orientado a obtener resultados lo más intuitivos posi-

bles para el usuario que los va a interpretar. De esta forma, deberá controlar la complejidad del resultado, intentando reducirla si es demasiado elevada. Para ello, deberemos identificar los factores que influyan en la complejidad del resultado y proponer un mecanismo para abordarlos.

Estructura de la memoria: En el siguiente capítulo presentaremos los conceptos básicos sobre los que se fundamentan el trabajo de investigación realizado. Comenzaremos presentando la Teoría de Subconjuntos Difusos. Veremos los principales conceptos que posteriormente utilizaremos para la definición de un modelo multidimensional difuso. La parte final del capítulo está dedicada a el Data Warehousing y la tecnología OLAP. Veremos las características de este tipo de sistemas y la necesidad que tienen de utilizar otros modelos de manipulación de datos. El principal es la visión multidimensional de los datos. Tras presentarlo, introduciremos algunas de las formalizaciones que se han realizado. Concluiremos este apartado con otras alternativas dentro de los modelos multidimensionales que se han desarrollado para tratar la imprecisión en diferentes aspectos.

El capítulo 3 entraremos de lleno en el modelo que proponemos que auna la visión multidimensional de datos con el tratamiento de información imprecisa haciendo uso de la lógica difusa. Haremos una formalización de la estructura y las operaciones.

Posteriormente, una vez definida la estructura, presentaremos el proceso de extracción de reglas propuesto sobre el modelo multidimensional difuso. Comenzaremos presentando algunos conceptos básicos de minería de datos y reglas de asociación. Posteriormente formalizaremos el método y presentaremos un estudio sobre su funcionamiento aplicado a datos reales de diferentes dominios.

En el capítulo 5 recogemos varios ejemplos de datacubos difusos definidos utilizando datos reales. En el último capítulo se recogen las principales conclusiones

de la investigación realizada y las futuras actuaciones para continuarla.

El apéndice A está dedicado a presentar el sistema F-Cube Factory. Se trata de un servidor OLAP que implementa los resultados de la presente memoria. Haremos una introducción a su arquitectura y veremos brevemente sus principales funcionalidades.

Capítulo 2

Conceptos previos

No hay una manera correcta de equivocarse

LEY DE IRENE
Ley de Murphy

En este capítulo presentaremos los conceptos sobre los que se cimienta el trabajo de investigación realizado. Comenzaremos presentando la Teoría de Conjuntos Difusos, dado que será una de las herramientas que utilizaremos para el desarrollo del modelo. El siguiente apartado se dedica a los conceptos básicos de Data warehousing y OLAP donde se encuadra la investigación.

2.1. Teoría de Conjuntos Difusos

En la presente memoria abordaremos el problema de la representación de la imprecisión en un modelo multidimensional. En el pasado se han propuesto múltiples aproximaciones a este problema en otros ámbitos cercanos (p.e. modelos de

bases de datos relaciones) mediante el uso de la Teoría de Conjuntos Difusos ([Zad65]). Nosotros seguiremos un enfoque similar.

En este apartado realizaremos una pequeña introducción a la teoría de conjuntos difusos para aclarar conceptos que posteriormente utilizaremos y fijar notación y representaciones. Esta introducción no pretende ser exhaustiva sino una presentación de los principales conceptos. Para mayor detalle véase [KY95].

2.1.1. Conjuntos y Bolsas Difusas

La noción de pertenencia de un elemento a un conjunto está muy ligada al cumplimiento por dicho elemento de la propiedad que define al conjunto. De esta forma se puede ver una propiedad como una función que, a cada elemento del universo de discurso $\mathcal{U}$, le asigna un valor en el conjunto $\{0,1\}$. Si un elemento pertenece al conjunto, es decir, cumple la propiedad, el valor asignado será 1. En caso contrario, 0. Con esto, una propiedad P determina un conjunto S_P formado por los elementos:

$$S_P = \{u \in \mathcal{U} / P(u) = 1\} \tag{2.1}$$

La Teoría de Conjuntos Difusos extiende esta definición. Ahora se sigue considerando las propiedades como funciones sobre el universo $\mathcal{U}$, pero cuya imagen será el intervalo cerrado [0,1]. Una propiedad, $\widetilde{P}$, de estas características se denominará difusa y determinara un conjunto con la siguiente expresión:

$$S_{\widetilde{P}} = \{< u, i > / \widetilde{P}(u) = i \wedge u \in \mathcal{U} \wedge i \in [0,1]\} \tag{2.2}$$

Definición 2.1 *Conjunto de elementos cada uno de los cuales tiene asociado un nivel de pertenencia al mismo en el intervalo [0,1].*

Para un conjunto difuso F, este grado de pertenencia de cada elementos vendrá determinado por la *función de pertenencia* μ_F cuya expresión será:

$$\mu_F : \mathcal{U} \rightarrow [0,1] \tag{2.3}$$

que para cada $u \in \mathcal{U}$ nos dará el grado con que este elemento pertenece al conjunto.

Cuando $\mathcal{U}$ es finito, $\mathcal{U} = \{x_1, x_2, ..., x_n\}$, podemos establecer un conjunto difuso *F* mediante la notación siguiente:

$$F = \mu_F(x_1)/x_1 + \mu_F(x_2)/x_2 + ... + \mu_F(x_n)/x_n = \sum_i \mu_F(x_i)/x_i \qquad (2.4)$$

El conjunto de todos los posibles subconjuntos difusos que se puede definir sobre un universo $\mathcal{U}$, las partes difusas, se denomina $\widetilde{\mathcal{P}}(\mathcal{U})$.

Definición 2.2 *El soporte de un conjunto difuso F definido sobre $\mathcal{U}$ es el conjunto de elementos con pertenencia mayor que 0 ($Soporte(F) = \{u \in \mathcal{U}/\mu_F(u) > 0\}$), es decir, todos aquellos elementos que pertenecen en algún grado al conjunto.*

Si esta definición la restringimos a aquellos elementos con pertenencia igual a 1, tendríamos el núcleo del conjunto.

Definición 2.3 *El núcleo de un conjunto difuso F definido sobre $\mathcal{U}$ es el conjunto de todos los elementos con pertenencia igual a 1 ($Nucleo(F) = \{u \in \mathcal{U}/\mu_F(u) = 1\}$).*

Diremos que un conjunto difuso está normalizado si:

$$sup_x \mu_F(x) = 1 \qquad (2.5)$$

En algunas ocasiones puede ser interesante no sólo saber los elementos que pertenecen en algún grado, sino aquellos que lo hacen cuando menos en un valor umbral determinado α. Estos conjuntos se denominan α-cortes.

Definición 2.4 *Sea $\alpha \in [0, 1]$, el α-corte del conjunto F será el conjunto de todos los elementos tales que su pertenencia sea mayor o igual que α ($F_\alpha = \{u \in \mathcal{U}/\mu_f(u) \geq \alpha\}$).*

Otro tipo de agrupación de elementos es el multiset, también conocido como bolsa. En este caso se trata una agrupación de valores donde estos pueden aparecer repetidos, ed decir, existe más de una ocurrencia de un mismo elemento. Una bolsa viene caracterizada por la función $card$.

Definición 2.5 *Sea E una bolsa definida con el dominio D_x. La función $card$ se define como*

$$card_E : D_x \rightarrow \mathbb{N} \tag{2.6}$$

donde, $\forall x \in D_x$, $card_E(x)$ representa el número de repeticiones del elemento x en la bolsa E.

Este concepto de bolsa fue extendido por Yager ([Yag86]) al campo difuso. Siendo $I = [0, 1]$, una bolsa difusa sobre el dominio D_x se caracteriza por una bolsa sobre el dominio $D_x \times I$. De esta manera, la función $card$ recibiría como parámetros dos valores: un elemento, x, del dominio D_x y un valor entre 0 y 1, μ. El valor devuelto en este caso representaría el número de ocurrencias del elemento x con grado de pertenencia μ en la bolsa difusa.

Veamos un ejemplo: sea $D_x = \{a, b, c, d\}$ y la bolsa *E*=$\{0'2/a, 0'5/b, 0'1/b, 0'2/a, 0'3/a, 0'7/d\}$. Esta bolsa tiene tres copias el elemento a con grados 0'2, 0'2 y 0'3. De esta forma $card(a, 0'2) = 2$, indicando que el elemento a aparece dos veces con grado 0'2, y $card(a, 0'3) = 1$, indicando en este caso que el elemento sólo aparece una vez con grado 0'3.

Al conjunto de todas las posibles bolsas difusas que se pueden definir sobre el dominio D_x lo notaremos como $\widetilde{\mathcal{B}}(D_x)$. El concepto que hemos introducido de α-corte para conjunto difusos tiene la misma aplicación para las bolsas, salvo que en este caso un α-corte podrá ser una bolsa difusa, es decir, pueden aparecer elementos repetidos.

Hemos introducido una extensión de las bolsas al campo difuso. En la literatura se pueden encontrar múltiples extensiones dando lugar a las bolsa difusas o

multisets ([Bli89, Knu81, Lak76, KSK96, DMBSV03]), con sus correspondientes operaciones.

La razón de introducir las bolsas difusas se debe a que el resultado de una consulta sobre un atributo en el que se pueden repetir valores es una bolsa. En el caso de trabajar con valores difuso, este conjunto será una agrupación del tipo bolsa difusa que hemos presentado.

En la siguiente sección presentamos las operaciones habituales sobre los conjuntos difusos. En el caso de las bolsas difusas, las operaciones caen fuera del ámbito de esta memoria.

2.1.2. Operaciones sobre conjuntos difusos

Al igual que se ha extendido el concepto de conjunto, también debemos de extender las operaciones habituales sobre conjuntos para poder aplicarlas sobre conjuntos difusos.

Antes de poder introducir la intersección entre conjuntos difusos necesitamos presentar la definición de la t-normas.

Definición 2.6 *Se denomina **t-norma** a toda función* $t : [0,1] \times [0,1] \rightarrow [0,1]$ *que verifica las siguientes propiedades:*

1. $t(\alpha, 1) = \alpha, \forall \alpha \in [0,1]$ ***(Frontera).***
2. $\beta \leq \gamma \Rightarrow t(\alpha, \beta) \leq t(\beta, \gamma), \forall \alpha, \beta, \gamma \in [0,1]$ ***(Monotonía).***
3. $t(\alpha, \beta) = t(\beta, \alpha), \forall \alpha, \beta \in [0,1]$ ***(Conmutativa).***
4. $t(\alpha, t(\beta, \gamma)) = t(t(\alpha, \beta), \gamma), \forall \alpha, \beta, \gamma \in [0,1]$ ***(Asociativa).***

A una función de este tipo la notaremos como $\otimes$.

Las t-normas representan el operador *tipo y* para la lógica difusa. Yager ([Yag93a, Yag93b]) presenta dos familias de operadores que caracterizan este com-

Nombre	**Expresión**
Mínimo	$t(a,b) = min(a,b)$
Producto	$t(a,b) = a \cdot b$
Lukasiewicz	$t(a,b) = max(0, a+b-1)$
Intersección drástica	$t(a,b) = \begin{cases} a & \text{si } b = 1 \\ b & \text{si } a = 1 \\ 0 & \text{en otro caso} \end{cases}$

Cuadro 2.1: Ejemplo de t-normas

portamiento. Se trata de los operadores MOM y MAM. Estos últimos agrupan a las funciones que presentan el comportamiento de la agregación tipo *y-lógico*.

Definición 2.7 *Sea H una función definida como*

$$H : \widetilde{\mathcal{B}}(I) \rightarrow I \tag{2.7}$$

es un operador tipo MAM si cumple las siguientes propiedades:

1. *Si $A \geq B$ entonces $H(A) \geq H(B)$.*
2. *Si $D = A \bigcup B$ entonces $H(A) \geq H(D)$.*
3. *Si $D = A \bigcup 1$ entonces $H(A) = H(D)$.*

En [Yag93a] Yager muestra cómo estos operadores tipo MAM son una generalización de la agregación *tipo y*.

Utilizando una t-norma t, la intersección de conjuntos difusos tendría una función de pertenencia con la siguiente expresión:

$$\mu_{A \cap B}(u) = t(\mu_A(u), \mu_B(u)) = \mu_A(u) \otimes \mu_B(u), \forall u \in \mathcal{U} \tag{2.8}$$

De igual forma, para la unión de conjuntos difusos nos tenemos que basar en una función auxiliar. En este caso se trata de las t-conormas.

Nombre	**Expresión**
Máximo	$u(a,b) = max(a,b)$
Suma algebraica	$u(a,b) = a + b - ab$
Lukasiewicz	$u(a,b) = min(1, a+b)$
Unión drástica	$u(a,b) = \begin{cases} a & \text{si } b = 0 \\ b & \text{si } a = 0 \\ 1 & \text{en otro caso} \end{cases}$

Cuadro 2.2: Ejemplo de t-conormas

Definición 2.8 *Se denomina **t-conorma** a toda función* $u : [0,1] \times [0,1] \to [0,1]$ *que verifica las siguientes propiedades:*

1. $u(\alpha, 0) = \alpha, \forall \alpha \in [0,1]$ *(Frontera).*
2. $\beta \leq \gamma \Rightarrow u(\alpha,\beta) \leq u(\beta,\gamma), \forall \alpha,\beta,\gamma \in [0,1]$ *(Monotonía).*
3. $u(\alpha,\beta) = u(\beta,\alpha), \forall \alpha,\beta \in [0,1]$ *(Conmutativa).*
4. $u(\alpha, u(\beta,\gamma)) = u(u(\alpha,\beta),\gamma), \forall \alpha,\beta,\gamma \in [0,1]$ *(Asociativa).*

A una función de este tipo la notaremos como $\oplus$.

En este caso, las t-conormas representan el comportamiento de las agregaciones *tipo o*. Los operadores MOM antes comentados son una generalización de este tipo de operadores.

Definición 2.9 *Sea* G *una función definida como*

$$G : \widetilde{\mathcal{B}}(I) \to I \tag{2.9}$$

es un operador tipo MOM si cumple las siguientes propiedades:

1. *Si* $A \geq B$ *entonces* $G(A) \geq G(B)$.

2. *Si* $D = A \bigcup B$ *entonces* $G(D) \geq G(A)$.
3. *Si* $D = A \bigcup 0$ *entonces* $G(A) = G(D)$.

En [Yag93a] Yager muestra cómo estos operadores tipo MOM son una generalización de la agregación *tipo o*. Puede observarse que la diferencia entre los operadores MAM y MOM radica en cual es el elemento neutro en la agregación. En el primer caso este elemento es el valor 1. En el segundo se considera el 0.

Basándonos en las t-conormas, la unión de conjuntos difusos tendría la expresión

$$\mu_{A \cup B}(u) = t(\mu_A(u), \mu_B(u)) = \mu_A(u) \oplus \mu_B(u), \forall u \in \mathcal{U} \tag{2.10}$$

La última función que nos queda por presentar es el complementario de un conjunto. Para hacerlo necesitamos un operador de negación.

Definición 2.10 *Se denomina* ***negación*** *a toda función* $c : [0,1] \rightarrow [0,1]$ *que verifique las siguientes propiedades:*

1. $c(0) = 1$ *y* $c(1) = 0$ ***(Frontera).***
2. $\alpha \leq \beta \Rightarrow c(\alpha) \geq c(\beta), \forall \alpha, \beta \in [0,1]$ ***(Monotonía).***

En muchas situaciones es deseable exigir más propiedades a las negaciones:

1. c es *continua*.
2. $c(c(\alpha)) = \alpha, \forall \alpha \in [0,1]$ ***(Involutiva).***

Con esta función, el conjunto complementario de un conjunto difuso F sería:

$$\mu_{\neg F}(u) = c(\mu_F(u)), \forall u \in \mathcal{U} \tag{2.11}$$

Nombre	Expresión
Estándar	$c(a) = 1 - a$
Umbral	$c(a) = \begin{cases} 1 & \text{si } a < umbral \\ 0 & \text{si } a \geq umbral \end{cases}$

Cuadro 2.3: Ejemplo de negaciones

2.1.3. Principio de extensión

La Teoría de Conjuntos Difusos aporta un método para extender la operaciones sobre conjuntos normales al caso difuso. Es lo que se conoce como el *Principio de Extensión de Zadeh* ([Zad75a]).

Definición 2.11 *Sea $f : \mathcal{U}_1 \times \mathcal{U}_2 \times ... \times \mathcal{U}_n \rightarrow \mathcal{Y}$ una función donde $\mathcal{U}_i, i = 1..n$, e $\mathcal{Y}$ son universos de referencia. Sean también F_1, F_2,..., F_n conjuntos difusos tales que $F_i \subseteq \mathcal{U}_i$.*

La extensión de la función f se define como la función $f : \widetilde{\mathcal{P}}(\mathcal{U}_1) \times \widetilde{\mathcal{P}}(\mathcal{U}_2) \times ... \times \widetilde{\mathcal{P}}(\mathcal{U}_n) \rightarrow \widetilde{\mathcal{P}}(\mathcal{Y})$, definida como $y \in \mathcal{Y}$ de la siguiente manera:

$$(f(F_1, F_2, .., F_n))(y) = sup_{(x_1,...,x_n) \in f^{-1}(y)}\{inf\{\mu_{F_1}(x_1), ..., \mu_{F_n}(x_n)\}\} \tag{2.12}$$

2.1.4. Número difuso

Uno de los tipos de conjuntos difusos de mayor relevancia son los definidos sobre el conjunto $\mathbb{R}$ de los números reales, es decir, con función de pertenencia $\mu_F : \mathbb{R} \rightarrow [0, 1]$. Estos conjuntos difusos pueden capturar los conceptos de números e intervalos aproximados, útiles en muchas aplicaciones. Por esto, introducimos el concepto de *número difuso* ([Zad75a, Zad75b]).

Definición 2.12 *Un **número difuso** F es un subconjunto difuso de $\mathbb{R}$ que verifica las siguientes propiedades:*

1. *La función de pertenencia es convexa, es decir*

$$\forall x, y \in \mathbb{R}, \forall z \in [0,1], \mu_F(z) \geq min\{\mu_F(x), \mu_F(y)\} \tag{2.13}$$

2. *F_α es un intervalo cerrado de $\mathbb{R}$ para todo $\alpha \in (0,1]$.*
3. *El soporte de F está acotado.*
4. *F está normalizado.*

Casos particulares de números difusos son:

- Los números reales (figura 2.1.a).
- Intervalos de números reales (figura 2.1.b).
- Valores aproximados (figura 2.1.c).
- Intervalos aproximados o difusos (figura 2.1.d).

En la figura 2.2 presentamos una de las formas más habituales de representar los números difusos. Lo que se utiliza es una función trapezoidal que se puede caracterizar por 4 parámetros. De forma genérica, un número difuso se puede definir como una función por partes.

Definición 2.13 *Un conjunto difuso F será un **número difuso** si y sólo si existe un intervalo cerrado $[a,b] \neq \emptyset$ tal que*

$$\mu_F(x) = \begin{cases} 1 & \text{si } x \in [a,b] \\ l(x) & \text{si } x \in (-\infty, a) \\ r(x) & \text{si } x \in (b, \infty) \end{cases} \tag{2.14}$$

donde $l : (-\infty, a) \to [0,1]$ que es monótona creciente, continua por la derecha, y tal que $l(x) = 0$ para $x \in (-\infty, \omega_1)$; $r : (b, \infty) \to [0,1]$, monótona decreciente, continua por la izquierda y tal que $r(x) = 0$ para $x \in (\omega_2, \infty)$.

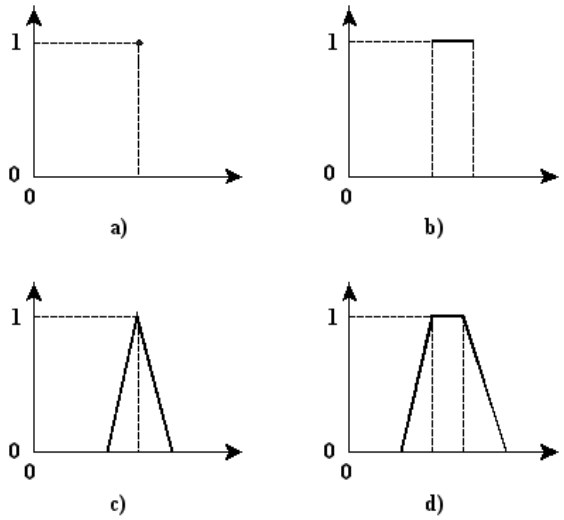


Figura 2.1: Ejemplos de números difusos

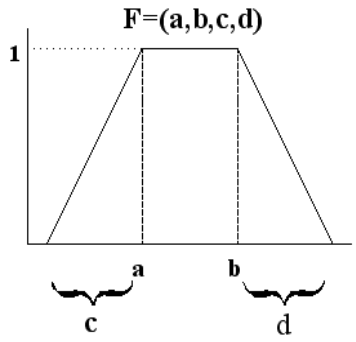


Figura 2.2: Número difuso trapezoidal

La demostración cae fuera del ámbito de esta introducción. Puede consultarse en [KY95]. Las operaciones aritméticas habituales sobre números reales se extienden a los números difusos mediante el *Principio de Extensión* antes presentado (sección 2.1.3).

2.1.5. Variables lingüísticas

El concepto de número difuso se puede utilizar para la definición de variables cuantitativas difusas, donde los posibles valores que puede tomar son números difusos. En el caso de que estos valores representen expresiones lingüísticas sobre el dominio (p.e. alto, bajo, muy bajo, etc.) estas construcciones son denominadas *variables lingüísticas* ([Zad75a, Zad75b]).

Una variable lingüística expresa mediante términos lingüísticos, interpretados como números difusos, una *variable base* (p.e. temperatura, presión, altura, etc.).

Definición 2.14 *Una **variable lingüística** está caracterizada por una quíntupla* $(X, T(X), \mathcal{U}, G, M)$*, en la que:*

- X *es el nombre de la variable.*
- $T(X)$ *es el conjunto de valores lingüísticos o etiquetas lingüísticas de* X*. Cuando los elementos de* $T(X)$ *tienen una sola palabra se denominan términos atómicos. En caso contrario se habla de términos compuestos.*
- $\mathcal{U}$ *es el universo de discurso de la variable.*
- G *es una regla sintáctica (normalmente en forma de gramática) que determina la forma de generar valores* $T(X)$*,*
- M *es una regla semántica que asocia a cada elemento de* $T(X)$ *su significado. Para cada valor* $L \in T(X)$*,* $M(L)$ *será un subconjunto difuso de* $\mathcal{U}$*.*

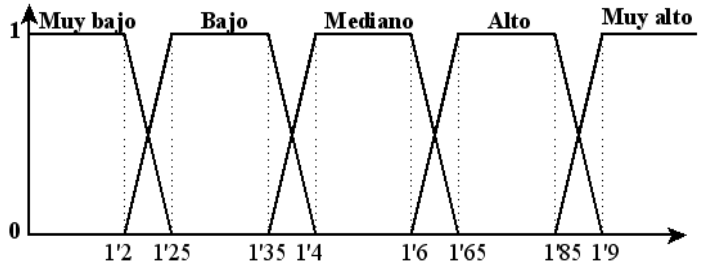


Figura 2.3: Ejemplo de variable lingüística sobre la altura

Un ejemplo de variable lingüística para abordar la altura se puede ver en la figura 2.3. En este ejemplo, el conjunto de etiquetas que consideramos sería $T(X) = \{Muy\ bajo, Bajo, Mediano, Alto, Muy\ alto\}$.

En una variable lingüística, el conjunto de etiquetas lingüísticas agrupan los valores del dominio subyacente, construyendo una jerarquía sobre este. Una propiedad importante asociada al número de etiquetas que utilizamos es la **granularidad**:

- Una granularidad alta (*coarse*) se corresponde con un número bajo de etiquetas. El dominio está poco particionado y tiene el inconveniente de que se pueda perder expresividad.

- Por el contrario, una granularidad fina (*fine*) se asocia con un número alto de etiquetas. Una granularidad demasiado baja puede provocar un aumento de la complejidad en la descripción del dominio.

El análisis de la granularidad en la representación de la información mediante etiquetas es un problema muy común en áreas de conocimiento tales como las bases de datos y los sistemas de ayuda a la decisión ([DVV93a]).

Al trabajar con etiquetas a veces no se realiza a nivel del conjunto de referencia, sino directamente sobre la etiquetas. En estos casos, se precisan operaciones que, recibiendo como argumento etiquetas de los conjuntos de referencia, el resultado sea a su vez una etiqueta (no forzosamente de alguno de los conjuntos de partida). A la hora de operar con etiquetas lingüísticas se puede presentar el problema de que tenemos que contemplar el uso de conjuntos de etiquetas con granularidades distintas. El problema se puede abordar desde dos puntos de vista:

- Trabajar al nivel de granularidad mayor (*coarsest*) de tal forma que las etiquetas de menor granularidad son traducidas a etiquetas de mayor grano.
- Trabajar al nivel más fino (*finest*). En este caso es justo al contrario, dado que convertimos las etiquetas de mayor grano a menor para operar.

Ambos enfoques tienen sus ventajas e inconvenientes y su elección dependerá de cada caso concreto. Para un estudio en más profundidad de las operaciones entre etiquetas véase [DVV93b]. Para abordar estos problemas se han propuesto modelos de representación de información lingüística más complejos ([ML99]).

2.1.6. Agregación de información difusa

A la hora de trabajar con datos, en muchas ocasiones es necesarios agregarlos para obtener información a menor nivel de detalle. En estos procesos se requieren funciones que realicen dicha agregación. En esta sección veremos algunas de ellas.

En los siguientes puntos comenzaremos presentando algunos mecanismos de agregación de información. Comenzaremos presentando la familia de operadores OWA y una extensión del mismo que hemos desarrollado. Posteriormente veremos una extensión de los operadores clásicos de agregación de bases de datos (p.e. suma, media, etc.) propuesta por Rundensteiner y Bic ([RB89]) y la adaptación de los mismos para utilizarlos en el modelo multidimensional difuso.

Los resultados de estas agregaciones suelen ser complicados de interpretar para un usuario dado que el resultado viene expresado mediante conjuntos difusos. Por ello, presentaremos una propuesta ([BSSV03]) para obtener un resumen en forma de expresión lingüística que sea más fácil de interpretar.

2.1.6.1. Familia de Operadores OWA

En esta sección presentaremos brevemente la familia de operadores OWA propuesta por Yager ([Yag88]). En el capítulo 3 utilizaremos una extensión de este operador para trabajar sobre las relaciones jerárquicas. A continuación lo presentamos junto a algunas de sus principales propiedades y en los siguientes puntos realizaremos una extensión del mismo para dar respuesta a nuestras necesidades particulares.

La familia OWA representa una clase de operadores de agregación parametrizables que incluyen el máximo, mínimo y la media. La capacidad que tienen para modelar diferentes agregaciones han hecho que se hayan utilizado en múltiples áreas como son la toma de decisiones, reconocimientos de patrones, recuperación de información, etc.

Lo primero que necesitamos para un operador OWA es saber qué se entiende por *vector de pesos*.

Definición 2.15 *Un vector* $v = [v_1, ..., v_n]$ *es un* vector de pesos *de dimensión n si y solo si* $v_i \in [0, 1]$ *y* $\sum_i v_i = 1$.

Con esta definición ya podemos presentar la familia propiamente dicha.

Definición 2.16 *Siendo* w *un* vector de pesos *de dimensión n, la función* $OWA_w : R^n \rightarrow R$ *es un* operador de media ordenada ponderada (OWA) *de dimensión n si*

$$OWA(a_1, ..., a_n) = \sum_i w_i a_{\sigma_i} \tag{2.15}$$

donde $\{\sigma_1, ..., \sigma_n\}$ *es una permutación tal que* $a_{\sigma_{i-1}} \geq a_{\sigma_i}$ *para todo* $i \in \{2, ..., n\}$.

Dependiendo del vector de pesos que utilicemos, el comportamiento del operador será distinto. Si $w_1 = 1$ y $w_i = 0$ para $i \in \{2, ..., n\}$, el resultado será el máximo de los valores contemplados. Desde un punto de vista lógico, la interpretación sería quedarse con el mayor grado de cumplimiento, es decir, buscar que al menos uno cumpla la propiedad correspondiente (O-lógico). Si por el contrario tenemos, $w_n = 1$ y $w_i = 0$ para $i \in \{1, ..., n-1\}$, el comportamiento sería quedarse con el menor grado con el que se cumple la propiedad (Y-lógico).

Definición 2.17 *Para un operador OWA con vector de pesos* w *definimos la función* $\alpha : [0,1]^n \to [0,1]$ *como*

$$\alpha(w) = \frac{1}{n-1} \sum_{i}^{n} (n-i) w_i \tag{2.16}$$

Esta función nos da una medida del carácter del operador: cuanto más cercano es este valor a 1, más se asemeja el operador al comportamiento del O-lógico, si por el contrario, este valor está cerca de 0, el comportamiento predominante será el Y-lógico.

Para medir la cantidad de información que utiliza un operador OWA se utiliza la medida de dispersión o entropía.

Definición 2.18 *Dado un vector de pesos w, la dispersión o entropía del operador OWA asociado es*

$$disp(w) = -\sum_{i} w_i ln w_i \tag{2.17}$$

Como ya hemos comentado, este operador ha sido ampliamente utilizado y se han realizado extensiones del mismo para representar la importancia o pesos en las fuentes de los valores ([Yag98, Tor97, SM99]), agregación de información

lingüística sin ([FH93, FH96]) y con pesos en las fuentes ([Tor97]). Ninguna de estas propuestas satisface nuestros requerimientos (Sección 3.4). En los siguientes puntos presentaremos una extensión de este operador.

2.1.6.2. OWA Difuso

En nuestro modelo lo que necesitaremos será agregar etiquetas lingüísticas y valores concretos de una manera eficiente (Sección 3.4). Por esto, necesitamos extender el comportamiento de la familia de operadores OWA para que pueda trabajar sobre valores difusos, y más concretamente, sobre números difusos subyacentes a las etiquetas.

Una de las características más importante de los operadores OWA es la ordenación que se debe realizar de los valores para realizar la media ponderada. En el caso difuso, tendremos también que realizar la ordenación de valores. Este problema ha sido ampliamente tratado en la literatura, existiendo múltiples propuestas ([DVV98, BD85]). Así pues, con cada método que utilicemos, obtendremos una familia de operadores OWA difusos distinta.

Definición 2.19 *Siendo w un* vector de pesos *de dimensión n, y OM un método de ordenación de números difusos, la función $\widetilde{OWA}_w^{OM} : \widetilde{R}^n \rightarrow \widetilde{R}$ es un* operador de media ponderada difuso (FOWA) *de dimensión n si*

$$\widetilde{OWA}^{OM}(\tilde{a}_1, ..., \tilde{a}_n) = \bigoplus_i w_i \widetilde{a}_{\sigma_i} \tag{2.18}$$

donde $\{\sigma_1, ..., \sigma_n\}$ es una permutación tal que $\tilde{a}_{\sigma_{i-1}} \leq_{OM} \tilde{a}_{\sigma_i}$ para todo $i \in \{2, ..., n\}$ y $\bigoplus$ es la suma extendida.

La caraterización de estos operadores (α) no varía en su definición, dado que ésta dependen del vector de pesos (w) que no se ha visto modificado al extenderlo a valores difusos.

2.1.6.3. Operador A_β^{OM}

Uno de los problemas que presenta los operadores OWA es la definición del vector de pesos a utilizar. Se han propuesto algunas técnicas para aprender los pesos basándose en ejemplos del comportamiento que se quiere reproducir ([Tor02]). Si lo que se quiere reproducir es el comportamiento lógico de la función (el valor de la función α) sería interesante poder obtener los pesos que se deberían asociar a la función. En nuestro caso, al tratar con las relaciones en la jerarquías, el número de elementos que agregar será diferente de un camino a otro en la jerarquía y lo que quedremos reproducir es el comportamiento de Y-lógico u O-lógico. En este punto presentamos un operador FOWA simple que hemos desarrollado en el que los pesos se definen en función del comportamiento que se desea obtener. Este operador lo llamaremos A_β^{OM}.

Definición 2.20 *Un operador de agregación A_β^{OM} será una función $A_\beta^{OM} : \widetilde{[0,1]}^n \rightarrow \widetilde{[0,1]}$ definida como*

$$A_\beta^{OM}(a_1, ..., a_n) = \widetilde{OWA}_w^{OM}(a_1, ..., a_n) \tag{2.19}$$

donde $w_1 = \beta$, $w_n = 1 - \beta$, y $w_i = 0$ para todo $i \in \{2, ..., n-1\}$.

Propiedad 2.1 *Un operador A_β^{OM} es un operador OWA con $\alpha(w) = \beta$.*

Demostración. El vector de pesos correspondiente al operador A_β^{OM} es $w = [w_1, ..., w_n]$ tal que $w_1 = \beta$ y $w_n = 1 - \beta$, teniendo el resto de los pesos a cero. Si calculamos el valor de $\alpha(w)$ obtenemos

$$\begin{gathered}\alpha(w) = \frac{1}{n-1}\sum_i^n (n-i)w_i = \\ = \frac{(n-1)w_1+(n-2)w_2+...+1w_{n-1}+0w_n}{n-1} = \\ = \frac{(n-1)\beta+(n-2)0+...+1\cdot 0+0(1-\beta)}{n-1} = \frac{(n-1)\beta}{n-1} = \beta\end{gathered}$$

Este operador presenta una propiedad que nos resulta muy útil

Propiedad 2.2 *Dados dos operadores de agregación, A_β^{OM} sobre n valores y $A_{\beta'}^{'OM}$ sobre m valores, entonces $\alpha(A_\beta^{OM}) = \alpha(A_\beta^{'OM}) = \beta$,*

es decir, fijando el valor de β se sabe cómo se comportará el operador independiente del número elementos para el que se defina.

Dependiendo del valor de β, el operador tendrá un comportamiento más cercano al O-lógico o el Y-lógico. Para notar que nos referimos a un operador de agregación con un comportamiento más cercano al O-lógico escribiremos A_β^{OM}. Si, por el contrario, queremos notar que el comportamiento es más cercano al Y-lógico, escribiremos $A_{1-\beta}^{OM}$.

Definición 2.21 *Dados dos operadores A_β^{OM} y $A_\beta^{\prime OM}$, diremos que son duales si $\beta' = 1 - \beta$.*

Dado que el operador de agregación está definido utilizando un operador OWA, podemos definir la dispersión asociada.

Definición 2.22 *Dado un operador de agregación A_β^{OM}, la dispersión $disp(A_\beta^{OM})$ tiene la expresión*

$$disp(A_\beta^{OM}) = -\beta ln\beta - (1-\beta)ln(1-\beta) \tag{2.20}$$

Esta expresión se obtiene directamente del cálculo de la dispersión del operador $\widetilde{OWA}$ que utilicemos. La dispersión de este será

$$disp(w) = -\sum_i w_i lnw_i = -w_1 lnw_1 - ... - w_n lnw_n =$$
$$-\beta ln\beta - 0ln0 - ... - (1-\beta)ln(1-\beta) = -\beta ln\beta - (1-\beta)ln(1-\beta).$$

Propiedad 2.3 *Dados dos operadores de agregación, A_β^{OM} sobre n valores y $A_{\beta'}^{'OM}$ sobre m valores, si $\beta = \beta'$ o $\beta = 1 - \beta'$ entonces $disp(A_\beta^{OM}) = disp(A_{\beta'}^{'OM})$.*

La demostración es directa de la definición 2.22.

2.1.6.4. Operadores de Rundensteiner y Bic adaptados

Rundensteiner y Bic ([RB89]) propusieron un modelo relacional posibilista al que dotaron de los operadores de agregación habituales (suma, máximo, mínimo, media, conteo). Aunque se hace referencia a un modelo posibilista se puede utilizar desde el punto de la lógica difusa debido a la equivalencia de ambas ([Zad88]).

Para nuestro modelo multidimensional difuso necesitaremos un conjunto de operaciones de agregación para realizar algunas de las operaciones (*roll-up* y *slice*). Lo que proponemos es utilizar una adaptación de los operadores propuestos por Rundensteiner y Bic.

Los autores definen varios tipos distintos de operadores de agregación dependiendo de la información con la que trabajan:

- Agregación de consultas difusas sobre valores precisos. Se trata de agregar valores cuyo dominio es preciso. La imprecisión viene dada por la consulta que selecciona los valores que agregar.

- Agregación de consultas precisas sobre valores difusos. Ahora se quiere agregar información de naturaleza imprecisa (se centra en conjuntos difusos) en donde la partición de los valores se hace mediante una propiedad precisa (no existe imprecisión asociada a la selección de los valores a agregar).

- Agregación de consultas difusas sobre valores difusos. En este caso el dominio subyacente a los valores a los que aplicar el operador es difuso (presenta imprecisión). La imprecisión puede también venir dada con la selección de los valores que agregar.

Los operadores se presentan basándose en un modelo de base de datos relacional posibilista. Nosotros realizaremos la definición basada en el concepto de

bolsa difusa que hemos presentado previamente.

En el caso del primer tipo de agregación (consulta difusa sobre valores precisos), se traduce en que tenemos un conjunto de elementos precisos que llevan asociado un valor difuso dado por la evaluación de la consulta. Es decir, se puede ver como una bolsa difusa definida sobre un dominio preciso. En este caso, la definición de los operadores de agregación sería la siguiente.

Definición 2.23 *Sea E una bolsa difusa defina sobre el dominio D_x. La aplicación del operador de agregación G sobre la bolsa E sería*

$$f_G(E) = \{\mu_{G(y)}/y | \mu_{G(y)} = sup_\alpha(G(E^\alpha) = y)\} \tag{2.21}$$

De la definición se puede ver que el mecanismo de agregación es aplicar la función que se desea aplicar (suma, media, máximo, mínimo, etc.) a cada alfa corte de la bolsa difusa. La principal propiedad de esta definición es que es *consistente* con los operadores normales en el caso preciso, es decir, si se aplica sobre datos precisos el resultado es el mismo que el aplicar las funciones habituales ([RB89]).

El segundo tipo de operador se define para agregar información en forma de conjuntos difusos. En este caso, tendríamos para agregar conjunto difusos de la forma $u_i = \{\mu_{i1}/u_{i1}, ..., \mu_{in}/u_{in}\}$. La definición general de los operadores sería la siguiente.

Definición 2.24 *Sea E un bolsa definida sobre el dominio $\widetilde{\mathcal{P}}(D_x)$ (el conjunto de todos los posibles conjuntos difusos definidos sobre D_x). La aplicación del operador de agregación f sobre la bolsa E sería*

$$f_G(E) = \{u/y | \exists u_{1i}...u_{mj}(G(\{u_{1i}, ..., u_{mj}\}) = y \wedge u = min(\mu_{1i}, ..., \mu_{mj}))\} \tag{2.22}$$

Como puede verse de la definición, lo que hace el operador es considerar todas las posibles combinaciones de entre un elemento de cada uno de los conjuntos difusos y aplicar el operador a cada uno de ellos, obteniendo el grado de pertenencia como el mínimo de los grados de pertenencia de los valores en cada caso.

Veamos un ejemplo para entenderlo mejor: sea *E*={ {*1'0/130*}, {*1'0/100, 0'9/110, 0'7/120*}} y el operador a utilizar la $suma$. La aplicación de la operación sería

$$f_{suma}(E) = \{1'0/130\} + \{1'0/100, 0'9/110, 0'7/120\} = \{1'0/(100 + 130), 0'9/(110 + 130), 0'7/(120 + 130)\} = \{1'0/230, 0'9/240, 0'7/250\}$$

Esta definición de operadores es *consistente* con la definición de las operaciones de agregación en el caso preciso, como en el caso anterior.

Para el caso de la media, los autores definen dos maneras de aplicarlo que no cumple con esta proposición general. En la primera considera realizar la aplicación de la $suma$ como lo acabamos de realizar, y dividir el resultado aplicando el $conteo$ según el esquema primero. El problema que vemos a esta solución es que el resultado puede no ser un conjunto difuso. La segunda realiza una media ponderada donde los pesos son el valor de pertenencia de cada valor. Este segundo enfoque es más simple y el principal inconveniente que le encontramos es que se obtiene un valor preciso partiendo de los conjuntos difusos. Esta simplificación eliminando imprecisión no nos parece la más adecuada. Por eso, nuestra elección ha sido seguir el esquema general de operadores que (1) el resultado es un conjunto difuso (por lo que las operaciones serían cerradas) y (2) realiza un manejo de la imprecisión, considerándola para futuras agregaciones.

En el último caso de agregaciones (consultas difusas sobre valores difusos) se refiere a la agregación como en el caso anterior pero tratándose de agregar una bolsa difusa en el que cada elemento es un conjunto difuso. Rundensteiner y Bic proponen establecer un umbral para agregar sólo aquellos conjuntos difusos con

grado de pertenencia superior al umbral prestablecido. Este enfoque realiza una simplificación eliminando imprecisión de los datos subyacentes.

Lo que tendremos para agregar será una bolsa difusa de la forma E=$\{\mu_1/u_1$, ..., $\mu_m/u_m\}$, en la que cada elemento será un conjunto difuso de la forma u_i=$\{\mu_{i1}/u_{i1}$, ..., $\mu_{in}/u_{in}\}$. De esta forma, cada elemento individual a agregar tendría la forma $\mu_i/\{\mu_{i1}/u_{i1}$, ..., $\mu_{in}/u_{in}\}$. Lo que nosotros proponemos es combinar el grado de certeza μ_i con el conjunto difuso para obtener otro que incluya ambas. En concreto, cada nuevo elemento sería u'_i=$\{min(\mu_{i1}, \mu_i)/u_{i1}$, ..., $min(\mu_{in}, \mu_i)/u_{in}\}$, de tal forma que realizaríamos la agregación sobre la bolsa E'=$\{u'_1$, ..., $u'_m\}$, que se reduce a aplicar la definición de operadores para el caso anterior.

De esta forma, tenemos un conjunto de operaciones de agregación cerradas (el resultado de una agregación se puede agregar utilizando alguno de los operadores definidos) y que son consistentes con las operaciones en el caso preciso.

2.1.6.5. Resumen lingüístico

Los operadores que acabamos de presentar trabajan sobre bolsas difusas, obteniendo como resultado un conjunto difuso. De cara a un usuario no experto, interpretar un esta salida es complicada. U´n resultado más fácil de entender de cara a un usuario es un número difuso y la interpretación lingüística que se puede obtener de este.

Blanco *et al.* ([BSSV03]) proponen utilizar esta idea para construir un resumen sobre bolsas difusas que haga el resultado más inteligible de cara a el usuario. La idea detrás de la propuesta podría ser resumida como, dada una bolsa difusa numérica $\widetilde{B}$ con soporte $[a, b] \in \mathbb{R}$, obtener el número difuso $\widetilde{F}$ definido sobre el mismo soporte tal que $\mathcal{M}(\widetilde{B}, \widetilde{F})$ es mínimo, donde $\mathcal{M}$ representa una medida de distancia, divergencia, etc. entre $\widetilde{B}$ y $\widetilde{F}$.

La formulación matemática del problema sería

$$\text{Minimizar } \mathcal{M}(\widetilde{B}, \widetilde{F}(m_1, m_2, a_1, a_2)) \tag{2.23}$$

sujeto a:

- $\widetilde{F}(m_1, m_2, a_1, a_2)$ sea un número difuso. Esto se traduce en las siguientes restricciones:
  - $-a_1 \leq 0$
  - $-a_2 \leq 0$
  - $m_1 - m_2 \leq 0$
- La imprecisión del número obtenido no supere la del conjunto de partida. Es decir $\mathcal{F}(\widetilde{F}) - \mathcal{F}(\widetilde{B}) \leq 0$. Para medirla los autores proponen utilizar la medida propuesta por Delgado *et al.* en [DVW98].

La medida $\mathcal{M}$ propuesta para realizar el ajuste tomar el opuesto a una medida de compatibilidad obtenida como resultado de evaluar la sentencia cuantificada

$$\text{"Q elementos de } \tilde{B} \text{ son } \tilde{F}\text{"},$$

utilizando $Q(x) = x$ como cuantificador relativo y el método GD ([DSV99]) para su evaluación. De esta manera, la medida utilizada sería

$$\mathcal{M}(\tilde{B}, \tilde{F}) = -GD_Q(\tilde{B}/\tilde{F}) \tag{2.24}$$

Para realizar la búsqueda del número difuso que mejor ajuste la bolsa difusa, a la hora de utilizarlos en nuestro modelo hemos utilizado un proceso de enfriamiento simulado. Hemos escogido este método debido a los buenos resultados que ha demostrado en múltiples áreas y a que se trata de un método bastante eficiente.

2.2. Data Warehousing y OLAP

Esta tesis doctoral se encuadra dentro de los conceptos de Data Warehousing y OLAP. En la presenta sección realizaremos una breve introducción a estos conceptos de forma que quede claro cuál es el contexto del trabajo.

2.2.1. Data warehousing

En [Inm96], Inmon identifica algunos de los principales problemas que surgieron cuando los usuarios comenzaron a realizar actividades orientadas al análisis a partir de datos desestructurados extraídos de bases de datos transaccionales. Encontrar los datos pertinentes para una estrategia de consulta era todo un problema. Y sobre todo porque se producían inconsistencias en los informes acerca del mismo fenómeno debido a la inconsistencia de datos que se presentaban en las tablas.

Inmon acuñó el término ”Data Warehouse” y sugirió que la finalidad de los Data Warehouses es hacer datos precisos, que sean consistentes en toda la empresa, accesible a todos los usuarios de una forma eficiente que no ocurre si los datos estuviesen residiendo en una base de datos operacional. La popularidad de los Data Warehouses creció rápidamente. Una posible definición de Data Warehouse sería la siguiente.

Definición 2.25 ***(Inmon y Hackathorn, [IH94])***. *Un* ***Data Warehouse*** *es una colección de datos orientados al tema, integrados, temporales, y no-volátiles para la toma de decisiones*[1].

Las compañías averiguaron al intentar implementar un Data Warehouse, que no se trata de una pieza de software que se pueda comprar, sino que más bien es un proceso de reingeniería de la información que se va formando con la organización.

[1] *”a Data Warehouse is a subject-oriented, integrated, timevariant, and non-volatile collection of data in support of management’s decisions”*

Por esta razón hay autores que prefieren usar el término como proceso: ”Data Warehousing” en lugar del término estático ”Data Warehouse” que carga con una semántica innecesaria de almacenamiento de datos. No se tienen los datos como se tienen en cualquier repositorio, sino que han de ser datos resumidos que ayuden más a los usuarios finales. Data warehousing implica en sí mismo la existencia de funciones que den soporte a la creación y mantenimiento de acceso a los usuarios finales para obtener datos completos y consistentes de la empresa. Aquí es donde surge el concepto de *OLAP* (*on-line analytical processing*).

Definición 2.26 ***(OLAP Council, [OLAa])*** *”OLAP es una categoría de tecnología software que permite a los analistas, directivos y ejecutivos acceder a los datos de forma rápida, consistente e interactiva a través de una amplia variedad de vistas de la información que han sido obtenidas de datos sin procesar para reflejar la dimensionalidad real de la empresa como la entiende el usuario”* [2].

Estas herramientas de análisis tienen unas características muy concretas que hacen que los sistemas normales de bases de datos no sean válidos. Por ello, se hacía necesario el desarrollo de nuevas herramientas que dieran respuestas a estas necesidades ([Cod93]).

2.2.1.1. Arquitectura de un Data Warehouse

Veamos a continuación un pequeño gráfico que nos ayudará a entender la arquitectura de un Data Warehouse (figura 2.4).

En esta arquitectura se tiene en cuenta que:

[2] *”On-Line Analytical Processing (OLAP) is a category of software technology that enables analysts, managers and executives to gain insight into data through fast, consistent, interactive access to a wide variety of possible views of information that has been transformed from raw data to reflect the real dimensionality of the enterprise as understood by the user”*

- Dentro de las fuentes de información pueden aparecer fuentes de datos no convencionales (p.e. bases de conocimiento, bases de datos documentales, HTML, XML, etc.)
- El sistema encapsulador incluye también un módulo que se denomina monitor, de manera que el encapsulador propiamente dicho se encarga de trasladar la información hacia el repositorio mientras que el monitor está en contacto directo con la fuente de conocimiento y avisa si se producen cambios.
- El integrador es el responsable de filtrar, resumir y unificar la información de forma que esté disponible en el Data Warehouse de manera más adecuada a los usuarios.

Si bien esta arquitectura sería la deseable hay que hacer notar que en los sistemas comerciales habituales no se contemplan muchas de las generalizaciones que supone, las principales restricciones de dichos sistemas son:

- Todas las fuentes y el Data Warehouse se construyen según un único modelo de datos, habitualmente el modelo relacional.
- La propagación de las fuentes de información al Data Warehouse normalmente se realiza off-line.
- Nunca se derivan consultas desde el integrador a las fuentes de información.

La arquitectura más detallada, propuesta por Chaudhuri y Dayal ([CD97]), viene recogida en la figura 2.5. Como puede verse incluye herramientas para extraer datos de diferentes bases de datos operacionales y fuentes externas, para limpiar, transformar e integrar estos datos, para cargar los datos en el Data Warehouse propiamente dicho y para actualizar periódicamente el Data Warehouse reflejando las actualizaciones en las fuentes, y eliminando los datos obsoletos, pasándolos quizás a un sistema de almacenamiento más lento.

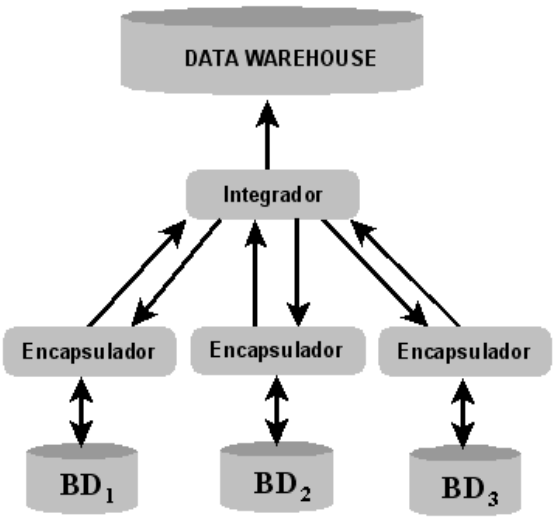


Figura 2.4: Arquitectura de un Data Warehouse

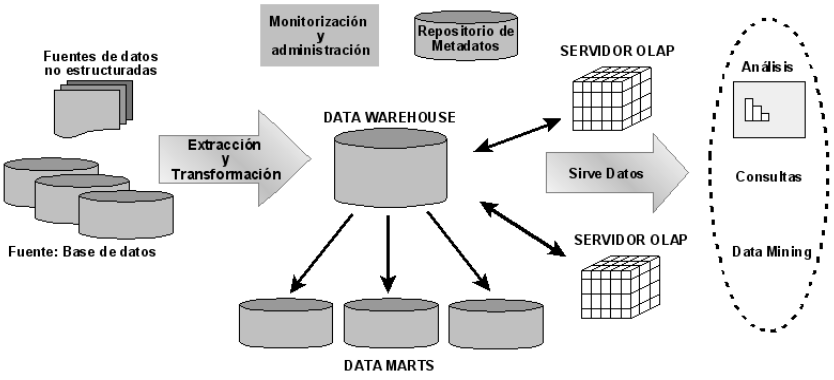


Figura 2.5: Arquitectura detallada de un Data Warehouse

También pueden verse los *Data Marts* departamentales, que son subconjuntos departamentales orientados a problemas más concretos (por ejemplo, el departamento de marketing puede incluir información acerca de clientes, productos y ventas), estos Data Marts pueden generarse y utilizarse de forma más flexible y rápida dado que no requieren un consenso de toda la empresa para generarse, pero pueden llevar a problemas de integración muy complejos si la empresa es realmente grande.

Los datos existentes en el Data Warehouse y en los Data Marts se manejan por medio de uno o varios servidores OLAP, y es aquí donde vemos la relación entre ambos conceptos. Estos servidores presentan vistas multidimensionales de los datos a una gran variedad de interfaces: interfaces de consulta directa, herramientas para generar informes, herramientas de análisis exploratorio (gráficos, estadística descriptiva, etc.) y herramientas de minería de datos propiamente dichas. Todo esto hace que con los servidores OLAP y sus robustas máquinas de cálculo, los datos históricos almacenados son mucho mejor utilizados. Los usuarios tienen una mayor producción de ideas gracias a la combinación de los accesos a las herramientas y una potente máquina analítica.

Finalmente hay un repositorio para almacenar metadatos y herramientas para monitorizar y almacenar todo el sistema.

El Data Warehouse puede distribuirse con el objetivo de equilibrar y mejorar la disponibilidad de los datos. En estas arquitecturas distribuidas el repositorio de metadatos se suele replicar para cada fragmento, la administración se lleva a cabo de forma centralizada. Una solución alternativa para la descentralización es el uso de Data Marts que pueden verse como mini-Data Warehouses asociados a cada departamento de la empresa donde cada uno de los cuales tiene sus repositorios y administración propias.

Diseñar y poner en marcha un Data Warehouse es un complejo proceso que supone el desarrollo de las siguientes actividades:

- Definir la arquitectura, planificar la capacidad, seleccionar los servidores de almacenamiento, los servidores OLAP y de bases de datos y las herramientas.
- Integrar todas las estructuras software seleccionadas.
- Diseñar el esquema del Data Warehouse así como las vistas asociadas.
- Definir la organización física del Data Warehouse, situación de los datos, distribución y métodos de acceso.
- Diseñar e implementar las herramientas de extracción de datos, limpieza, transformación, carga y actualización.
- Rellenar el repositorio con el esquema, definiciones de vistas y otros metadatos.
- Diseñar e implementar los interfaces de usuario.
- Poner en marcha el Data Warehouse y sus aplicaciones.

2.2.2. OLAP y OLTP

Como ya hemos visto, OLAP sirve para realizar un análisis exhaustivo de los datos que se han ido almacenando a lo largo de la historia de una empresa. Millones de registros que necesitan ser analizados por las personas más responsables de la empresa para lograr una mejor producción y reducción de costes.

Dado el objetivo que tienen estos sistemas, en algunas ocasiones se han denominado también ”Sistemas de Soporte a la Decisión” y en muchos sistemas comerciales OLAP y DSS se han identificado.

Los requerimientos funcionales y de rendimiento que presentan las aplicaciones OLAP son completamente diferentes de las que se dan en los sistemas orientados al procesamiento de transacciones on-line (on-line transaction processing) OLTP.

Las aplicaciones de tipo OLTP son fundamentalmente de transacciones cortas, atómicas y aisladas. Además estas transacciones requieren datos detallados y al día y afectan fundamentalmente a pocos registros a los cuales se accede fundamentalmente a través de la llave primaria. Las bases de datos sobre las que se opera son de gran tamaño (cientos de megabytes o algunos gigabytes) y por tanto la consistencia y capacidad de recuperación de las mismas son de vital importancia además del criterio esencial de rendimiento es optimizar la gestión de transacciones. Por todo ello la base de datos se diseña para estos objetivos.

Por el contrario los Data Warehouse están orientados al soporte de decisiones. Los datos importantes son entonces aquellos que están consolidados, resumidos y de tipo histórico. Puesto que los Data Warehouse contienen datos consolidados, quizás provenientes de distintas bases de datos y que cubren largos periodos de tiempo es de esperar que sean mucho mayores que las bases de datos clásicas (cientos de gigabytes o terabytes). El acceso a esta información se realiza mediante la visión de los datos que dan los servidores OLAP. El trabajo habitual de estos sistemas se centra en la resolución de consultas complejas, fundamentalmente ”ad hoc”, que pueden involucrar a millones de registros e implicar una serie de reuniones, búsquedas y agregaciones. Por ello el procesamiento de consultas y el tiempo de respuesta son más importantes que el tráfico de transacciones.

En la tabla 2.4 viene recogidas de forma esquemática las principales diferencias entre ambos enfoques.

Un sistema OLTP procesa un número muy elevado de transacciones por día. Cada transacción contiene una pequeña porción de datos. Un Data Warehouse a menudo procesará una transacción por día. Pero esta transacción contiene miles o millones de registros. Más que llamarlo transacción se debería denominar carga de datos productivos. En el caso de los servidores OLAP, estos pueden dar respuestas a un número pequeño de consultas que pueden necesitar el trabajar con información resumida.

Aspecto	**Base de datos clásica (OLTP)**	**Data Warehouse**
Usuarios	Diseñadores, DBAs, Operadores de entrada de datos	Decisores, ejecutivos
Función	Operaciones diarias (on-line)	Soporte de decisiones, procesamiento analítico
Diseño	Orientado a aplicaciones	orientado al usuario
Datos	Actuales, atómicos, relacionales, aislados	Históricos, resumidos, multidimensionales, integrados
Uso	Repetitivo, rutinario	ad-hoc
Acceso	Lectura/escritura, transacciones simples	Lectura, consultas complejas
Necesidades	Gestión de transacciones, datos consistentes	Gestión de consultas, datos ajustados (organizados)

Cuadro 2.4: Principales diferencias entre sistemas OLTP y OLAP

Lo que cuidamos es la consistencia del sistema cuando se comienza y cuando termina dicha carga, y el estado de consistencia del sistema una vez terminada una carga correcta. Si se forzase a parar una de estas cargas antes de estar completada, no tenemos que cargar el sistema registro a registro, sino que más bien se sobrescribe todo el sistema con una foto del sistema tomada antes de que la carga comenzase.

Muchas son las diferencias como hemos visto. Además se establece otra que son los usuarios que caracterizan a cada tipo de sistema. Los usuarios de un sistema OLTP forman la base de la empresa y se puede decir que llevan las ruedas de la empresa. Normalmente necesitan datos poco complejos y concretos, requiriendo consultas de pocos registros. Estos usuarios realizan las mismas tareas una y otra vez muchas veces. Estos usuarios se sitúan normalmente en la base de la empresa, empleados que tan sólo acceden a información precisa que afecta muy poco a políticas de producción y mercado.

La administración de un sistema OLTP se centrará más en términos como prestaciones y fiabilidad del mismo. Este administrador debería sentirse como si estuviese supervisando la maquinaria principal de la compañía. Si el sistema OLTP se para, la compañía se para.

Los usuarios de un Data Warehouse, por otro lado, están vigilando el funcionamiento de la organización. Estos vigilan qué datos son nuevos, y estudian los datos erróneos para corregirlos. Estos usuarios normalmente nunca tratan con un registro o cuenta en un momento dado. Más bien realizan consultas que implican informes más o menos grandes, y para realizarlas se requiere que grandes cantidades de registros sean buscados y resumidos en una pequeña respuesta. Además, siempre están modificando el tipo de consultas que quieren realizar a la base de datos. Normalmente los usuarios de este tipo de sistemas están en la parte alta de una empresa, es decir, aquélla en la que se toman las decisiones importantes para el futuro de la compañía (figura 2.6), actuando el data warehouse como repositorio

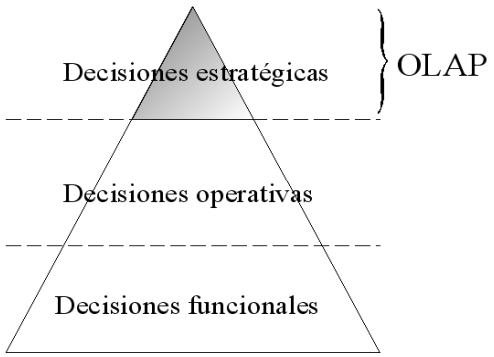


Figura 2.6: Pirámide de decisiones empresarial

de conocimiento para la gestión del conocimiento en la empresa ([Chu04]).

2.2.3. El modelo multidimensional

Veamos a continuación las razones que llevaron a construir un modelo nuevo para realizar un sistema OLAP, para posteriormente exponer las principales características y operaciones del mismo.

2.2.3.1. Razones para su creación

Desde su aparición en los 80, las bases de datos relacionales revolucionaron todo. Su idea principal era poder acceder a cualquier dato en cualquier momento. Hasta entonces los sistemas jerárquicos forzaban a los usuarios a realizar las consultas de una forma fija, y los sistemas relacionales liberaban un poco ese acceso.

En sus inicios no tuvo muchos problemas para su desarrollo, pero se descubrió que su punto débil era el procesamiento de transacciones, el cual era muy

lento. Con el tiempo y el gran esfuerzo de los investigadores este problema se pudo mejorar, pero el avance de estos sistemas ha producido una fijación en usar la metodología relacional o de transacciones en todo tipo de sistemas.

El problema se concreta cuando se sigue con la idea de querer diseñar un Data Warehouse con herramientas OLTP. Estos sistemas, como puede ser el relacional, no están preparados para la finalidad que tiene un Data Warehouse y OLAP ([Tho97]). Como reconoce Codd en su artículo ([Cod93]), el modelo relacional cumple con las expectativas que despertó en su momento en lo referente al almacenamiento y acceso eficiente de los datos, pero no para el tipo de consultas y análisis de un sistema como OLAP. Además, estos sistemas deben de dar respuestas en un tiempo más o menos corto dado que están pensados para trabajar de forma interactiva con el usuario (p.e. refinando resultado obtenidos de consultas previas).

Por todo esto, se hace necesario un nuevo modelo de datos para dar soporte a estos sistema: el modelo multidimensional.

2.2.3.2. Estructura

Una vez vista la necesidad de este nuevo modelo, pasemos ahora a ver su estructura y operaciones para entender mejor las facilidades que presta para el análisis de datos. Aunque no existe un modelo estándar, existen una serie de características comunes a las diferentes propuestas. Posteriormente, el Sección 3.2 presentaremos una formalización de un modelo multidimensional.

En un modelo de datos multidimensional hay un conjunto de hechos o medidas que son el objeto de nuestro análisis: ventas, presupuestos, inventario, etc. La mayoría de los hechos más útiles son numéricos, continuamente valuados y aditivos. La razón de que sean de dichas características es la siguiente: las consultas a la tabla de hechos necesitarán a su vez consultas de cientos, miles o incluso millones de registros para construir el conjunto respuesta. Esta gran cantidad de

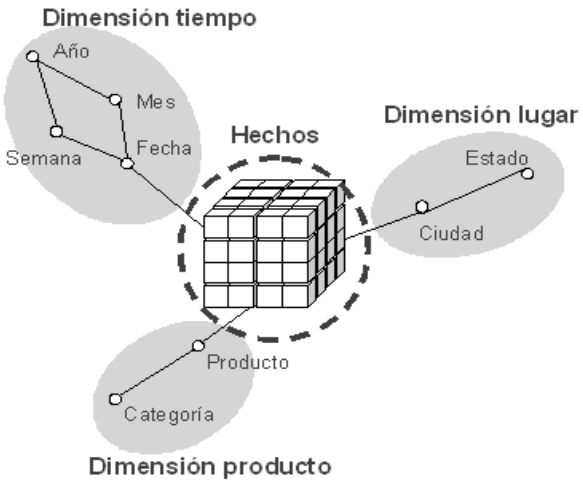


Figura 2.7: Ejemplo de estructura multidimensional

registros será resumida en unas cuantas docenas de registros y serán necesarias operaciones sobre los datos: agregados.

Cada medida depende de un conjunto de Dimensiones, que proporcionan el contexto. En un espacio multidimensional para designar un punto son necesarias todas sus coordenadas, dependiendo del número de dimensiones que tenga el espacio. Pues esto es lo que ocurre en este modelo, ya que al indicar un valor para cada dimensión se situará un hecho concreto dentro del espacio formado. En la figura 2.7 se recoge un ejemplo.

En esta figura vemos que el modelo tiene tres dimensiones, formando un cubo. A los modelos con n dimensiones se les denomina hipercubos, y por esta razón a todos los modelos construidos se les denomina cubos. En la figura tenemos las dimensiones: Producto, Tiempo y Lugar. Por tanto, si se estuviesen estudiando las Ventas como medida, para cada combinación de los valores de dichas dimensiones puede ser que existan valores para dicha medida. No para todas las coordenadas

existirán hechos, dado que habrá combinaciones que pueden no presentarse nunca.

Por tanto, como este factor va a estar presente en muchos de los cubos que se construyan en cualquier entorno, será necesario un buen tratamiento de lo que se denomina matrices dispersas, es decir, matrices n-dimensionales que tengan muchos huecos en su estructura.

Sobre las dimensiones se pueden definir jerarquías. Estas lo que van a permitir es acceder a los datos a diferentes niveles de detalle, es decir, ver los datos a diferentes granularidades. En el ejemplo de figura 2.7, hemos definido una jerarquía con dos niveles en la dimensión Producto: producto, propiamente dicho, y categoría; de tal forma que podríamos acceder a los hechos o a nivel de producto o agrupándolos según las categorías a las que pertenecen.

Está claro que para cada nivel de la estructura jerárquica formada en cada dimensión serán necesarios cálculos para estudiar las medidas en cada uno de ellos. Esto es, será necesario el uso de agregados para estudiar los hechos en los diferentes niveles de jerarquía. Sería muy útil que existan diferentes tipos de agregados que puedan ser usados: suma, media, número de registros, etc.

2.2.3.3. Operaciones

Es precisamente esta estructura jerárquica en cada dimensión la que servirá de base para definir todas las operaciones que se puedan realizar sobre el modelo.

Movimiento en la estructura jerárquica: Para moverse en los diferentes niveles de jerarquía de una dimensión se definen las siguientes operaciones:

- Roll-up: significaría subir en la jerarquía, esto es, aumentar el tamaño del grano al que están definidos los hechos. Al aplicar esta operación necesitamos resumir información para adaptar el nivel de detalle de los hechos. En esta proceso de resumen utilizaremos operadores de agregación.

- Drill-down: Esta operación es justo la contraria a la anterior. Ahora lo que pretendemos es reducir el nivel de grano, obteniendo un mayor nivel de detalle. Esto se traduce en cambiar el nivel de definición de los hechos a niveles inferiores de las jerarquías.

Un ejemplo de la traducción de estas operaciones sobre un esquema multidimensional viene recogida en la figura 2.8.

Operaciones de selección y proyección: Muchos análisis pueden que no dependan de todos los valores (selección) o de todas las dimensiones (proyección) que consideramos. Las operaciones que se encargan de esta funcionalidad son slice y dice:

- Slice: esta operación consiste en reducir la dimensionalidad del esquema eliminando alguna dimensión. Aplicar esta operación implica pérdida de detalle en los hechos (aumentamos la granularidad) por lo que tendremos que utilizar operadores de agregación para la obtención de los nuevos (figura 2.9).

- Dice: es este caso, lo que hacemos es restringir los valores que consideramos en las dimensiones según alguna condición. No modificamos la estructura del datacubo en cuanto a las dimensiones y niveles de éstas, sino restringiendo los valores que consideramos. En este caso, no modificamos el nivel de detalle de los hechos pero sí su número, dado que aquellos que tuvieran como coordenadas valores que no consideramos debemos de eliminarlos.

Operación de pivotaje: Esta operación lo que persigue es la modificación de la definición de la estructura de los datacubos. Lo que implica es un cambio en el orden de las dimensiones. Un ejemplo viene recogido en la figura 2.10.

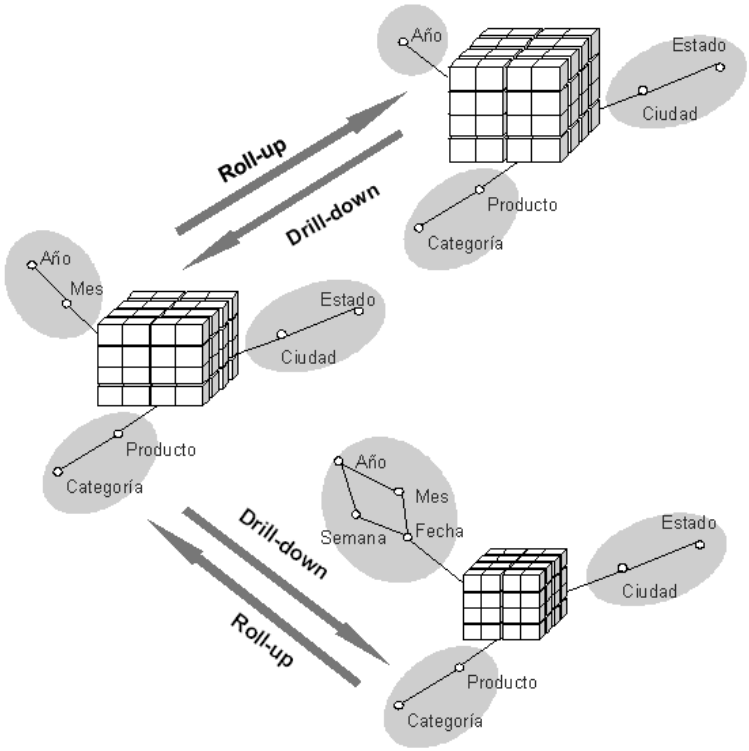


Figura 2.8: Ejemplo de aplicación de las operaciones roll-up y drill-down sobre el datacubo de la figura 2.7

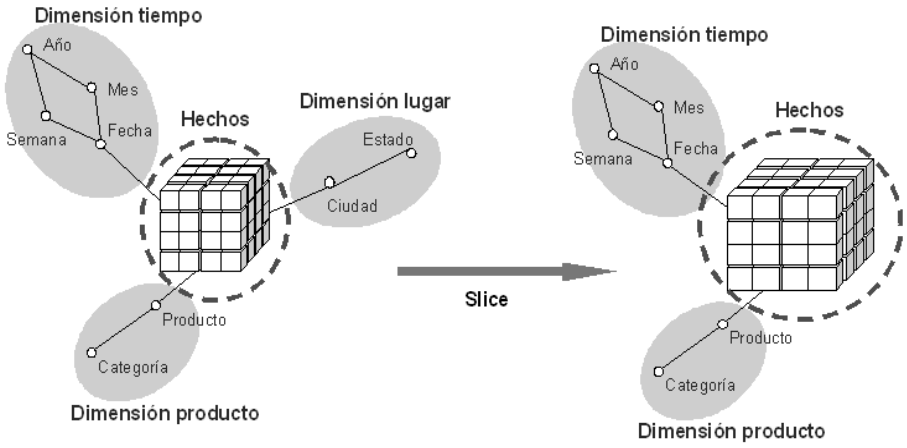


Figura 2.9: Ejemplo de aplicación de slice sobre el datacubo de la figura 2.7

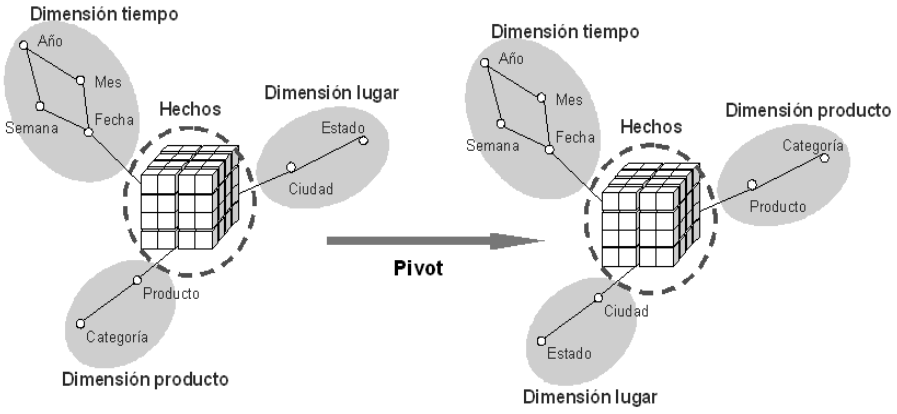


Figura 2.10: Ejemplo de aplicación de pivot sobre el datacubo de la figura 2.7

2.2.3.4. Modelos de implementación multidimensional

Existen diversas formas de representar el modelo multidimensional en un esquema físico. Se estructuran en dos tipos principalmente: esquemas ROLAP y MOLAP.

Servidores ROLAP: El modelo de datos multidimensional se puede implementar en servidores relacionales, dando lugar a lo que se conoce como Relational-OLAP servers (ROLAP), representando en ellos tanto el modelo como sus operaciones transformándolo todo en estructuras relacionales (tablas y relaciones). Veamos cuáles son los diferentes esquemas de bases de datos relacionales que reflejan esas visiones multidimensionales.

La mayoría de los Data Warehouses usan un esquema en *estrella* ([Kim96]) para representar los datos multidimensionales (figura 2.11). La base de datos que se implementa consiste en una tabla simple de hechos y una tabla para cada dimensión. Cada tupla de la tabla de hechos consta de un puntero (normalmente llave exterior) a cada dimensión, lo que proporciona las coordenadas multidimensionales y los datos de las medidas que el modelo considere (cantidad, costos, etc).

Los esquemas en estrella no proporcionan explícitamente soporte para las jerarquías de atributos. Los esquemas *en copo de nieve* ([Kim96]) tal como se muestra en la misma figura son un refinamiento de los esquemas en estrella que proporcionan dicho soporte y además suponen que las tablas están normalizadas. El único inconveniente es que suponen el manejar mayor número de tablas (en las consultas se han de realizar un número mayor de *joins* entre tablas). El esquema en estrella puede generalizarse con la inclusión de distintas tablas de hechos que comparten tablas de dimensiones, es lo que se denomina *constelaciones de hechos*.

Adicionalmente a las tablas de hechos y dimensiones, los sistemas pueden

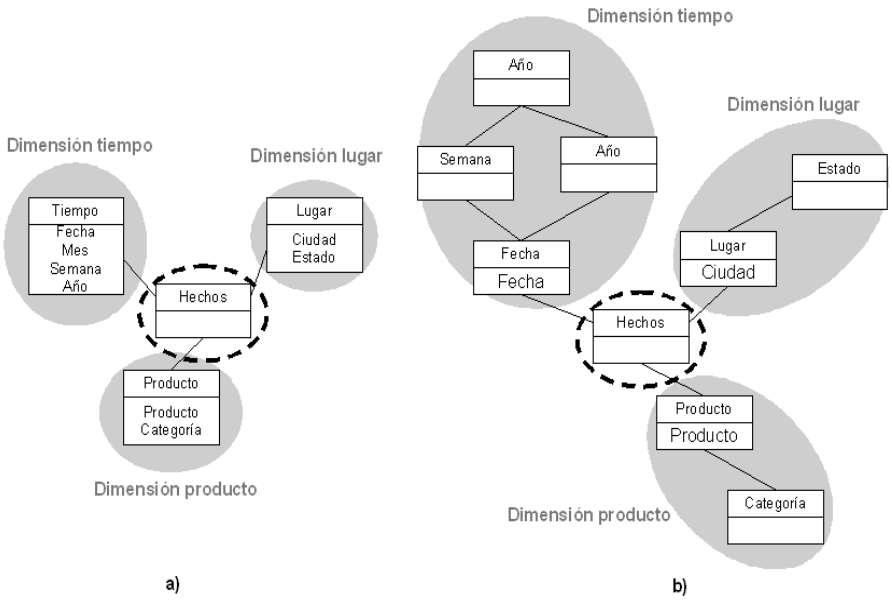


Figura 2.11: Implementación del esquema de la figura 2.7 utilizando a) modelo en estrella b) modelo en copo de nieve

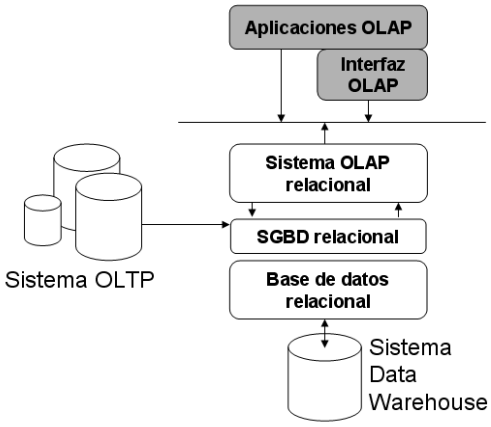


Figura 2.12: Arquitectura de un data warehouse para sistemas ROLAP

almacenar tablas resumen que almacenan datos preagregados. En los casos más simples, los datos preagregados corresponden a la agregación de una tabla de hechos sobre una o más de sus dimensiones.

Según Wu y Buchman ([WB97]), en estos sistema la arquitectura del data warehouse sería la recogida en la figura 2.12. Como puede verse, tanto la capa de almacenamiento como la de manejo de datos utilizan un enfoque relacional. La capa del *sistema OLAP relacional* proporciona un acceso multidimensional a los datos subyacentes, proporcionando operaciones multidimensionales a los usuarios. Ejemplos de este tipo de sistemas son OLAP4ALL ([OLAb]) y DBMiner ([dbm]).

Servidores MOLAP: En contraste con ROLAP, el OLAP multidimensional (MOLAP) es un modelo de datos de propósito especial y las operaciones se hacen directamente.

En lugar de almacenar la información como registros y los registros como

tablas, las bases de datos multidimensionales almacenan los datos como matrices. Las bases de datos multidimensionales (MDDs) son capaces de proporcionar un rendimiento de consulta muy alto lo que se consigue anticipando y restringiendo la forma en que se accede a los datos. En general, la información en una base de datos multidimensional es de una granularidad más gruesa que la que se considera en una base de datos relacional estándar, y por tanto los índices asociados son menores y pueden residir en memoria. Una vez que el índice se analiza, se capturan algunas páginas de las bases de datos, algunas herramientas están diseñadas para trabajar con estas páginas en memoria compartida lo que aumenta aún más el rendimiento. Otro aspecto muy interesante de las bases de datos multidimensionales es que la información está físicamente almacenada en arrays lo que significa que los valores de las celdillas se pueden actualizar sin que afecte a los índices. Como inconveniente se tiene que el menor cambio en las dimensiones obliga a estructurar toda la base de datos.

Otra ventaja de MOLAP con respecto a ROLAP es la posibilidad de tener operaciones OLAP nativas, con un servidor de bases de datos multidimensional es fácil pivotar la información, realizar drill-down y en general realizar cálculos complejos que involucren a distintas celdas. No hay necesidad de recurrir a reuniones complejas, subconsultas y uniones y no hay que sufrir los problemas que tales operaciones relacionales conllevan ([Col96]). Estos problemas se obvian dado que los datos están almacenados en una estructura multidimensional en lugar de en tablas que tratan de recrearla. En la literatura se pueden encontrar diferentes trabajos que tratan la mejora de la eficiencia para la realización de las consultas ([KLKL98, ZDN97, AAD+96, BBK+00]).

Los modelos MOLAP pueden almacenar eficientemente dimensiones múltiples utilizando una tecnología de almacenamiento de matrices poco densas (*sparse-matrix technology*). La idea básica es eliminar en lo posible celdillas vacías ya que probablemente no todas las celdas de un hipercubo contendrán datos. Se ob-

tienen entonces submatrices que se almacenan de forma comprimida. Si bien no todos los sistemas MOLAP proporcionan esta tecnología aquellos que la usan consiguen un alto rendimiento. Ejemplos de estas soluciones son la utilización de árboles para indexar los hechos especialmente concebidos para el manejo de información multidimensional ([Gut84, TS94, CM99, LJF94, KS98, KF94, CM98, EKK00]). Otros enfoques seguidos buscan mejorar el almacenamiento de las matrices dispersas para reducir las lecturas necesarias y mejorar los tiempos de acceso ([SS94]) o aumentar la densidad de las matrices multidimensionales desde el diseño del esquema multidimensional ([NNT03]).

Los servidores MOLAP existentes no tienen muchos elementos en común. A diferencia del modelo relacional no existe un acuerdo sobre lo que debe ser un modelo de datos MOLAP y tampoco hay un método estándar de acceso tales como lenguajes de consultas o API's. En este último sentido, el OLAP Council ([OLAa]) propuso el MDAPI (API para el modelo multidimensional), con tan poca acogida que ningún sistema llegó a implementarlo. Mayor éxito han tenido dos propuestas posteriores:

- JOLAP ([JOL]): Se trata de un API desarrollado en Java para la creación y mantenimiento de datos y metadatos en OLAP. Está desarrollado por un grupo de empresas para que fuera independiente de los sistemas que los implementen. Algunas empresas, como Oracle e Hyperion, incorporan esta API, aunque con ciertas modificaciones para adaptarlas a las funcionalidades de sus productos.

- XMLA (XML for Analysis, [XML]): Consiste en la especificación de un estándar para la interfaz de servicios web diseñada específicamente para operaciones de tipo OLAP y de extracción de conocimiento. Esta iniciativa surgió de la colaboración entre Hyperion y Microsoft, a la que posteriormente se unió SAS.

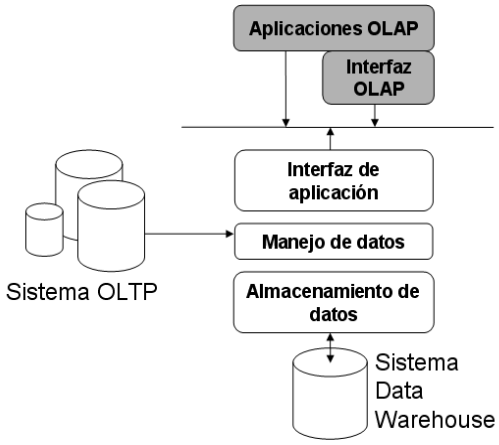


Figura 2.13: Arquitectura de un data warehouse para sistemas MOLAP

Aun existiendo diferentes propuestas, sigue sin haber un estándar aceptado de forma extendida como interfaz para estos sistemas.

En este caso, la arquitectura de un data warehouse según la filosofía MOLAP propuesto por Wu y Buchman ([WB97]) sería la recogida en la figura 2.13. Como puede verse, se utiliza directamente un almacenamiento multidimensional, por lo que las aplicaciones acceden directamente al sistema sin necesitar una interfaz de conversión como en el caso ROLAP. Ejemplos de estos sistemas son PowerOLAP ([Pow]), ShowCase Essbase ([Sho]) e Hyperion BI Platform ([Hyp]).

Servidores HOLAP: Los enfoques anteriores fueron los primeros en surgir. Posteriormente lo que se pensó fue en unir ambos enfoques de tal manera que se puedan aprovechar las ventajas de ambos enfoques. Los sistemas que siguen esta filosofía se denominan OLAP híbrido o HOLAP (Hybrid OLAP).

Esta hibridación entre los enfoques se puede ver desde diferentes puntos de vista. Por un lado algunos autores consideran esta hibridación a la hora de mane-

jar matrices dispersas ([KL03]). El enfoque MOLAP es más eficiente cuando se trata de consultar matrices densas. De esta forma, se utilizaría un esquema MOLAP para almacenar las regiones densas definidos en cada datacubo. Las dispersas serían almacenadas utilizando un esquema ROLAP.

Otra manera de realizar la hibridación (según se recoge en [ALTK97]) sería considerar la estructura multidimensional como una caché de los datos almacenados en la base de datos relacional. De esta manera, al resolver una consulta OLAP se consultaría las información mantenida en la caché. Si no es posible responderla, el sistema construiría la consulta SQL necesaria para recuperar la información de la base de datos relacional. Este enfoque trata de aprovechar el almacenamiento eficiente de grandes cantidades de información en bases de datos relacionales, y la eficiencia de las consultas sobre modelos multidimensionales puros (consiguiendo una mejora de eficiencia respecto a los sistemas ROLAP puros).

Dinter *et al.* ([BDH99]) recogen una definición genérica de estos sistemas:

Definición 2.27 ***(Dinter et al., [BDH99])*** *un sistema HOLAP soporta (e integra) el almacenamiento de los datos multidimensional y relacional de una manera equivalente con el objetivo de beneficiarse de las correspondientes características y técnicas de optimización.*[3]

Según estos autores, estos sistemas deben de cumplir unas características para considerarse realmente HOLAP:

- Transparencia respecto a los sistemas MOLAP y ROLAP subyacentes. Los sistemas HOLAP integran subsistemas MOLAP y ROLAP. Estos sistemas deben de esconder las peculiaridades de cada uno, siendo invisible para el

[3] *"We define a hybrid OLAP system as a system which supports (and integrates) multidimensional and relational storage for data in an equivalent manner in order to benefit from the corresponding characteristics and optimization techniques"*

usuario el tipo de almacenamiento que se utilice (relacional o multidimensional).

- Modelos de datos comunes y esquema multidimensional global. El sistema debe de proporcionar un modelo de datos y un lenguaje de consulta común.

Además, establecen otras características en el caso de que estos sistemas sean distribuidos.

Los sistemas comerciales han evolucionado siguiendo la filosofía HOLAP. Algunos ejemplos son SQL Server de Microsoft ([Sal98, sql]), Oracle Express ([Ora]), SAS OLAP Server ([SAS]) y DB2 OLAP Server de IBM ([db2]).

Servidores O3LAP: Una cuarta vía de implementación de estos sistemas es la utilización de sistemas de bases de datos basados en objetos o incluso objetos persistentes proporcionados por lenguajes que siguen este paradigma de programación. Estos sistemas son conocidos bajo las siglas O3LAP: Object-Oriented OLAP.

Según Buzydlowski *et al.* ([BSH98]) este enfoque presenta ventajas tanto a nivel de implementación (nivel físico) y conceptual. En el nivel físico las ventajas que presentan serían:

- Existen estándares para las bases de datos orientadas a objetos, como la propuesta por ODMG (Object-Oriented Database Management Group).

- En el campo de las bases de datos orientadas a objetos se han realizado numerosas investigaciones, de forma que algunos problemas como la autorización de usuarios o la actualización de la base de datos mediante transacciones han sido estudiados.

- Algunas características fácilmente implementables por BDOO pueden ser muy útiles a la hora de desarrollar data warehouses (p.e. la gestión de ver-

siones de objetos puede utilizarse para manejar los cambios de las dimensiones o hechos a lo largo del tiempo). También la existencia de estándares de objetos distribuidos (p.e. CORBA) puede ayudar a su implementación.

En el modelado conceptual, principalmente se refieren a la posibilidad que dan los objetos para trabajar con diferentes tipos de datos dentro de un modelo multidimensional. Dado que los objetos encapsulan tanto datos, visualización y métodos de manipulación, se podría incluir información como imágenes u otros tipos de información estructurada de forma simple.

Siguiendo este enfoque, existen propuestas de modelos multidimensionales que utilizan esta filosofía de orientación a objetos ([TP98, NTW00]). Sin embargo, no conocemos la existencia de ningún sistema comercial que utilice este paradigma.

2.2.4. Modelos multidimensionales clásicos

Una vez que hemos presentado de forma genérica el modelo multidimensional, presentaremos brevemente algunas de las principales formalizaciones que se han realizado. Una revisión de diferentes modelos lógicos para OLAP puede encontrarse en [VS99]. Estos modelos no contemplan la imprecisión. En la siguiente sección, introduciremos algunos de los modelos propuestos que tratan en alguna de sus facetas la imprecisión y/o incertidumbre en los datos.

En el capítulo 3 presentamos una formalización del modelo multidimensional en el que se define en detalle tanto la estructura como las operaciones sobre la misma.

2.2.4.1. Modelo de Agrawal *et al.*

Propuesto en [AGS95], fue uno de las primeras formalizaciones que se ha hecho de un modelo multidimensional. Se propone un modelo con la siguiente

funcionalidad:

- Tratamiento simétrico no sólo de todas las dimensiones sino también de las medidas.
- Soporte para múltiples jerarquías en cada dimensión.
- Soporte para el cálculo de agregados ad-hoc, es decir, no se limitan los agregados a aquellos predefinidos.
- Soporte para un modelo de consulta.

Para dar respuesta a esta funcionalidad, proponen un modelo de datos donde un DataCubo se define como sigue.

Definición 2.28 *([AGS95]) Un DataCubo C es una estructura con la siguiente estructura:*

- *k dimensiones, y para cada dimensión un nombre D_i, un dominio dom_i desde el cual se toman los valores.*
- *Elementos definidos como una aplicación $E(C)$ desde $dom_i \times ... \times dim_k$ a un n-tupla, 0 ó 1. Con esto, $E(C)(d_1, ..., d_k)$ se refiere al elemento en la posición $d_1, ..., d_k$ del DataCubo C. El valor 0 representa que esa posición no tiene definidos valores, 1 si la combinación existe pero no sabemos nada mas, y la tupla si tenemos información adicional.*
- *Una n-tupla con los nombres describiendo cada uno de los elementos de las n-tuplas que define el DataCubo.*

Sobre esta estructura definen, aparte de las operaciones habituales, introducir una dimensión como hecho (*push*), sacar un hecho para ser tratado como una

dimensión (*pull*), y la combinación de datacubos (*join*). Este conjunto de operaciones es mínimo y cerrado (el resultado es siempre un datacubo). El modelo no define de forma explícita las jerarquías en la dimensiones, sino mediante operaciones que agrupan los valores en las dimensiones.

2.2.4.2. Modelo de Li y Wang

Loa autores proponen ([LW96]) una formalización de un modelo multidimensional como la relación de múltiples dimensiones con los hechos. Las jerarquías en las dimensiones, como en el modelo anterior, no aparecen de forma explícita sino mediante agrupaciones de valores por otros.

Un DataCubo es considerado instanciación de un *esquema n-dimensional de cubo*[4].

Definición 2.29 *([LW96]) Sea n un entero positivo y $\mathcal{V}$ un conjunto de valores escalares. Un* esquema n-dimensional de cubo *es un conjunto $\{(D_1, R_1), ..., (D_n, R_n)\}$, donde $D_1, ..., D_n$ son nombre distintos de dimensiones y $R_1, ..., R_n$ son son conjuntos de nombres de atributos.*

Un cubo n-dimensional *sobre el esquema $\{(D_1, R_1), ..., (D_n, R_n)\}$ es un par (F, μ), donde*

- $F = \{(D_1, r_1), ..., (D_n, r_n)\}$ *con r_i siendo una relación en R_i para cada $1 \leq i \leq n$,*
- *y μ es una aplicación de $\{(D_1, t_1), ..., (D_n, t_n) | \forall 1 \leq i \leq n : t_i \in r_i\}$ a $\mathcal{V}$.*

Para completar el modelo, presentan una extensión del álgebra relacional, denominada *grouping algebra*. En ella, se incluyen operaciones que requieran ordenaciones (*order-oriented operations*), es decir, divisiones del dominio en intervalos según un orden dado, y agregaciones (*aggregation operations*).

[4] *n-dimensional cube scheme*

2.2.4.3. Modelo de R. Kimball

Este modelo ([Kim96]) se trata más de un modelo de implementación que de un modelo conceptual. Kimball propone la construcción de un modelo multidimensional utilizando para ello un esquema relacional. En este, los hechos corresponderían a una tabla y cada dimensión a otra con una relación de llave externa con los hechos. Este es el modelo de implementación *en estrella* que hemos presentado en la sección 2.2.3.4 al hablar de los servidores ROLAP.

En este modelo, el movimiento en las jerarquías se realiza utilizando operaciones GROUP BY relacional con los operadores de agregación habituales (p.e. suma, máximo, mínimo, etc.).

2.2.4.4. Modelo de Gray *et al.*

En [GCB+97] no se presenta como tal un modelo multidimensional sino que se extiende las operaciones de agregación tipo *group by* de los sistemas relacionales para el caso de datos multidimensionales. Las operaciones que se proponen son:

- **ROLLUP**: se trata de una extensión de la operación *group by* en el que se realiza la agregación según n dimensiones ordenadas aplicando el esquema siguiente:

  GROUP BY *n-dimensiones* $\cup$ GROUP BY *n-1 primeras dimensiones* $\cup$
  GROUP BY *n-2 primeras dimensiones* $\cup$ ... $\cup$ GROUP BY *1 dimensión*

- **CUBE**: se trata de aplicar aplicar la operación anterior (*rollup*) sobre todas las dimensiones, es decir, el resultado es la unión de aplicar rollup sobre un conjunto de dimensiones. El resultado que se obtiene es la agregación de cada posible subconjunto de las dimensiones que consideremos.

2.2.4.5. Modelo de Cabibbo y Torlone

En este modelo ([CT97, CT98]) se presenta un modelo multidimensional (llamado $\mathcal{MD}$) con dos partes muy diferenciadas:

- Dimensiones: que representan categorías lingüísticas que corresponden a las diferentes aproximaciones a los datos.
- *f-tables*: que representan a los hechos y se relacionan con las dimensiones. Se definen como funciones que relacionan coordenadas simbólicas con las medidas.

Los autores definen una dimensión con diferentes niveles de la siguiente manera.

Definición 2.30 *([CT97]) Una dimensión $\mathcal{MD}$ está formada por:*

- *un conjunto finito de niveles $L \subseteq \mathcal{L}$, donde $\mathcal{L}$ es un conjunto de nombres para los niveles;*
- *un orden parcial $\leq$ sobre los niveles en L, donde si $l_1 \leq l_2$ decimos que l_1 es agrupado por l_2;*
- *una familia de funciones, incluyendo la función $R-UP_{l_1}^{l_2}$ desde $DOM(l_1)$ a $DOM(l_2)$ que para cada par de niveles $l_1 \leq l_2$, donde si $R-UP_{l_1}^{l_2}(o_1) = o_2$ significa que o_1 es agrupado por o_2.*

En este modelo, como puede verse, se define de forma explícita las jerarquías en las dimensiones, estableciendo una ordenación entre los niveles mediante un orden parcial y funciones (*R-UP*$_{l_1}^{l_2}$) que establecen la relación entre los elementos de estos niveles.

Definición 2.31 *([CT97]) Un esquema $\mathcal{MD}$ está formado por:*

- *un conjunto finito D de dimensiones;*
- *un conjunto F de esquemas $f - tables$ de la forma $f[A_1 : l_1 < d_1 > , ..., A_n : l_n < d_n >] : l_0 < d_0 >$, donde f es un nombre, cada A_i ($1 \leq i \leq n$) es un nombre distinto llamado* atributo*, y cada l_i ($0 \leq i \leq n$) es un nivel de la dimensión d_i;*

Para el modelo proponen un lenguaje de consulta (*MD-CAL*), presentado en [CT97], contemplando las funciones de agregación para realizar el cambio de nivel de detalle.

2.2.4.6. Modelo de Datta y Thomas

El modelo propuesto ([DT99]) hace una clara diferenciación entre los hechos y las dimensiones.

Definición 2.32 *([DT99]) Un DataCubo es una 6-tupla, $< D, M, A, f, V, g >$ donde los cuatro componentes son:*

- *Un conjunto de n dimensiones $D = \{d_1, ..., d_n\}$, donde cada d_i es un nombre dimensión extraído del dominio de dimensiones ($dom_{dim(i)}$);*
- *un conjunto de k* medidas *$M = \{m_1, ..., m_k\}$ donde cada m_i es una medida extraída del dominio de medidas ($dom_{measures(i)}$);*
- *El conjunto de dimensiones y medidas es disjunto ($D \bigcap M = \emptyset$);*
- *un conjunto de t atributos $A = \{a_1, ..., a_t\}$ donde cada a_i es un atributo extraído del dominio de atributos ($dom_{attr(i)}$);*
- *una aplicación uno-a-muchos $f : D \rightarrow A$, tal que los conjuntos de atributos correspondientes a dos dimensiones son disjuntos ($\forall i, j \; i \neq j, f(d_i) \bigcap f(d_j) = \emptyset$);*

- *V un conjunto de tuplas de tamaño k tal que $v_i =< \mu_1, ..., \mu_k >$, donde cada μ_i es una instancia de la medida i-ésima;*
- *g es una aplicación $g : dom_{dim(i)} \times ... \times dom_{dim(n)} \rightarrow V$, que asocia a cada conjunto de coordenadas los hechos relacionados.*

Sobre esta estructura, formaliza las operaciones habituales sobre el modelo, contemplando algunas más:

- *Producto cartesiano*: un operador binario que construye un nuevo datacubo mediante la combinación de otros dos. Un caso particular del producto cartesiano es el operador JOIN que se usa cuando los datacubos tienen una o más dimensiones en común.
- *Diferencia*: este operador devuelve un datacubo cuyo contenido son las diferencias entre otros dos compatibles (elimina la parte común a ambos).

Las jerarquías en las dimensiones no aparecen de forma explícita, por lo que la agrupación de valores se realiza mediante una operación que agrupa valores (*partición*). Sobre el modelo, definen un álgebra de consulta con estas operaciones.

2.2.5. Modelos Multidimensionales con tratamiento de imprecisión

El problema de del manejo de imprecisión o incertidumbre ha sido tratada en los modelos multidimensionales desde diferentes puntos de vista. A continuación presentamos algunas de las propuestas presentes y qué problemas abordan.

2.2.5.1. Modelo de Dyreson

Propuesto por primera vez en [Dyr96], aborda el problema del cálculo de agregados cuando existen huecos en el espacio multidimensional. Define un *datacubo*

incompleto como aquel en el que se presenta este problema (en contrapartida de los *datacubos completos*).

Para operar sobre los datacubos incompletos, los define como la unión de subdatacubos completos (llamados *cubbetes*). De esta manera, define la satisfacibilidad de una consulta (posibilidad de resolver la consulta) basándose en si existe un *cubette* que puede dar respuesta a la misma. Aunque en el trabajo no se presenta el modelo multidimensional utilizado como base, el modelo usado trabaja de forma explícita con jerarquías dado que distingue entre *cubette units*, que están definidos al nivel más alto de las jerarquías, y *cubette measure*, definidos a niveles de mayor detalle.

Este modelo modela la imprecisión debida a la falta de hechos de zonas del espacio multidimensional definido. En el resto de los aspectos (hechos imprecisos y dimensiones imprecisas) no es considerado.

2.2.5.2. Modelo de Pedersen *et al.*

El modelo propuesto en [PJD01] permite la definición de hechos a diferentes niveles en cada dimensión (diferente granularidad en su definición). En este modelo una dimensión sería una estructura definida como sigue.

Definición 2.33 *([PJD01]) Un tipo dimensión $\mathcal{F}$ es una 4-tupla* $(\mathcal{L}, \sqsubseteq_{\mathcal{F}}, \top_{\mathcal{F}}, \bot_{\mathcal{F}})$ *donde*

- $\mathcal{L} = \{\mathcal{L}_j, j = 1, ..., k\}$ *son llamados* tipos categóricos *y representan los niveles;*
- $\sqsubseteq_{\mathcal{F}}$ *es un orden parcial entre los elementos de* $\mathcal{L}$*;*
- *y* $\top_{\mathcal{F}}$ *y* $\bot_{\mathcal{F}}$ *representan los elementos superior e inferior, respectivamente, según el orden establecido.*

Con esto, una dimensión D del tipo $\mathcal{F} = (\mathcal{L}, \sqsubseteq_{\mathcal{F}}, \top_{\mathcal{F}}, \bot_{\mathcal{F}})$ es una 2-tupla $D = (C, \sqsubseteq)$ donde

- *$C = \{C_j\}$ es un conjunto de categorías C_j tal que su tipo es $\mathcal{L}_j$,*
- *y $\sqsubseteq$ en un orden parcial definido sobre $\bigcup_j C_j$.*

Como puede verse, se define de forma explícita las jerarquías en las dimensiones, realizándolo mediante la utilización de un orden entre los niveles.

Los DataCubos en este modelo son llamados *objetos multidimensionales* o *MO*.

Definición 2.34 *([PJD01]) Un* objeto multidimensional *es una 4-tupla $M = (\mathrm{f}, F, D, R)$ donde*

- *$\mathrm{f} = (\mathcal{T}, \mathcal{D} = \{\mathcal{F}_i\})$ es un esquema de hechos, donde $\mathcal{T}$ es un tipo hecho y $\mathcal{F}$ es un conjunto de tipos dimensiones;*
- *F es un conjunto de hechos de tipo f;*
- *D es un conjunto de dimensiones;*
- *$R = \{R_i, i = 1, ..., n\}$ es un conjunto de relaciones donde cada R_i establece la relación entre los elementos de la dimensión D_i, definidos a cualquier nivel, y los hechos de F.*

Como puede verse, la principal aportación de este modelo radica en que los hechos se pueden definir a cualquier nivel dentro de las dimensiones.

Para evaluar un consulta, calcula la imprecisión asociada. Si se considera suficiente, resuelve la consulta. En caso contrario se sugieren consultas alternativas a la considerada. A la hora de agregar con imprecisión, resuelve los cálculos desde tres perspectivas:

- Respuesta conservadora: en este caso sólo considera los elementos que se sabe seguro que pertenecen a la agregación.

- Respuesta liberal: se consideran todos los hechos que pueden pertenecer.

- respuesta ponderada: se consideran todas los hechos que pueden pertenecer pero ponderándolos según la probabilidad de pertenencia.

Las tres repuestas son presentadas al usuario, como muestra de la imprecisión. Las operaciones que se definen sobre el modelo corresponden al roll-up, dice, slice, antes comentadas, y a la unción de dos modelos multidimensionales propuestos (obteniendo un único modelo representado la unión de ambos).

Posteriormente ([JKPT04]) se ha extendido el modelo para modelar en las dimensiones la inclusión parcial de elementos. Para ello, en la definición de los tipos dimensiones incorpora una nueva relación $\sqsubseteq_{\mathcal{F}}^{P}$ que captura la inclusión parcial de categoría o niveles (asignándole un valor en el intervalo [0,1] a dicha relación).

A la hora de realizar operaciones sobre el datacubo, propone un algoritmo para evaluar la imprecisión de los caminos dentro de la jerarquía y seleccionar aquel que presente una imprecisión menor. Todo este tratamiento de la imprecisión se hace con métodos definidos expresamente para el modelo propuesto.

En el caso de las consultas, se sigue manteniendo las mismas tres políticas de manejo de imprecisión: en la conservadora sólo se consideran los elementos con una inclusión total, en la liberal todos aquellos que puedan estar incluidos (tengan inclusión parcial o total) y la ponderada se consideran pesos para la agregación. En este último caso, el cálculo de los pesos propone hacerlos basados en las inclusiones parciales (dando como ejemplo de peso el producto de los grados de inclusión).

2.2.5.3. Modelo de Laurent

Este modelo ([Lau02, Lau03b]) trata la imprecisión en los hechos y en las dimensiones. En el primer caso, a cada hecho se le asigna un valor en el intervalo $[0, 1]$ para indicar el grado de confianza en su cumplimiento. En el caso de las dimensiones, propone utilizar relaciones difusas o particiones, también difusas, para abordar el modelado de la imprecisión. Es complicado encontrar la definición de este modelo dado que no existe ninguna publicación de carácter internacional que presente la estructura del modelo. Los detalles del mismo se pueden encontrar en [Lau02].

Más concretamente, una dimensión tendrá la siguiente estructura.

Definición 2.35 *([Lau02]) Una dimensión D es una 5-tupla $< v, X, dim(D), H_P, H_R >$ donde:*

- *v es el nombre de la dimensión.*
- *X es el universo de referencia.*
- *$dom(D)$ es el conjunto de elementos utilizados para definir la dimensión. Se denomina* dominio de la dimensión.
- *H_P es una relación de orden ($<$) de las particiones difusas definidas sobre los valores de la dimensión.*
- *H_R es una relación difusa sobre las relaciones difusas definidas.*

Con esto, un DataCubo es definido en el modelo como una aplicación $C : D_1 \times D_2 \times ... \times D_n \rightarrow D_c \times [0, 1]$ donde cada D_i es una dimensión y D_c es una medida.

Las operaciones son extendidas para manejar este valor (FUZZY ROLL-UP, FUZZY-SLICE, FUZZY-DICE y FUZZY-PROJECTION). En cuanto a los operadores

de agregación necesarios, se establecen cuáles deben ser la propiedades que tienen que cumplir para poder ser utilizados. La conmutatividad y la asociatividad de las operaciones también se estudian ([Lau03b]).

Este modelo trabaja con imprecisión introduciendo el uso de lógica difusa, lo que hace complicada la interpretación de los resultados obtenidos de las consultas dado que no se oculta al usuario. Además, maneja dos jerarquías (las definidas mediante particiones y relaciones) de manera completamente separada. En cuanto a las relación difusa de cada dimensión, se ha de definir de forma extensiva, dando el valor para cada par de valores existentes en el dominio. De esta forma, añadir un nuevo valor o concepto implicaría definir su relación con todos los ya existentes.

2.2.5.4. Modelo de Kaya y Alhajj

Los autores proponen en [AK03, KA05] un modelo multidimensional basado en lógica difusa. El modelo está pensado para modelar de forma más intuitiva la división de valores cuantitativos según intervalos difusos para evitar el problema de los intervalos con bordes abruptos.

Para cada atributo numérico x se define un conjunto de conjuntos difusos $F_x = \{f_x^1, f_x^2, ..., f_x^l\}$ asociados con x. El grado de pertenencia de un valor a cada conjunto será una función $\mu_{f_x^j} : D_x \rightarrow [0, 1]$. Basado en esta división de los atributos definen la estructura de un DataCubo.

Definición 2.36 *([AK03]) Considerando un DataCubo con n dimensiones $d_1, ...d_n$. Cada dimensión contiene $\sum_{i=1}^{k} l_i + 1$, donde k es el número de atributos de la dimensión X; l_i es el número de conjuntos difusos para el atributo x_i en la dimensión X; y "$+1$" representa un valor espacial llamado "*Total*", el cual almacena los valores agregados del resto de valores.*

En cada dimensión se pueden considerar múltiples niveles para agrupar los

definidos. En estas relaciones no se considera imprecisión. Por esto, la única ayuda que da para modelar dominios complejos es permitir definir intervalos numérico difusos. La inclusión parcial de los niveles o la imprecisión en los hechos no es considerada.

2.2.5.5. Modelo de English *et al.*

Este modelo ([EGY04]) se trata en realidad de una propuesta de modelo espacio-temporal. Cada *relación espacio-temporal* que se defina en el modelo estará formada por un 3-tupla $< T_m, L_n, V_{mn} >$ donde

- T_m es la coordenada que establece el tiempo.
- L_n es el lugar o localización.
- V_{mn} representa un conjunto vacío o los valores de los atributos no relacionados con el espacio o el tiempo para esa localización en ese tiempo en concreto.

Esta representación lo que hace es definir una array multidimensional que se puede ver como un DataCubo (MDC). De esta manera, las operaciones habituales de los sistemas OLAP se pueden utilizar sobre esta estructura. A estas operaciones le añaden cuatro operaciones propias de los sistemas de información geográfica (GIS):

- Búsqueda espacial: encontrar regiones que en un momento dado tengan valores de atributos similares a una región seleccionada.
- Búsqueda temporal: igual que la anterior pero fijando unas regiones y buscando el momento en que presenten similares valores.
- Búsqueda espacio-temporal: buscar regiones con calores similares a una escogida a lo largo del tiempo y el espacio.

- Búsqueda de tele-conexiones: encontrar regiones en el espacio a lo largo del tiempo que están conectadas de forma no directa. Esta búsqueda localiza fenómenos que ocurran en el espacio y que puedan estar relacionados con otros en otras regiones.

Las regiones son conjuntos de celdas que forman un *minimum bounding cube*. El tratamiento de la imprecisión se introduce a la hora de manejar los criterios de selección de regiones. Utilizando la *matriz de relación MBC* (MBR) almacena cómo satisface una región un determinado criterio, donde una función μ nos daría el grado en que dos regiones satisfacen la relación. Como ejemplo, utilizando estas relaciones, la búsqueda temporal, para un tiempo T_k respeto a la región R_i, tendría la expresión siguiente:

$$\Phi_S(MDC) : \{R_j | Disjuntas(R_i, R_j) \wedge ocurre(T_k) \wedge (\mu(R_i, R_j) \geq \varepsilon)\} \quad (2.25)$$

Así pues, este modelo modela imprecisión para establecer los criterios al aplicar algunas de las operaciones habituales en los sistemas GIS. No permite modelar imprecisión en los hechos o dimensiones, por lo que resulta poco útil para modelar dominios complejos.

Capítulo 3

Modelo Multidimensional Difuso

Todos los ordenadores esperan a la misma velocidad

LEY DE CRAYNE
Ley de Murphy

3.1. Introducción

Desde el nacimiento de OLAP ([Cod93]) se ha visto que las estructuras disponibles de bases de datos tenían algunas limitaciones. Desde las primeras propuestas ([GCB+97]) se ha intentado extender los sistemas existentes con la funcionalidad necesaria para dar soporte a la tecnología OLAP. Sin embargo, la necesidad de dar respuestas a consultas ad-hoc que implican agregación de gran cantidad de datos plantea problemas de eficiencia y complejidad de estructuras. Como consecuencia, nuevos modelos con una visión multidimensional de los datos aparecieron para subsanar estos aspectos ([AGS95, CT97, LW96]). Ex-

isten múltiples propuestas de modelos multidimensionales sin llegar a establecerse un estándar. En la actualidad, dada su aplicación en otros campos (p.e. datos hospitalarios) y el uso de fuentes de datos semi-estructuradas (p.e. XML) y no-estructuradas (p.e. texto plano), surgen nuevas necesidades. Entre estas cabe destacar el manejo de información imprecisa y estructuras menos rígidas que permitan un modelado y uso más intuitivo. Se han propuesto modelos para el tratamiento de diferentes aspectos de imprecisión e incertidumbre en modelos multidimensionales (en la Sección 2.2.5 se han presentado). Sin embargo, o no tratan todos los aspectos de la imprecisión o, si lo hacen, presentan problemas, sobre todo de cara a ocultar la complejidad añadida de cara a el usuario final.

En este capítulo presentaremos un modelo multidimensional basado en lógica difusa para el manejo de información imprecisa en las jerarquías y los hechos. Comenzaremos formalizando un modelo multidimensional preciso como base para extender los conceptos al uso de jerarquías imprecisas y hechos difusos. Posteriormente incorporaremos el uso de etiquetas lingüísticas en las relaciones jerárquicas, pensando en el uso de conocimiento proporcionado por expertos para enriquecer los esquemas.

Sobre estos modelos definiremos las principales operaciones de los modelos multidimensionales y probaremos sus propiedades. También introduciremos el concepto de *vista de usuario* como herramienta para ocultar la complejidad del tratamiento de la imprecisión e incertidumbre.

Una vez presentados los modelos, utilizaremos un ejemplo simple para mostrar su utilización y su comportamiento.

3.2. Modelo Multidimensional Preciso

Como hemos comentado, comenzaremos presentando un modelo multidimensional preciso. En primer lugar presentaremos la estructura del mismo. Posterior-

mente presentaremos las operaciones habituales de los sistemas OLAP (roll-up, drill-down, slice, dice y pivotaje) y su aplicación sobre la estructura propuesta.

3.2.1. Estructura Multidimensional Precisa

Comenzaremos comentando la estructura de las dimensiones y las jerarquías que podemos definir en ellas. Tras éstas, definiremos qué serán los hechos para concluir con la estructura que tendrán los DataCubos.

3.2.1.1. Dimensiones

Las dimensiones establecen el contexto del análisis sobre los hechos. Para acceder a los hechos a diferentes niveles de detalle, podemos definir jerarquías sobre la dimensiones, estableciendo los niveles de granularidad posibles. Cada uno de estos niveles será un conjunto de nombres o etiquetas que definen subconjuntos de elementos de los niveles inferiores que agrupan.

Definición 3.1 *Una* **dimensión** *sobre el dominio es una tupla* $d = (l, \leq_d, l_\perp, l_\top)$ *donde* $l = \{l_1, ..., l_n\}$ *tal que cada* l_i *es un conjunto* $l_i = \{c_{i1}, ..., c_{im}\}$ *tal que* $l_i \cap l_j = \emptyset$ *si* $i \neq j$, $\leq_d$ *es una relación de orden parcial tal que* $l_i \leq_d l_k$ *si* $\forall c_{ij} \in l_i \Rightarrow \exists c_{kp} \in l_k / c_{ij} \subseteq c_{kp}$. $l_\perp$ *y* $l_\top$ *son dos elementos de* l *tal que* $\forall l_i \in l\ l_\perp \leq_d l_i$ *y* $l_i \leq_d l_\top$.

A cada elemento l_i lo denominaremos nivel. Para identificar el nivel l_i de la dimensión d lo notaremos como $d.l_i$. Un elemento de la dimensión d_i y nivel l_j lo notaremos c^i_{jk}. Los niveles especiales $l_\perp$ y $l_\top$ los denominaremos *nivel base* y *nivel superior* respectivamente.

La figura 3.1 muestra un ejemplo de una posible jerarquía sobre las edades de las personas. La definición correspondiente sería

$$Edad = (Edad, Grupo, Mayor\ Edad, Todo, \leq_{Edad}, Edad, Todas) \quad (3.1)$$

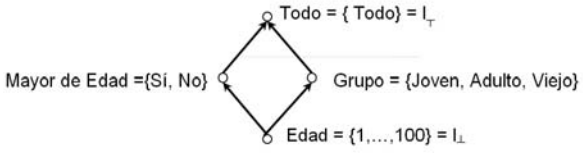


Figura 3.1: Ejemplo de jerarquía sobre las edades

y la relación se definiría como:

$$\begin{gathered} Edad \leq_{Edad} Edad, \\ Grupo \leq_{Edad} Grupo, \\ MayorEdad \leq_{Edad} MayorEdad, \\ Todo \leq_{Edad} Todo, \\ Edad \leq_{Edad} Grupo, \\ Edad \leq_{Edad} MayorEdad, \\ Edad \leq_{Edad} Todo, \\ Grupo \leq_{Edad} Todo, \\ MayorEdad \leq_{Edad} Todo \end{gathered} \tag{3.2}$$

Cada valor en los niveles representa la etiqueta de un conjunto de valores. Por ejemplo, el valor *"Sí"* del nivel *Mayor Edad* sería en realidad la definición de un conjunto de valores tal que *"Sí"*=$\{$*18,...,100*$\}$. Igualmente, el valor *"No"* del mismo nivel sería el conjunto *"No"*=$\{$*1,...,17*$\}$. En el caso del nivel *Todo*, el único valor que consideramos (*Todo*) representaría al conjunto de todas las edades (*"Todo"*=$\{$*1,...,100*$\}$), de tal manera que "Si" $\subseteq$ "$Todo$" y "No" $\subseteq$ "$Todo$". El conjunto de todos los nombres o etiquetas utilizados será lo que denominaremos el dominio de la dimensión.

Definición 3.2 *Para cada dimensión d el dominio será $dom(d) = \bigcup_i l_i$.*

En el ejemplo anterior tendríamos que *dom(Edad)* = {*1,...,100, Joven, Adulto, Viejo, Sí, No, Todo*}.

Los niveles de una dimensión están relacionados mediante el orden parcial. Esta relación establece una jerarquía entre ellos. Un nivel l_i podrá tener niveles inferiores que estén directamente conectados en la jerarquía. Es decir, niveles menores que l_i y que no exista otro nivel menor que l_i y que a su vez mayor que estos. El conjunto de los niveles que cumplan esta condición para un nivel, lo denominaremos *conjunto de hijos del nivel*.

Definición 3.3 *Para cada* l_i *el conjunto*

$$H_i = \{l_j / l_j \neq l_i \wedge l_j \leq_d l_i \wedge \neg \exists l_k \; l_j \leq_d l_k \leq_d l_i\} \tag{3.3}$$

Se denominará conjunto de hijos del nivel l_i.

Con esto, tendríamos que, por ejemplo, para el nivel *Todo* el conjunto de hijos sería $H_{Todo} = \{Grupo, MayorEdad\}$. En cualquier dimensión, dada la definición que hemos hecho del conjunto de hijos, tendríamos que para el nivel base su conjunto de hijos sería siempre el conjunto vacío.

Igualmente tendremos niveles inmediatamente superiores en la jerarquía. En este caso, al conjunto de niveles superiores directamente relacionados con otro lo denominaremos *conjunto de padres* de dicho nivel.

Definición 3.4 *Dado* l_i

$$P_i = \{l_j / l_i \neq l_j \wedge l_i \leq_d l_j \wedge \neg \exists l_k \; l_j \leq_d l_k \leq_d l_i\} \tag{3.4}$$

y se denominará conjunto de padres del nivel l_i.

Sobre la jerarquía de edades definiríamos el conjunto de padres del nivel *Edad* como $P_{Edad} = \{MayorEdad, Grupo\}$. En el caso del nivel superior de la jerarquía estaríamos en una situación análoga al conjunto de hijos para el nivel base. Es decir, el conjunto de padres será el conjunto vacío.

3.2.1.2. Hechos Precisos

Las variables del dominio que queremos analizar definirán los hechos del DataCubo. Éstos serán los que en mayor medida acotarán el dominio de análisis del DataCubo que definamos, limitando qué medidas podemos obtener. Las dimensiones aportan la contextualización de estas variables, localizando cada hecho dentro del espacio que definen. Cada uno de estos hechos será una agrupación de variables (una tupla).

Definición 3.5 *Considerando un conjunto de atributos $A_1, ..., A_n$ con dominios $D_1, ..., D_n$, denominaremos* **hecho** *a cualquier $h = (x_1, ..., x_n)$ tal que $x_i \in D_i \ \forall i = 1, ..., n$, es decir, cualquier n-tupla definida sobre los dominios de los atributos que interesa estudiar.*

Suponiendo que quisiéramos un DataCubo para analizar las ventas de una empresa, y que las variables que analizar fueran la cantidad vendida (número natural, es decir, dominio en $\mathbb{N}$) y el precio de la venta (dominio $\mathbb{R}^+$), un hecho posible sería $(1, 3'5)$ que representaría un hecho en el que el atributo *cantidad* tendría el valor 1 y el *precio* sería 3'5. Así, cualquier par de valores en el dominio $\mathbb{N} \times \mathbb{R}^+$ sería un hecho válido para este ejemplo.

Los niveles de las dimensiones que utilicemos para acceder a cada hecho, nos dará el nivel de detalle con el que estarán definidos. Al disminuir el nivel de detalle en un DataCubo (realizar roll-up), los hechos del DataCubo se han de subir a dicho nivel. Para poder realizarlo, necesitaremos operadores que nos permitan realizar esta disminución de detalle, obteniendo nuevos hechos que resuman la información al detalle de la nueva granularidad. En este proceso se utilizan operadores de agregación.

En el caso preciso, un operador de agregación será una función que, dado un conjunto de valores, devuelve un único valor representando a ese conjunto. Las funciones Max, Min, Media, Conteo, etc. son ejemplos de este tipo de operadores,

ampliamente utilizados. Estos operadores trabajan sobre bolsas de valores, dado que es posible que un mismo valor se presente múltiples veces en los valores que agregar.

Definición 3.6 *Siendo $\mathcal{B}(X)$ el conjunto de todas las posibles bolsas definidas utilizando los elementos de X, y D_x un dominio natural o numérico, definimos un operador de agregación G como una función $G : \mathcal{B}(D_x) \rightarrow D_x$.*

Siguiendo el ejemplo puesto para los hechos, si quisiéramos disminuir el nivel de detalle, necesitaríamos agregar los hechos definidos. En el caso del *precio*, necesitaríamos un operador que aceptara bolsas de números reales y cuyo resultado fuera a su vez un número real. Un operador válido sería la *Media* ($Media : \mathcal{B}(\mathbb{R}^+) \rightarrow \mathbb{R}^+$).

Si tuviéramos que resumir los hechos con valores para el *precio* $\{1'5, 2'0, 3'5, 1'5, 2'0\}$ utilizando la *Media*, obtendríamos un nuevo hecho que resumiría estos valores a un único hecho con valor $2'1$.

Como puede verse, aplicar uno de estos operadores implica un pérdida de información. Esto se debe a que obtenemos un valor que resume la información del conjunto de hechos al nivel de detalle que queremos. Si luego quisiéramos recuperar el nivel de detalle de partida no siempre es posible (p.e. los posibles conjuntos de *n* valores que tienen el mismo máximo es infinito). En la mayoría de los modelos multidimensionales propuestos este problema no se trata. Nosotros queremos dotar a nuestro modelo de mecanismos que impidan esta pérdida de información, de tal forma que desde cualquier nivel de detalle se pueda recuperar la información subyacente. Lo que proponemos es que un DataCubo vaya almacenando la información sobre los niveles de detalle sobre los que se vaya accediendo para impedir la pérdida de información. Para ello, definimos la estructura *historia*.

Definición 3.7 *Un objeto de tipo* **historia** *será la estructura recursiva siguiente:*

$$\begin{aligned} H^0 &= \Omega \\ H^{n+1} &= (A, l_b, F, G, H^n) \end{aligned} \tag{3.5}$$

donde:

- Ω *es la clausura de la recursividad,*
- F *es un conjunto de hechos,*
- l_b *es un conjunto de niveles* $(l_{1b}, ..., l_{nb})$*,*
- A *es una aplicación de* l_b *en* F *(*$A : l_b \rightarrow F$*),*
- G *es un operador de agregación.*

Esta estructura lo que hará será almacenar lo diferentes niveles de detalle por los que ha pasado un DataCubo (y cómo se ha obtenido) para poder recuperarla si posteriormente se necesita obtener un mayor nivel de detalle. El manejo de la estructura *historia* se verá al definir las operaciones.

3.2.1.3. DataCubo Preciso

Una vez que tenemos definidos las estructuras para las dimensiones y los hechos, podemos establecer cuál será la estructura de los DataCubos.

Un DataCubo estará formado por las estructuras que antes hemos definido. Así pues, tendrá una componente que representará a los hechos que analizar. Otro de los elementos que utilizaremos al definir un DataCubo será las dimensiones que caracterizan a estos hechos. En este caso será un conjunto ordenado de las dimensiones. Además, tendremos que tener en cuenta a qué nivel de cada dimensión tenemos definidos los hechos y una aplicación que nos diga para cada coordenada que definen los valores de estos niveles el hecho relacionado. Para impedir la

pérdida de información al cambiar el nivel de detalle, el DataCubo irá almacenando los estados por los que pasa utilizando una estructura de tipo historia. Por esto, un DataCubo se define como sigue.

Definición 3.8 *Un* **DataCubo** *es una tupla* $C = (D, l_b, F, H, A)$ *tal que* $D = (d_1, ..., d_n)$ *es un conjunto de dimensiones,* $l_b = (l_{1b}, ..., l_{nb})$ *conjunto de niveles tal que* l_{ib} *pertenece a* d_i*,* $F = R \cup \emptyset$ *donde* R *es el conjunto de hechos y* $\emptyset$ *un símbolo especial,* H *es un objeto de tipo historia, y* A *es una aplicación definida como* $A : l_{1b} \times ... \times l_{nb} \rightarrow F$*, que para cada conjunto de valores de las dimensiones devuelve el hecho relacionado con estas coordenadas.*

Si para un $\vec{a} = (a_1, ..., a_n)$ se tiene $\emptyset$, se indica que para esta combinación de valores no existe un hecho definido.

Para comenzar los análisis, definiremos un DataCubo al mayor nivel de detalle y sobre él iremos operando. A este DataCubo de partida lo llamaremos *básico*.

Definición 3.9 *Diremos que un DataCubo es básico si* $l_b = (l_{1\perp}, ..., l_{n\perp})$ *y* $H = \Omega$*.*

Sobre este DataCubo básico aplicaremos las diferentes operaciones de los modelos multidimensionales para ir refinando los análisis.

3.2.1.4. Ejemplo

Para aclarar los conceptos presentados veremos un ejemplo de esquema multidimensional y cómo se traduciría a la estructura que hemos presentado. En la figura 3.2 viene recogido un ejemplo de esquema multidimensional. Este intenta modelar el análisis de las causas de las quejas realizadas por los clientes. La estructura del DataCubo que representa este esquema sería:

$$C_{Quejas} = (\{Cliente, Producto, Tiempo, Causa\}, \{Cantidad\} \cup \emptyset, \Omega, A) \tag{3.6}$$

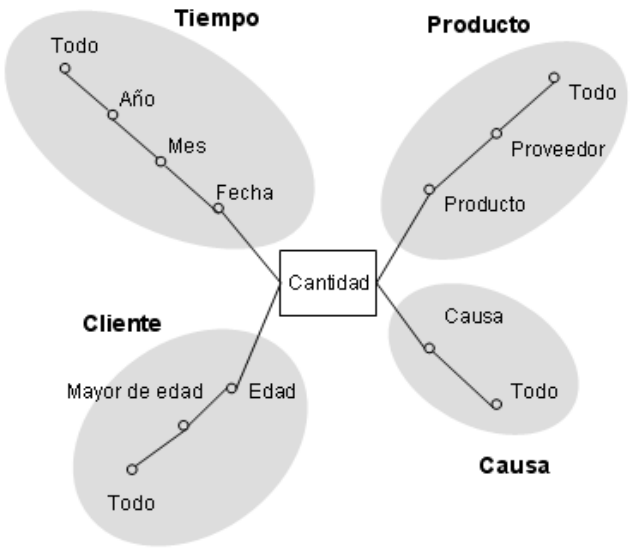


Figura 3.2: Ejemplo de esquema multidimensional

donde las dimensiones están definidas como:

$$\begin{gathered} Cliente = (\{Edad, Mayor\ Edad, Todos\}, \leq_{Cliente}, Edad, Todos) \\ Producto = (\{Producto, Proveedor, Todos\}, \leq_{Producto}, Producto, Todos) \\ Tiempo = (\{Fecha, Mes, Año, Todos\}, \leq_{Tiempo}, Fecha, Todos) \\ Causa = (\{Causa, Todos\}, \leq_{Causa}, Todos) \end{gathered} \tag{3.7}$$

El único elemento que nos quedaría por definir sería la relación A entre dimensiones y hechos. En este caso, la relación tendría la forma

$$A : Edad \times Producto \times Fecha \times Causa \rightarrow \{Cantidad\} \cup \emptyset \tag{3.8}$$

3.2.2. Operaciones

Una vez que tenemos nuestra estructura multidimensional, necesitamos dotarla de las operaciones básicas. En este apartado realizaremos la definición de las operaciones para cambiar el nivel de las jerarquías (roll-up y drill-down) junto a la selección, proyección (dice y slice, respectivamente) y pivotaje. En la sección 3.7 veremos un ejemplo de la combinación de diferentes operaciones para dar respuesta a una consulta.

3.2.2.1. Roll-up

A la hora de aplicar la operación de roll-up, necesitamos agregar los hechos de niveles inferiores al establecido. Para cada valor del nuevo nivel de definición de hechos en el nuevo DataCubo, necesitamos obtener el conjunto de hechos con los que están relacionamos para obtener el valor del nuevo hecho, que resume a estos. Para obtenerlos lo haremos de forma recursiva: si el nivel considerado es el básico, para cada valor tendremos que obtener el conjunto de hechos que están relacionado con éste; si por el contrario es un nivel más alto en la jerarquía el

conjunto de hechos relacionados sería la unión de los conjuntos de hechos relacionados con los elementos de niveles pertenecientes al conjunto de hijos del nivel que están agrupados por el valor. En el caso preciso este conjunto se define como sigue.

Definición 3.10 *Sea* $\overrightarrow{c} = (c_b^1, ..., c_{ij}, ..., c_b^n)$. *Para cada valor* c_{ij} *perteneciente a* l_r *tendremos el conjunto*

$$F_{c_{ij}} = \begin{cases} \bigcup\limits_{l_k \in H_{l_r}} F_{c_{kp}} / c_{kp} \in l_k \wedge c_{kp} \subseteq c_{ij} & \textit{si } l_r \neq l_b \\ \{h/h \in F \wedge \exists \overrightarrow{c}\, A(\overrightarrow{c}) = h\} & \textit{si } l_r = l_b \end{cases} \tag{3.9}$$

Una vez tenemos la definición de $F_{c_{ij}}$, podemos introducir la operación roll-up.

Definición 3.11 *El resultado de aplicar la operación* **roll-up** *sobre la dimensión* d_i, *el nivel* l_r $(l_r \neq l_\perp)$, *utilizando el operador* G *sobre un DataCubo* $C = (D, l_b, F, A, H)$ *es otro DataCubo* $C' = (D, l_b', F', A', H')$ *tal que:*

- $l_b' = (l_{1b}, ..., l_r, ..., l_{nb})$,
- $A'(c_b^1, ..., c_j^r, ..., c_b^n) = G(\{h/h \in F_{c_j^r}\})$,
- F' *es la imagen de* A',
- $H' = (A, l_b, F, G, H)$.

Como puede verse, lo que se hace es modificar la granularidad, haciendo que los hechos estén definidos al nivel l_r en lugar de l_b, con lo que disminuimos el nivel de detalle.

De la definición de la operación de roll-up que hemos hecho, podemos empezar a ver el funcionamiento de la estructura *historia* dentro del modelo. Al

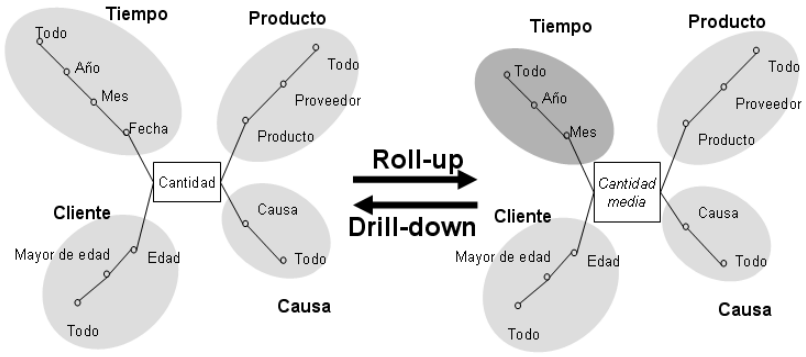


Figura 3.3: Ejemplo de la aplicación de las operaciones roll-up y drill-down

aplicar la operación, estamos disminuyendo el nivel de detalle. Agregamos la información utilizando un operador G que resume la información hasta la granularidad deseada. La estructura *historia* lo que consigue es que se almacene los diferentes niveles de granularidad por los que ha ido pasando el DataCubo, para volver a ellos si fuera necesario.

Si aplicamos una operación de roll-up sobre el esquema presentado en la sección3.2.1.4 utilizando la dimensión *Tiempo* y nivel *Mes* con operador de agregación *Media* lo que obtendríamos sería un nuevo DataCubo en el que los hechos estaría definidos a nivel de meses en lugar de fechas concretas, con lo que la granularidad habría aumentado, disminuyendo el nivel de detalle (Figura 3.3).

3.2.2.2. Drill-down

La operación anterior implica disminuir el nivel de detalle, aumentando el tamaño del grano al que están definidos los hechos. La operación drill-down es justo lo contrario: aumentar el nivel de detalle de los hechos, definiéndolos a niveles más bajos de la jerarquía.

Para poder llevar a cabo ese aumento del detalle, necesitamos que la información esté disponible en el DataCubo de forma cuando menos implícita. Este papel es el que juega la estructura *historia* que hemos definido. De esta forma, realizar un drill-down no será más que recuperar la información a un nivel mayor de detalle en el que se encontraba antes de realizar un roll-up.

Definición 3.12 *El resultado de aplicar* **drill-down** *sobre el DataCubo* $C = (D, l_b, F, A, H)$ *con* $H = (A', l'_b, F', H')$ *es otro DataCubo* $C' = (D, l'_b, F', A', H')$.

Como puede verse de la definición, lo que se hace es recuperar la información almacenada en la estructura de tipo historia. Cada aplicación de esta operación implica deshacer un roll-up anterior, de esta forma podemos llegar hasta el DataCubo básico de partida sin pérdida de información.

Si aplicáramos una operación tipo drill-down sobre el DataCubo resultado del ejemplo de la sección anterior (Figura 3.3) lo que obtendríamos sería el DataCubo de partida (el DataCubo básico que comenzamos definiendo).

3.2.2.3. Dice

En muchos análisis es posible que necesitemos centrarnos en una parte del DataCubo. Esta área quedará delimitada por las coordenadas de las dimensiones que definen los hechos. La operación *dice* se encarga de hacer justo esto: seleccionar un área del DataCubo basada en condiciones sobre los valores de las dimensiones. Estas condiciones afectan a los valores sobre los que los aplicamos y los que están relacionados que pertenecen a otros niveles.

Definición 3.13 *El resultado de aplicar* **dice** *con una condición* β *sobre el nivel* l_r *de la dimensión* d_i *del DataCubo* $C = (D, l_b, F, A, H)$ *es otro DataCubo* $C' = (D', l'_b, F', A', \Omega)$ *donde:*

- $D' = \{d_1, ..., d'_i, ..., d_n\}$ *donde* $d'_i = (l'_i, \leq_{d_i}, l_b, l_\top)$ *con* $l' = \{l_j / l_b \leq_{d_i} d_{l_j}\}$ *y*

$$d'_i.l'_j = \begin{cases} \{c_{jk} / c_{jk} \in l_j \wedge \beta(c_{jk})\} & \textit{si } l'_j = l_r \\ \{c_{jk} / c_{jk} \in d_i.l_j \wedge \beta(c_{jk}) \wedge c_{jk} \subseteq c_r\} & \textit{si } l'_j \leq_d l_r \\ \{c_{jk} / c_{jk} \in d_i.l_j \wedge \beta(c_{jk}) \wedge c_r \subseteq c_{jr})\} & \textit{si } l_r \leq_d l'_j \end{cases}$$

- $A'(c_b^1, ..., c_b^i, ..., c_b^n) = (h, \alpha) / c_b^1 \in d'_1.l'_b \wedge ... \wedge c_b^n \in d'_n.l'_b \wedge A(c_b^1, ..., c_b^n) = (h, \alpha)$,
- F' *sería la imagen de* A'.

Lo que conseguimos al aplicar la operación es restringir los valores de la dimensión a aquellos que cumplan la condición y o que estén relacionados con alguno que la cumpla (en niveles superiores o inferiores).

En la aplicación de la operación vemos como en el nuevo DataCubo establecemos la estructura historia a Ω (eliminamos los estados anteriores). Esto es así pues al aplicar esta operación lo que estamos obteniendo es un DataCubo que representa un nuevo esquema multidimensional y un cambio del anterior. Por esto, este nuevo DataCubo se considerará *básico* para este nuevo esquema.

En el caso del DataCubo ejemplo, es posible que quisiéramos centrarnos en las quejas realizadas en un único mes. De esta forma lo que hacemos es quedarnos con una parte del espacio multidimensional de partida. Para hacerlo tendríamos que realizar una operación *dice* en la dimensión *Tiempo* y nivel *Mes* con la condición $\beta(x) =$*"x es Enero"*, para el caso de querer estudiar las quejas realizadas en este mes.

3.2.2.4. Slice

Algunos análisis pueden depender de no todas las dimensiones que tengamos definidas en un DataCubo. En estos casos debemos reducir la dimensionalidad,

modificando los hechos para reflejar ésta reducción. Como en el caso de la operación de roll-up, lo que sucede es una disminución del detalle al que están definidos los hechos. Por ello, utilizaremos un operador de agregación para realizar esta modificación. En este caso, cada hecho nuevo será el resultado de agregar los que compartan las coordenadas de las dimensiones no implicadas en la operación. Además, estas coordenadas serán las de estos nuevos hechos.

Como puede verse, esta operación modifica la estructura del DataCubo. Por esto, y como en el caso de *dice*, el resultado es un esquema multidimensional distinto, aunque obtenido a partir de uno existente. Por esto, la estructura de tipo *historia* del DataCubo resultado volverá a ser Ω.

Definición 3.14 *El resultado de aplicar* **slice** *sobre la dimensión d_i con el operador de agregación G en el DataCubo $C = (D, l_b, F, A, H)$ sería un DataCubo $C' = (D', l'_b, F', A', \Omega)$ tal que:*

- $D' = (d_1, ..., d_{i-1}, d_{i+1}, ..., d_n)$,
- $l'_b = (l_{ib}, ..., l_{i-1b}, l_{i+1b}, ..., l_{nb})$,
- $A'(c_b^1, ..., c_b^{i-1}, c_b^{i+1}, ..., c_b^n) = G(\{h / \exists c_b^i A(c_b^1, ..., c_b^{i-1}, c_b^i, c_b^{i+1}, ..., c_b^n) = h\})$,
- *F' es la imagen de A'.*

Si quisiéramos eliminar la componente temporal del análisis de las quejas, lo que deberíamos hacer es no considerar la dimensión *Tiempo*. Esto se traduce en una reducción de la dimensionalidad de esquema (de cuatro dimensiones pasaríamos a tres) por lo que necesitaríamos agregar los hechos para establecer la granularidad a esta nueva dimensionalidad. Una operación *Slice* aplicada sobre el DataCubo, y aplicada a la dimensión *Tiempo*, utilizando la función de agregación *Suma* lo que nos daría sería un nuevo DataCubo con tres dimensiones y como hechos el número de quejas para cada combinación de los valores de estas dimensiones (Figura 3.4).

Figura 3.4: Ejemplo de la aplicación de la operación Slice

3.2.2.5. Pivot

Otra de las operaciones que se pueden presentar es el cambio en la definición de las coordenadas de cada hecho, modificando el orden de las mismas. Esto se traduce en un cambio en el orden de las dimensiones dentro de la estructura del DataCubo. Esta operación implica la obtención de un nuevo esquema multidimensional en cuanto a su definición en el modelo.

Definición 3.15 *El resultado de* **pivotar** *las dimensiones d_i y d_j en un DataCubo $C = (D, l_b, F, A, H)$ es otro DataCubo $C' = (D', l'_b, F, A', \Omega)$ donde:*

- $D' = (d_1, ..., d_{i-1}, d_j, d_{i+1}, ..., d_{j-1}, d_i, d_{j+1}, ..., d_n)$,
- $l'_b = (l_{1b}, ..., l_{i-1b}, l_{jb}, l_{i+1b}, ..., l_{j-1b}, l_{ib}, l_{j+1b}, ..., l_{nb})$,
- $A(^1_b, ..., c^{i-1}_b, a^i_b, a^{i+1}_b, ..., c^{j-1}_b, c^j_b, c^{j+1}_b, ..., c^n_b) =$
  $A(c^1_b, ..., c^{i-1}_b, a^j_b, c^{i+1}_b, ..., c^{j-1}_b, a^i_b, a^{j+1}_b, ..., c^n_b)$.

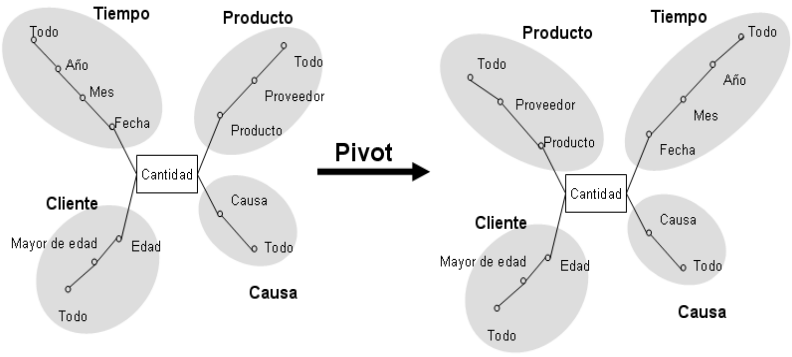


Figura 3.5: Ejemplo de la aplicación de la operación pivot

Esta operación es puramente estructural dado que no modifica los valores del DataCubo sino el orden de acceso a ellos. Sobre el DataCubo anterior, aplicarla sobre las dimensiones *Tiempo* y *Producto* (Figura 3.5) implicaría pasar de la estructura

$$C_{Quejas} = (\{Cliente, Producto, Tiempo, Causa\}, \{Cantidad\} \cup \emptyset, \Omega, A) \tag{3.10}$$

a la siguiente

$$C'_{Quejas} = (\{Cliente, Tiempo, Producto, Causa\}, \{Cantidad\} \cup \emptyset, \Omega, A) \tag{3.11}$$

3.2.3. DataCubo Válido

Una vez definidas las operaciones sobre nuestra estructura nos queda definir qué esquemas de los posibles que podemos obtener con nuestra estructura representan un esquema multidimensional válido.

Definición 3.16 *Un DataCubo es* **válido** *si es un* DataCubo básico *o se ha obtenido aplicando un número finito de operaciones sobre un* DataCubo básico.

3.3. Modelo Multidimensional Difuso

Una vez que hemos presentado un modelo multidimensional preciso, lo utilizaremos para extender los conceptos y poder representar la imprecisión e incertidumbre. Para ello utilizaremos la Teoría de los Conjuntos Difusos ([Zad65]). En esta caso iremos presentando los mismos conceptos que el caso preciso, explicando cuáles son las diferencias al considerar una estructura difusa y cómo se adaptan las definiciones a esta nueva casuística.

3.3.1. Estructura Multidimensional Difusa

Como en el caso preciso, comenzaremos con la estructura de las dimensiones, introduciendo las relaciones jerárquicas difusas, para presentar posteriormente los hechos y la estructura que tendrá un DataCubo en el caso difuso.

3.3.2. Dimensión Difusa

La estructura de las dimensiones en el caso difuso será similar al preciso. La definición 3.1 sería también válida para una dimensión difusa. La diferencia radicará en que los nombres o etiquetas de los niveles no representarán conjuntos en el sentido clásico, sino desde el punto de la *Teoría de Conjuntos Difusos* ([Zad65]). De esta manera podremos modelar conceptos difusos y relacionarlos en una jerarquía. El resultado final será un jerarquía difusa sobre los elementos de la dimensión.

En este caso, un elemento puede estar incluido en el conjunto que define otro, pero tendremos que tener en cuenta que habrá asociado un grado de pertenencia.

Cada elemento de un nivel lo que definirá es un conjunto difuso. Para establecer este grado de pertenencia, utilizaremos la *relación de parentesco*.

Definición 3.17 *Para cada par de niveles l_i y l_j tal que tendremos la relación*

$$\mu_{ij} : l_i \times l_j \rightarrow [0, 1] \tag{3.12}$$

A esta relación la denominaremos **relación de parentesco**.

Mediante esta relación podremos definir el grado de inclusión de los elementos de un nivel en sus niveles padre. Si esta función sólo toma como posibles valores 0 ó 1 y cada elemento de un nivel sólo toma este segundo para un valor de un nivel padre, nos encontramos en el caso de una relación de jerarquía precisa entre los dos niveles (el elemento o pertenece completamente o no pertenece). Este sería el caso de la relación entre los valores de los niveles *Mayor Edad* y *Edad* de la jerarquía de la figura 3.1. Para estos, la relación de parentesco que los une sería

$$\mu_{Mayor\ Edad,Edad}(Si, x) = \begin{cases} 1 & \text{si } x \in [18, 100] \\ 0 & \text{en otro caso} \end{cases}$$

$$\mu_{Mayor\ Edad,Edad}(No, x) = \begin{cases} 1 & \text{si } x \in [1, 17] \\ 0 & \text{en otro caso} \end{cases}$$

que equivaldría a la relación definida en el caso preciso.

Este concepto de jerarquía se puede suavizar si permitimos que la función entre los niveles tome valores en todo el intervalo [0,1] y que se puedan establecer más de un valor mayor que cero para un valor con diferentes valores de un nivel padre (pasar de relaciones uno-a-mucho a muchos-a-muchos). En este caso, estaríamos hablando de una relación de jerarquía difusa entre los niveles. Mediante esta relación podemos manejar la imprecisión entre los elementos en las jerarquías.

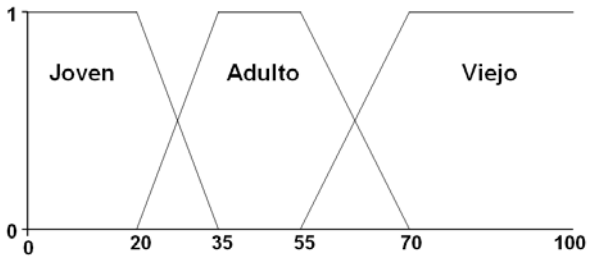


Figura 3.6: Definición de la relación de parentesco entre los niveles Grupo y Edad

El valor entre dos elementos en esta relación establece el grado de certeza de la relación de jerarquía, permitiendo también que un valor sea agrupado por diferentes valores en niveles superiores. Sobre nuestro ejemplo de dimensión, la relación entre *Grupo* y *Edad* lo que pretende es agrupar las posibles edades de las personas atendiendo a si son jóvenes, adultas o viejas. Este tipo de agrupamiento se modela de forma más natural si utilizamos este tipo de relación de jerarquía difusa. En la figura 3.6 viene recogida cómo sería la forma de la función $\mu_{Grupo,Edad}$.

Cuando los niveles son no consecutivos en la jerarquía, necesitamos extender esta relación. Esta nos dará el grado de pertenencia de un elemento al conjunto difuso que define un elemento en un nivel superior. Lo que haremos será considerar las relaciones de parentesco existentes entre los niveles intermedios. Aplicando estas ideas, definimos la *relación de parentesco extendida*.

Definición 3.18 *Para cada par de niveles l_i y l_j pertenecientes a d tal que $l_j \leq_d l_i \wedge l_j \neq l_i$ tendremos la relación $\eta_{ij} : l_i \times l_j \rightarrow [0,1]$ definida como*

$$\eta_{ij}(a,b) = \begin{cases} \mu_{ij}(a,b) & \textit{si } l_j \in H_{l_i} \\ \bigoplus_{l_k \in H_{l_i}} \bigotimes_{c \in l_k} (\mu_{ik}(a,c) \otimes \eta_{kj}(c,b)) & \textit{en otro caso} \end{cases} \tag{3.13}$$

donde $\bigoplus$ y $\bigotimes$ son una t-conorma y una t-norma respectivamente o de la familia de los operadores MOM y MAM, definidos por Yager ([Yag94]), que incluyen a las t-conormas y t-normas, respectivamente. Esta relación la denominaremos **relación de parentesco extendida**.

Esta relación lo que nos indica es el grado de conexión entre los valores de dos niveles no consecutivos en la jerarquía. Para hacerlo lo que se hace es considerar todos los posibles caminos entre los dos niveles agrupando los valores de cada uno mediante una función tipo y-lógico (familia de operadores MAM), y agrupar el valor obtenido en cada camino mediante una función de tipo o-lógico (familia de operadores MOM).

Para aclarar el procedimiento, veamos como se haría en el esquema de dimensión que hemos utilizado como ejemplo, calculando $\eta_{Todo,Edad}(Todo, 25)$. En este caso, entre los dos niveles existen dos posibles caminos. Veamos cada uno de ellos:

- *Todo – Mayor Edad – Edad*. Para este camino, tenemos dos posibilidades dependiendo de los valores del nivel intermedio que consideramos. En la figura 3.7.a se recogen ambos. Si consideramos los valores de ambas opciones, tenemos para este camino $(1 \otimes 1) \oplus (1 \otimes 0)$.

- *Todo – Grupo – Edad*. Como en el caso anterior, también tenemos varias opciones que vienen recogidas en la figura 3.7.b. Agrupando los valores tenemos $(1 \otimes 0{,}7) \oplus (1 \otimes 0{,}3) \oplus (1 \otimes 0)$.

Una vez calculados los dos caminos, agruparíamos los dos utilizando una t-conorma. Si utilizamos el máximo como t-conorma y el mínimo como t-norma, tendríamos

$$((1 \otimes 1) \oplus (1 \otimes 0)) \oplus ((1 \otimes 0,7) \oplus (1 \otimes 0,3) \oplus (1 \otimes 0)) =$$
$$(1 \oplus 0) \oplus (0,7 \oplus 0,3 \oplus 0) = 1 \oplus 0,7 = 1,$$

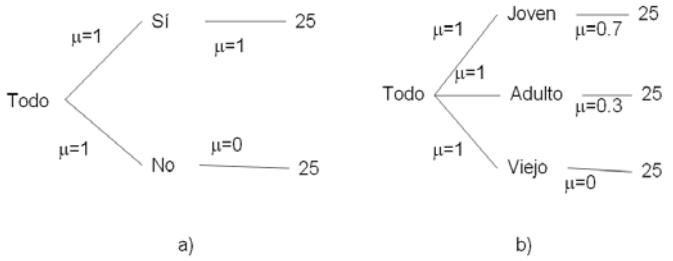


Figura 3.7: Ejemplo de cálculo de la relación de parentesco extendido. a) camino Todas – Mayor Edad – Edad b) camino Todas – Grupo – Edad

con lo que el valor para $\eta_{Todo,Edad}(Todo, 25)$ sería 1, interpretando con este resultado que la edad 25 está agrupada bajo la etiqueta *"Todo"* del nivel *"Todo"* con grado 1.

3.3.2.1. Hechos Difusos

Las variables del dominio que queremos analizar definirán los hechos del DataCubo. En el caso difuso, junto a las variables a analizar incluimos un valor que nos da el grado de certeza de dicho hecho. Mediante este grado podremos incluir datos con incertidumbre en el análisis y controlar su influencia en el mismo.

Si se desea utilizar diferentes fuentes de información, es posible que no todas sean igual de confiables en cuanto a la veracidad de sus datos (p.e. sean externas a la organización). En los modelos precisos no es posible distinguir la información según su fuente. En este caso, tendremos dos posibilidades:

- Considerar los datos. De esta forma estaríamos corriendo el peligro de tomar decisiones estratégicas basadas en datos que pueden no ser correc-

tos.

- No considerarlos. Ahora el problema es justo el contrario dado que nos arriesgaríamos a decisiones basadas en un conjunto sesgado de datos.

Este problema podríamos abordarlo estableciendo la importancia de la información según su fuente y así controlar la influencia en el análisis. Esto podemos hacerlo en el modelo difuso mediante el grado de certeza que hemos comentado, de forma que los datos procedentes de las fuentes menos confiables tuvieran un grado bajo para disminuir el posible error de utilizarlo, pero aun así considerándolos para el análisis.

Así pues, los hechos en el caso difuso quedarían definidos como sigue.

Definición 3.19 *Considerando un conjunto de atributos $A_1, ..., A_n$ con dominios $D_1, ..., D_n$, denominaremos* **hecho** *a cualquier par (h, α), donde $h = (x_1, ..., x_n)$ con $x_i \in D_i \; \forall i = 1, ..., n$, es decir, cualquier n-tupla definida sobre los dominios de los atributos que interesa estudiar, y $\alpha \in [0, 1]$.*

El valor α sería el que controlaría la influencia del hecho en el posterior análisis. Cuanto más cercano a 1 sea este valor, mayor será la influencia. Este enfoque es el mismo al utilizado en el modelo propuesto por Laurent ([Lau03b]).

En el caso del DataCubo de ventas con las medidas *cantidad* vendida y *precio*, usado como ejemplo en los hechos precisos, en este caso tendríamos que añadirle a cada hecho un valor α en el intervalo [0, 1]. Así pues, el mismo hecho de ejemplo en el caso preciso $(1, 3'5)$ en el caso difuso se establecería como $((1, 3'5), 1)$, suponiendo que tenga un valor de certeza 1.

En consecuencia, los operadores de agregación tienen que tener en cuenta este grado de certeza que consideramos en cada hecho. Además, el resultado de aplicarlos debe ser a su vez un hecho, dado que el conjunto de operaciones ha

de ser cerrado. Como en el caso preciso, el operador debe trabajar sobre bolsas de datos, aunque en este caso difusas (Sección 2.1.1). En la literatura se han propuesto múltiples métodos de agregación de valores difusos. Para no limitar la potencia del modelo, no nos centramos en ninguno en particular. Por ello, establecemos condiciones genéricas para la definición de los operadores de agregación. Así pues, un operador de agregación para nuestro modelo difuso respondería a la siguiente definición.

Definición 3.20 *Siendo $\widetilde{\mathcal{B}}(X)$ el conjunto de todas las posibles bolsas difusas definidas utilizando los elementos de X, $\widetilde{\mathcal{P}}(X)$ las partes difusas de X, y D_x un dominio natural o numérico, definimos un operador de agregación G como una función $G : \widetilde{\mathcal{B}}(D_x) \rightarrow \widetilde{\mathcal{P}}(D_x) \times [0, 1]$.*

Un ejemplo de operadores de agregación sobre valores difusos son los propuestos por Rundensteiner y Bic ([RB89]) para un modelo relacional difuso y que hemos comentado en la Sección 2.1.6.4 . Para utilizarlos en nuestro modelo multidimensional los adaptaremos a la estructura de los hechos.

Definición 3.21 *Si R es un operador de agregación definido por Rundensteiner y Bic, y F la bolsa difusa que agregar, definimos el operador G_R en nuestro modelo como $G_R(F) = (R(F'), 1)$ donde $F' = \{\alpha/h$ tal que $(h, \alpha) \in F\}$.*

Para clarificar esta definición veamos un ejemplo simple. Supongamos que nuestra bolsa de hechos a agregar es

$$F = \{(\{1/2, 0'7/8, 0'2/17\}, 1), (\{1/3, 0'7/4, 0'2/6, 0'1/18\}, 1)\} \qquad (3.14)$$

y queremos calcular la Media de los mismos. El parámetro sobre el que operaría el operador de Rundensteiner sería

$$F' = \{1/\{1/2, 0'7/8, 0'2/17\}, 1/\{1/3, 0'7/4, 0'2/6, 0'1/18\}\} \qquad (3.15)$$

y el resultado obtenido:

$$G_{Media}(F) = \\ (Media(\{1/\{1/2, 0'7/8, 0'2/17\}, 1/\{1/3, 0'7/4, 0'2/6, 0'1/18\}\}), 1) = \\ (\{1/3, 0'7/4, 0'2/6, 0'1/18, 0'7/8, 0'2/17\}, 1) \tag{3.16}$$

3.3.2.2. DataCubo Difuso

En el caso difuso, un DataCubo estará compuesto por los mismos elementos que en el caso preciso. Lo que variará será el tipo de éstos. Así pues, en lugar de tener un conjunto de dimensiones precisas, trabajará sobre dimensiones difusas. El mismo caso será el de los hechos, dado que ahora incorporarán un valor de certeza. La estructura de tipo historia utilizada para almacenar los cambios de nivel, también modificará el tipo de los datos que almacena para adaptarse al caso difuso. Por eso, la definición será similar a la 3.8.

Definición 3.22 *Un* **DataCubo difuso** *es una tupla $C = (D, l_b, F, H, A)$ tal que $D = (d_1, ..., d_n)$ es un conjunto de dimensiones difusas, $l_b = (l_{1b}, ..., l_{nb})$ conjunto de niveles tal que l_{ib} pertenece a d_i, $F = R \cup \emptyset$ donde R es el conjunto de hechos difusos y $\emptyset$ un símbolo especial, H es un objeto de tipo historia, y A es una aplicación definida como $A : l_{1b} \times ... \times l_{nb} \rightarrow F$, que para cada conjunto de valores de las dimensiones devuelve el hecho relacionado con estas coordenadas.*

La consideración de cuándo un DataCubo es básico será la misma que en el caso crisp (definición 3.9).

3.3.3. Operaciones

Una vez que hemos extendido la estructura para la consideración de jerarquías y hechos difusos, tenemos que estudiar cómo se ven modificadas las operaciones

para tratar estas estructuras. Como en el caso preciso, en la sección 3.7 veremos ejemplos de combinación de operaciones para dar respuesta a consultas de ejemplo.

3.3.3.1. Roll-up

En el modelo difuso hemos visto cómo se ve modificada la relación entre los elementos en una dimensión. Ahora un valor en un nivel podrá estar relacionado con varios de un mismo nivel superior, teniendo en cada caso un valor de pertenencia dado. Esto hace que al aplicar la operación de roll-up, el conjunto de hechos a agregar no sea disjunto para dos valores de un mismo nivel. Por esto, lo primero que tenemos que modificar es la obtención de este conjunto de hechos relacionados con cada valor en las niveles de una dimensión. En el caso difuso, consideraremos que un hecho está relacionado con un valor de un nivel si existe un camino en la jerarquía hasta el valor con todas las *relaciones de parentesco* estrictamente mayor que 0.

Definición 3.23 *Sea* $\overrightarrow{c} = (c_b^1, ..., c_{ij}, ..., c_b^n)$. *Para cada valor* c_{ij} *perteneciente a* l_r *tendremos el conjunto*

$$F_{c_{ij}} = \begin{cases} \bigcup\limits_{l_k \in H_{l_r}} F_{c_{kp}} / c_{kp} \in l_k \wedge \mu_{ik}(c_{ij}, c_{kp}) > 0 & \textit{si } l_r \neq l_b \\ \{h / h \in F \wedge \exists \overrightarrow{c}\, A(\overrightarrow{c}) = h\} & \textit{si } l_r = l_b \end{cases} \tag{3.17}$$

El conjunto $F_{c_{ij}}$ representa a todos los hechos que tienen alguna relación, por pequeña que sea, con este valor c_{ij}. Con esta definición ya podemos introducir la operación roll-up en nuestro modelo multidimensional difuso.

Definición 3.24 *El resultado de aplica la operación* **roll-up** *sobre la dimensión* d_i, *el nivel* l_r $(l_r \neq l_\perp)$, *utilizando el operar* G *sobre un DataCubo* $C = (D, l_b, F, A, H)$ *es otro DataCubo* $C' = (D, l_b', F', A', H')$ *tal que:*

- $l'_b = (l_{1b}, ..., l_r, ..., l_{nb})$,
- $A'(c_b^1, ..., c_j^r, ..., c_b^n) = G(\{(b, \alpha \otimes \eta_{rb}(c_j^r, c_b^r))/(b, \alpha) \in F_{c_j^r} \wedge A(c_b^1, ...c_b^r, ..., c_b^n) = (b, \alpha)\})$,
- F' *es la imagen de* A',
- $H' = (A, l_b, F, G, H)$.

Como puede verse, la definición es muy similar al caso crisp. Las principales diferencias radican en el conjunto de hechos que agregar en cada caso y en el propio operador de agregación que utilicemos. Los nuevos hechos serán conjuntos difusos. El papel de la historia se mantiene, almacenando el estado anterior al roll-up para poder recuperarlo si fuera necesario.

El agregar una gran cantidad de hechos utilizando los operadores difusos propuestos en la literatura es muy costoso. Por eso puede ser interesante no considerar hechos con una certeza muy baja cuya influencia en el resultado final es casi inapreciable. Si a la hora de aplicar la operación sólo queremos considerar los valores cuya relación sea mayor que un grado prestablecido, utilizaríamos la operación $roll - up^{\alpha}$, definida a continuación.

Para poder limitar el grado de relación de los elementos definimos el siguiente conjunto.

Definición 3.25 *Sea* $\overrightarrow{c} = (c_b^1, ..., c_{ij}, ..., c_b^n)$. *Para cada valor* c_{ij} *perteneciente a* l_r *y* $\alpha \in [0, 1]$ *tendremos el conjunto*

$$F_{c_{ij}}^{\alpha} = \begin{cases} \bigcup\limits_{l_k \in H_{l_r}} F_{c_{kp}}^{\alpha}/c_{kp} \in l_k \wedge \mu_{ik}(c_{ij}, c_{kp}) \geq \alpha & \text{si } l_r \neq l_b \\ \{h/h \in F \wedge \exists a_1, ..., a_n A(a_1, ..., a_n) = h\} & \text{si } l_r = l_b \end{cases} \tag{3.18}$$

que referencia todos los elementos que tienen relación con el elemento c_{ij} *con grado como mínimo* α.

Una vez tenemos tenemos el conjunto restringido de hechos, su utilización en el roll-up es directa.

Definición 3.26 *Sea* $\alpha \in [0,1]$*, el resultado de aplicar la operación* **roll-up**$^{\alpha}$ *sobre la dimensión* d_i*, el nivel* l_r $(l_r \neq l_\perp)$*, utilizando el operar* G *sobre un DataCubo* $C = (D, l_b, F, A, H)$ *es otro DataCubo* $C' = (D, l'_b, F', A', H')$ *tal que:*

- $l'_b = (l_{1b}, ..., l_r, ..., l_{nb})$,
- $A'(c_b^1, ..., c_j^r, ..., c_b^n) = G(\{(b, \alpha \otimes \eta_{rb}(c_j^r, c_b^r))/(b,\alpha) \in F_{c_j^r}^{\alpha} \wedge A(c_b^1, ...c_b^r, ..., c_b^n) = (b,\alpha)\})$,
- F' *es la imagen de* A',
- $H' = (A, l_b, F, G, H)$.

Si utilizamos $roll - up^1$ estaríamos en el caso preciso, dado que sólo consideramos dos posibles relaciones entre valores en las dimensiones: o está completamente relacionado o no lo está.

3.3.3.2. Drill-down

Esta operación no se ve modificada en el caso difuso, dado que sólo recupera la información almacenada en la estructura *historia* para deshacer la disminución de detalle realizada con una operación de roll-up. Por esto, la definición realizada para el caso preciso seguirá siendo válida, siempre teniendo en cuenta las diferencias en las estructuras implicadas.

3.3.3.3. Dice

Al realizar *dice* restringimos los valores de una dimensión a aquellos que cumplan una condición y a aquellos que estén relacionados con algún valor que sí la cumple. En el caso difuso, estos valores de otros niveles se han de obtener basándose en la relación de parentesco extendida. La definición de la operación sería la siguiente.

Definición 3.27 *El resultado de aplicar* **dice** *con una condición β sobre el nivel l_r de la dimensión d_i del DataCubo $C = (D, lb, F, A, H)$ es otro DataCubo $C' = (D', l'_b, F', A', \Omega)$ donde:*

- $D' = \{d_1, ..., d'_i, ..., d_n\}$ *donde* $d'_i = (l'_i, \leq_{d_i}, l_b, l_\top)$ *con* $l' = \{l_j / l_b \leq_{d_i} d_{l_j}\}$ *y*

$$d'_i.l'_j = \begin{cases} \{c_{jk}/c_{jk} \in l_j \wedge \beta(c_{jk})\} & \text{si } l'_j = l_r \\ \{c_{jk}/c_{jk} \in d_i.l_j \wedge \delta_{rj}(c_{jk})\} & \text{si } l'_j \leq_d l_r \\ \{c_{jk}/c_{jk} \in d_i.l_j \wedge \delta_{jr}(c_{jk})\} & \text{si } l_r \leq_d l'_j \end{cases}$$

  donde $\delta_{ij}(c) = \exists c_r \in l_r \beta(c_r) \wedge \eta_{ij}(c_r, c) > 0$,

- $A'(c^1_b, ..., c^i_b, ..., c^n_b) = (h, \alpha)/c^1_b \in d'_1.l'_b \wedge ... \wedge c^n_b \in d'_n.l'_b \wedge A(c^1_b, ..., c^n_b) = (h, \alpha)$,

- F' *sería la imagen de* A'.

De esta definición vemos que la diferencia con el caso preciso radica en la selección que hacemos de los valores. En el preciso eran aquellos que estaban incluidos en uno que sí lo cumple. En el caso difuso relajamos la condición, considerando son todos aquellos que tengan alguna relación con un valor que sí cumpla la condición.

Considerar todos los valores por baja que sea la relación puede no interesar en algunos casos. Para restringir la selección a los valores que estén relacionados con los que cumplen la condición con un grado cuando menos α, definimos la operación $dice^{\alpha}$.

Definición 3.28 *Sea $\alpha \in [0,1]$, el resultado de aplicar* **dice**$^{\alpha}$ *con una condición β sobre el nivel l_r de la dimensión d_i del DataCubo $C = (D, l_b, F, A, H)$ es otro DataCubo $C' = (D', l'b, F', A', \Omega)$ donde:*

- $D' = d_1, ..., d'_i, ..., d_n$ *donde* $d'_i = (l'_i, \leq_{d_i}, l_b, l_\top)$ *con* $l' = \{l_j / l_b \leq_{d_i} d_{l_j}\}$ *y*

$$d'_i.l'_j = \begin{cases} \{c_{jk} / c_{jk} \in l_j \wedge \beta(c_{jk})\} & \text{si } l'_j = l_r \\ \{c_{jk} / c_{jk} \in d_i.l_j \wedge \delta_{rj}(c_{jk})\} & \text{si } l'_j \leq_d l_r \\ \{c_{jk} / c_{jk} \in d_i.l_j \wedge \delta_{jr}(c_{jk})\} & \text{si } l_r \leq_d l'_j \end{cases}$$

  donde $\delta_{ij}(v) = \exists x \in l_r \beta(x) \wedge \eta_{ij}(x, v) \geq \alpha$,

- $A'(c_b^1, ..., c_b^i, ..., c_b^n) = (h, \alpha) / c_b^1 \in d'_1.l'_b \wedge ... \wedge c_b^n \in d'_n.l'_b \wedge A(c_b^1, ..., c_b^n) = (h, \alpha)$,

- F' *sería la imagen de* A'.

3.3.3.4. Slice

Al eliminar una dimensión del DataCubo, hemos de agregar valores para reducir la dimensionalidad. tanto en el caso preciso como el difuso el procedimiento es el mismo, dado que no se ven implicadas las jerarquías de las dimensiones. La diferencia radicará en que en el caso difuso tenemos que agregar hechos que son valores con un valor de certeza utilizando un operador de agregación que lo contemple. Por esto, la definición 3.14 seguirá siendo válida, siempre teniendo en cuenta que trabaja sobre estructuras distintas.

3.3.3.5. Pivot

Esta operación al ser una reordenación de las estructuras del DataCubo, no influye cual sea la estructura interna de estas. No necesita información de las jerarquías ni de los hechos para operador (es un cambio estructuras puro) por lo que la no hace falta adaptar la definición a la nueva estructura difusa.

Con esta operación hemos completado la extensión del modelo al caso difuso. Hemos definido la estructura multidimensional para contemplar la utilización de tanto hechos difusos como jerarquías difusas en la dimensiones. Para dotar de funcionalidad al modelo hemos formulado las principales operaciones (roll-up, drill-down, dice, slice y pivot) de los sistemas OLAP para dar respuesta a las consultas sobre un datacubo difuso. Para completar el modelo tendríamos que estudiar las propiedades de las operaciones. En la sección 3.5 se presentan las principales propiedades encontradas, dado que son compartidas por la extensión que se presenta a continuación.

Tras las propiedades presentaremos un ejemplo completo en el que se utilizan relaciones difusas y las lingüísticas que vamos a presentar a continuación.

3.4. Estructura Multidimensional con Jerarquías Lingüísticas

A la hora de abordar un análisis, es interesante contar con el conocimiento aportado por expertos para enriquecer los modelos. En este sentido, los expertos y los no expertos se sienten más cómodos trabajando con etiquetas lingüísticas en lugar que tener que aportar valores exactos. El modelo multidimensional propuesto permite modelar conceptos de forma relajada para representarlos de manera más intuitiva.

Un experto será complicado que establezca un valor concreto para representar

la relación entre dos elementos. En su lugar, le será más fácil calificarla con un grado del estilo *"la relación entre los elementos es alta"*.

El modelo difuso presentado no da esta facilidad. Por esto, lo que proponemos es extender el modelo difuso para permitir la definición de las relaciones entre los valores en la jerarquía utilizando etiquetas lingüísticas. Esto influirá tanto en la definición de la relación de parentesco como en la de parentesco extendido (modificando su cálculo).

En este sentido, tendremos que tener en cuenta que la propuesta que hagamos tiene que llegar a un compromiso entre facilidad de representación y uso intuitivo con una pérdida de eficiencia computacional lo menor posible para que el modelo siga siendo útil.

Las principales características que le exigiremos al modelo serán:

- El modelo debe de ser capaz de trabajar de forma conjunta con valores exactos y etiquetas en una dimensión dada. No en todas las relaciones jerárquicas tendremos la imprecisión de las etiquetas, por lo que no debe de forzarse a utilizarlas, introduciendo imprecisión donde se pueda evitar.

- Debe de poder trabajar sobre conjuntos distintos de etiquetas. Particularmente debe de ser capaz de manejar conjuntos de etiquetas con diferentes niveles de granularidad. Es posible que diferentes relaciones se definan con diferentes conjuntos de etiquetas debido a que las aporten personas distintas o que cada relación tenga necesidades distintas y no compatibles. Por esto, la solución que aportemos debe de poder trabajar independientemente de los conjuntos de etiquetas que se presenten en una dimensión dada.

- El manejo de la jerarquía debe de ser eficiente desde el punto de vista computacional para que no se pierda la operatividad de los análisis.

En el capítulo 2 hemos comentado algunas posibilidades propuestas para operar con etiquetas lingüísticas. Presentan diferentes problemas:

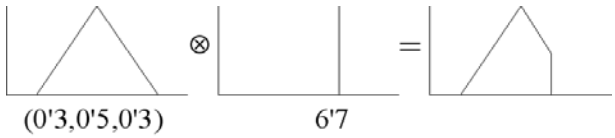


Figura 3.8: Ejemplo de aplicación del principio de extensión al *mínimo* como *t-norma*

- Transformar todas las etiquetas al nivel de granularidad menor y operar a este nivel. El principal problema de esta alternativa se presenta en la etiquetas de grano mayor. En este caso para operar son traducidas a etiquetas más precisas. Si el nivel más fino es numérico (existen relaciones jerárquicas difusas o incluso crisp) utilizar esta aproximación implica asociar a cualquier etiqueta un *valor característico* mediante algún método (el centro de gravedad del conjunto difuso y la moda son métodos ampliamente utilizados) y trabajar con la aritmética clásica.

- Transformar todas las etiquetas al nivel más grueso. En este caso se añade imprecisión a los valores definidos en niveles más finos. Ahora también necesitaremos operadores para esta granularidad, complicando el diseño y la eficiencia del proceso global.

- Trabajar con los etiquetas tal como están definidas utilizando operadores extendidos (utilizando el *Principio de Extensión*, [Zad75a]). Conforme se agrega, se iría complicando la representación de los valores (la agregación de etiquetas y valores concretos no tiene por qué ser una etiqueta, figura 3.8) por lo que se perdería eficiencia en el modelo.

Por esto, para cumplir con los requerimientos presentados, utilizaremos el operador A_{β}^{OM} que hemos desarrollado y presentado en el punto 2.1.6.3.

3.4.1. Dimensiones con jerarquías lingüísticas

Como ya hemos comentado, utilizar etiquetas lingüísticas para la definición de relaciones jerárquicas implica una modificación de la relación de parentesco y su extensión a cualesquiera niveles de la dimensión.

Cada etiqueta lingüística tendrá asociado un número difuso en el intervalo [0,1]. Por esto, tenemos que extender las relaciones para trabajar en lugar de sobre valores concretos ($x \in [0,1]$) a números difusos en el mismo intervalo ($\tilde{x} \in \widetilde{[0,1]}$). Por esto, la *relación de parentesco difusa* quedaría definida como sigue.

Definición 3.29 *Para cada par de niveles l_i y l_j tal que $l_j \in H_{l_i}$ tenemos la relación*

$$\widetilde{\mu}_{ij} : l_i \times l_j \rightarrow \widetilde{[0,1]} \tag{3.19}$$

A esta relación la denominaremos relación de parentesco difusa.

En el caso de niveles no consecutivos ya vimos que utilizábamos la relación de parentesco extendida. En esta, agregábamos los valores de la relaciones intermedias mediante operadores MAM y MOM. En el caso de trabajar sobre etiquetas utilizaremos el operador A_β^{OM} que hemos definido. De esta forma, la *relación de parentesco extendido difusa* sería la siguiente.

Definición 3.30 *Para cada par de niveles l_i y l_j de la dimensión d, tal que $l_j \leq_d l_i \wedge l_j \neq l_i$, tenemos la relación $\widetilde{\eta}_{ij} : l_i \times l_j \rightarrow \widetilde{[0,1]}$ definida como*

$$\widetilde{\eta}_{ij}(a,b) = \begin{cases} \widetilde{\mu}_{ij}(a,b) & \text{si } l_j \in H_{l_i} \\ A_\beta^{OM}(P_{l_1}, ..., P_{l_n}) & \text{en otro caso} \end{cases} \tag{3.20}$$

donde $l_k \in H_{l_i}$ y $P_{l_k} = A_\beta^{OM}(\delta_{c_1}, ..., \delta_{c_m}) \forall c_i \in l_k$, siendo $\delta_c = A_{1-\beta}^{OM}(\widetilde{\mu}_{ik}(a,b), \widetilde{\eta}_{kj}(c,b))$.
A esta relación la denominaremos relación de parentesco extendido difusa.

3.4.2. Operaciones

Al modificar la relación de parentesco extendida, las operaciones roll-up y dice se ven modificadas (como puede verse de su definición). Ahora tendrá que tener en cuenta las etiquetas para obtener cuál es el conjunto de hechos que agregar. Por esto, en el caso de roll-up, lo primero que tendremos que hacer, será extender la definición del conjunto $F_{c_{ij}}$ para utilizar la nueva definición de las relaciones jerárquicas.

Definición 3.31 *Sea $\overrightarrow{c} = (c_b^1, ..., c, ..., c_b^n)$. para cada valor c_{ij} perteneciente a l_r tenemos el conjunto*

$$\tilde{F}_{c_{ij}} = \begin{cases} \bigcup\limits_{l_k \in H_{l_r}} \tilde{F}_{c_{kp}} / c_{kp} \in l_k \wedge \widetilde{\mu}_{ik}(c_{ij}, c_{kp}) \neq \tilde{0} & \textit{si } l_r \neq l_b \\ \{h / h \in F \wedge \exists \overrightarrow{c}\, A(\overrightarrow{c}) = h\} & \textit{si } l_r = l_b \end{cases} \tag{3.21}$$

El conjunto $\tilde{F}_{c_{ij}}$ representa a todos los hechos relaciones con el valor c.

Con esta extensión de $\tilde{F}_{c_{ij}}$ ya podemos introducir la nueva operación roll-up.

Definición 3.32 *Sea $D_f : \widetilde{[0,1]} \rightarrow [0,1]$ un método de defuzificación. El resultado de aplicar la operación roll-up sobre la dimensión d_i, el nivel l_r ($l_r \neq l_\perp$), utilizando el operador G sobre un DataCubo $C = (D, l_b, F, A, H)$ es otro DataCubo $C' = (D, l'_b, F', A', H')$ tal que:*

- $l'_b = (l_{1b}, ..., l_r, ..., l_{nb})$,
- $A'(c_b^1, ..., c_j^r, ..., c_b^n) = G(\{(b, D_f(A_{1-\beta}^{OM}(\alpha, \widetilde{\eta}_{rb}(c_j^r, c_b^r)))) / (b, \alpha) \in \tilde{F}_{c_j^r} \wedge A(c_b^1, ...c_b^r, ..., c_b^n) = (b, \alpha)\})$,
- *F' es la imagen de A',*
- $H' = (A, l_b, F, G, H)$.

Para el caso de definir roll-up$^{\alpha}$ tenemos que actuar de forma similar. El primer paso es extender F_c^{α} para que contemple las etiquetas lingüísticas y, sobre este conjunto, definir la operación. En este caso, este conjunto se vería modificada como sigue.

Definición 3.33 *Sea D_f un método de defuzificación, $\overrightarrow{c} = (c_b^1, ..., c, ..., c_b^n)$ y $\alpha \in [0,1]$. Para cada valor c_{ij} perteneciente a l_r tenemos el conjunto*

$$\tilde{F}_{c_{ij}}^{\alpha} = \begin{cases} \bigcup\limits_{l_k \in H_{l_r}} \tilde{F}_{c_{kp}}^{\alpha} / c_{kp} \in l_k \wedge D_f(\widetilde{\mu}_{ik}(c_{ij}, c_{kp})) \geq \alpha & \text{si } l_r \neq l_b \\ \{h/h \in F \wedge \exists \overrightarrow{c}\, A(\overrightarrow{c}) = h\} & \text{si } l_r = l_b \end{cases} \tag{3.22}$$

El conjunto $\tilde{F}_{c_{ij}}^{\alpha}$ representa a todos los hechos relaciones con el valor c_{ij} con relación mínima α.

Con esto, la operación roll-up$^{\alpha}$ ya puede ser introducida. Se definición sería similar a la de roll-up salvo que este caso consideramos como conjunto de referencia $\tilde{F}^{\alpha}$ en lugar de $\tilde{F}$.

Definición 3.34 *Sea D_f un método de defuzificación. El resultado de aplicar la operación roll-up$^{\alpha}$ sobre la dimensión d_i, el nivel l_r $(l_r \neq l_{\perp})$, utilizando el operador G sobre un DataCubo $C = (D, l_b, F, A, H)$ es otro DataCubo $C' = (D, l_b', F', A', H')$ tal que:*

- $l_b' = (l_{1b}, ..., l_r, ..., l_{nb})$,
- $A'(c_b^1, ..., c_j^r, ..., c_b^n) = G(\{(b, D_f(A_{1-\beta}^{OM}(\alpha, \widetilde{\eta}_{rb}(c_j^r, c_b^r)))) / (b, \alpha) \in \tilde{F}_{c_j^r}^{\alpha} \wedge A(c_b^1, ...c_b^r, ..., c_b^n) = (b, \alpha)\})$,
- *F' es la imagen de A',*
- $H' = (A, l_b, F, G, H)$.

La otra operación que se ve modificada al utilizar jerarquías con relaciones lingüísticas es *dice*. En esta, se consideran aquellos elementos que cumplen una condición o están relacionados con uno en otro nivel que la cumple. Por esto, tendremos que extenderla para contemplar el uso de las etiquetas lingüísticas en estas relaciones. Así pues, la nueva definición quedaría como sigue.

Definición 3.35 *El resultado de aplicar* **dice** *con una condición β sobre el nivel l_r de la dimensión d_i del DataCubo $C = (D, l_b, F, A, H)$ es otro DataCubo $C' = (D', l'_b, F', A', \Omega)$ donde:*

- $D' = \{d_1, ..., d'_i, ..., d_n\}$ *donde* $d'_i = (l'_i, \leq_{d_i}, l_b, l_\top)$ *con* $l' = \{l_j / l_b \leq_{d_i} d_{l_j}\}$ *y*

$$d'_i.l'_j = \begin{cases} \{c_{jk} / c_{jk} \in l_j \wedge \beta(c_{jk})\} & \text{si } l'_j = l_r \\ \{c_{jk} / c_{jk} \in d_i.l_j \wedge \delta_{rj}(c_{jk})\} & \text{si } l'_j \leq_d l_r \\ \{c_{jk} / c_{jk} \in d_i.l_j \wedge \delta_{jr}(c_{jk})\} & \text{si } l_r \leq_d l'_j \end{cases}$$

  donde $\delta_{ij}(c) = \exists c_r \in l_r \beta(c_r) \wedge \tilde{\eta}_{ij}(c_r, c) \neq \tilde{0}$,

- $A'(c_b^1, ..., c_b^i, ..., c_b^n) = (h, \alpha) / c_b^1 \in d'_1.l'_b \wedge ... \wedge c_b^n \in d'_n.l'_b \wedge A(c_b^1, ..., c_b^n) = (h, \alpha)$,

- F' *sería la imagen de* A'.

Como puede verse, lo único que se modifica es la consideración que se hace los elementos relacionados (considerando la *relación de parentesco difusa* en el nuevo caso).

También debemos modificar la operación dice$^\alpha$ para establecer el umbral de los valores que consideramos. Como el caso de roll-up$^\alpha$ lo que haremos será considerar un umbral concreto sobre las relaciones, por lo que utilizaremos un método

de defuzificación. Así pues, la diferencia con la definición que acabamos de presentar será la condición δ_{ij} donde, para un α dado, en el caso de dice$^{\alpha}$ quedaría como

$$\delta_{ij}(c) = \exists c_r \in l_r \beta(c_r) \wedge D_f(\tilde{\eta}_{ij}(c_r, c)) \geq \alpha \tag{3.23}$$

En esta sección hemos introducido la utilización de etiquetas lingüísticas en la definición de las relaciones jerárquicas. Para ello, hemos tenido que redefinir algunos conceptos para poder trabajar sobre los números difusos subyacentes a las etiquetas (relación de parentesco y relación de parentesco extendida). Algunas operaciones también se han tenido que volver a formular, utilizando para la agregación de las etiquetas el operador A_{β}^{OM}. De esta forma, tenemos un modelo multidimensional capaz de utilizar etiquetas lingüísticas completamente funcional. En la siguiente sección se presenta las propiedades de las operaciones tanto para el caso difuso como para el lingüístico que acabamos de presentar. Posteriormente recogemos un ejemplo de DataCubo modelado utilizando tanto relaciones difusas como lingüísticas.

3.5. Propiedades de las operaciones

En este apartado estudiaremos principalmente las propiedades relacionadas con la asociatividad entre operaciones. Estas propiedades son importantes tanto de cara al sistema (para políticas de optimización de operaciones decidiendo el orden de aplicación) como para el usuario (para saber las dependencias entre operaciones y entender los resultados obtenidos).

Las presentamos después de presentar los tres modelos (preciso, difuso y lingüístico) debido a que son compartidas por todos.

3.5.1. Roll-up y drill-down

Propiedad 3.1 *No hay pérdida de información al aplicar roll-up sobre un DataCubo.*

Demostración. La propiedad anterior es equivalente a que, partiendo de un DataCubo sobre el que hemos aplicado operaciones de roll-up, es posible obtener de nuevo el DataCubo básico. Lo demostraremos mediante inducción:

- Supongamos que el DataCubo básico es $C = (D, l_b, F, A, \Omega)$. Si sobre él aplicamos roll-up utilizando el operador G, obtenemos como resultado $C' = (D, l'_b, F', A', H)$ donde $H = (A, l_b, F, G, \Omega)$. Para volver a obtener el DataCubo básico tendríamos que aplicar la operación de drill-down.
- Suponiéndolo válido para n operaciones de roll-up consecutivas, siendo $C^n = (D, l_b, F, A, H)$, tendríamos que $C^{n+1} = (D', l'_b, F', A', H')$ donde $H' = (A, l_b, F, G, H)$. Si aplicáramos una operación de drill-down obtendríamos de nuevo C^n, con lo que se reduce al caso anterior.

Esta propiedad es importante dado que implica que podemos cambiar el nivel de detalle de un DataCubo tantas veces como se desee, estando seguro de que no habrá pérdida de información en el proceso. Es decir, podremos recuperar el estado de partida dado que mantenemos toda la información necesaria. Esto es gracias al uso de la estructura *historia*.

Propiedad 3.2 *El orden de aplicación de operaciones que implican la agregación de valores influye en el resultado.*

Demostración. Cuando aplicamos roll-up o slice, necesitamos un operador de agregación. En ambas situaciones obtenemos nuevos hechos a partir de los hechos agregados. Así pues, una nueva agregación de valores dependerá de este nuevo conjunto de hechos.

Propiedad 3.3 *Drill-down sólo se puede aplicar sobre un DataCubo resultado de una operación de tipo roll-up.*

Demostración. Por definición, drill-down necesita para ser aplicada un DataCubo cuya estructura *historia* sea diferente de Ω. Slice, pivot y slice establecen esta componente a este valor. Roll-up es la única operación cuyo resultado cumple con esta precondición.

3.5.2. Pivot

Propiedad 3.4 *La operación de pivotaje es asociativa con todas las operaciones excepto con drill-down.*

Demostración. Si aplicamos pivot, el DataCubo resultado tendrá Ω como estructura *historia*. Por la propiedad anterior, drill-down no podría aplicarse tras esta operación. Pivot sólo cambia la estructura del DataCubo, no los valores de la misma. Este cambio no es importante para el resto de las operaciones, incluida ella misma. Así pues, el orden de aplicación de pivot no es importante para el resultado final.

3.5.3. Dice

Propiedad 3.5 *Aplicar dos dice's con condiciones β y β' o aplicar ambas en una única operación con condición $\beta \wedge \beta'$ es equivalente.*

Demostración. La operación dice lo que hace es eliminar aquellos elementos que no cumplan una condición. Al aplicarla con la condición β nos quedaríamos con el comjunto de valores que cumplen dicha condición. Al aplicar la condición β', lo haríamos sobre los valores que ya cumplen la condición β, por lo que el resultado serán aquellos valores que cumplen la condición β y además la β'.

Así pues, es equivalente a aplicar una condición β'' que fuera la conjunción de ambas condiciones ($\beta''(x) = \beta(x) \wedge \beta'(x)$).

Propiedad 3.6 *El orden en que se aplican dos operaciones de dice es independiente para el resultado.*

Demostración. Se demuestra de forma directa partiendo de la propiedad anterior. Dado que $\wedge$ es conmutativa, es equivalente aplicar $\beta \wedge \beta'$ o $\beta' \wedge \beta$, por lo que aplicar β y luego β' es equivalente a utilizarlas en orden inverso.

En esta sección hemos estudiado las principales propiedades de las operaciones. En concreto, hemos estudiado las asociatividad de las operaciones y la conservación de la información al operar. Como ya hemos comentado, estas propiedades son interesantes tanto para la implementación del modelo (p.e. estudio de posibles políticas de optimización de las consultas) como de cara al usuario, pues sabiendo las propiedades entenderá mejor el modelo y las posibilidades que le brinda.

3.6. Vista de Usuario

Como se ha comentado en las secciones anteriores, para aplicar algunas de las operaciones necesitamos utilizar un operador de agregación sobre bolsas difusas. La mayoría de los métodos propuestos en la literatura tienen como resultado un conjunto difuso que hace que su interpretación sea poco intuitiva incluso para un usuario con ciertos conocimientos. Por esto, lo que proponemos es el cubrir nuestro modelo con una capa que abstraiga la complejidad de estos resultados, dando como salida valores más fácilmente entendibles. Para realizar esta tarea proponemos la utilización de un operador de resumen difuso que, manteniendo la máxima información posible del conjunto difuso, ofrezca unos resultados más intuitivos.

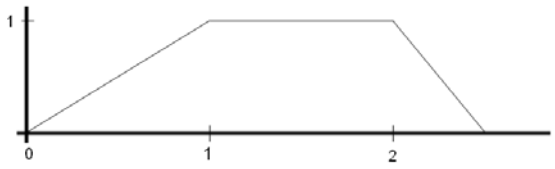


Figura 3.9: Resultado de aplicar resumen lingüístico

Para poder realizarlo necesitamos utilizar un operador cuya entrada sea del tipo hecho que hemos definido. Un ejemplo de este tipo de operadores se puede encontrar en Blanco et al. ([BSSV03]) presentado en el segundo capítulo. En este se propone el uso del número difuso que mejor se ajuste al conjunto o bolsa difusa resultado. Otro posible operador de resumen podría ser considerar la media ponderada de los valores utilizando α como peso en la ponderación. Para verlo más claro veamos un ejemplo: supongamos que en una celda de un DataCubo tenemos como hecho la bolsa difusa $\{1/1, 1/2, 0'9/0'5, 0'8/2'3, 0'2/0'3, 0'1/2'5\}$. Que un usuario pueda interpretar este hecho de forma directa resulta bastante complicado incluso si éste posee conocimiento de lógica difusa. En su lugar, utilizaríamos un operador de resumen lingüístico que lo muestre de una forma más entendible buscando la mínima pérdida de información. Para este ejemplo utilizaremos varios:

- Resumen lingüístico: como ya hemos comentado se trata de encontrar el número difuso que mejor se ajuste a la bolsa difusa y dar una expresión lingüística del mismo. Para este caso el resultado sería: (1,2,1,0'5) cuya expresión asociada es *"más o menos entre 1 y 2"*.

- Media ponderada: Aplicando este operador obtendríamos como expresión para la bolsa el valor 1'4, que sería lo que finalmente mostraríamos al usuario.

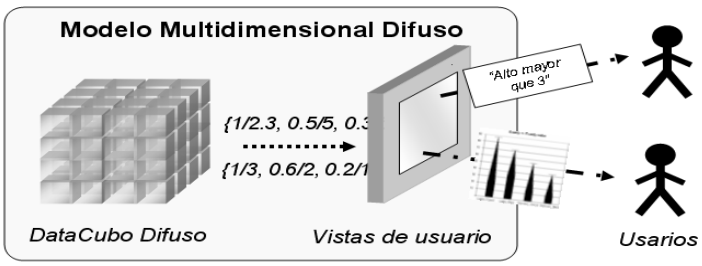


Figura 3.10: Estructura en dos capas

Como puede verse, por simple que sea el operador a utilizar, como la media ponderada, se obtiene una expresión del hecho más cercana al usuario. El perjuicio es que se pierde parte de la información al ajustar a representaciones más simples. Por eso, son deseables operadores más sofisticados con mejor aproximación al conjunto, como el resumen lingüístico.

Así pues, haciendo uso de este tipo de operador, definimos lo que hemos denominado *vista de usuario*.

Definición 3.36 *Sea M un operador de resumen difuso. Definimos la* **vista de usuario** *de un DataCubo $C = (D, l_b, F, A, H)$ según el operador M como la estructura $C_M = (D, l_b, F_M, A_M)$ donde:*

- $A_M(a_1, ..., a_n) = M(A(a_1, ..., a_n))$,
- F_M *es la imagen de* A_M.

De esta manera podríamos tener tantas vista de usuario como operadores de resumen se puedan utilizar (Figura 3.10).

Unas de las características que se piden a los sistemas tipo OLAP es que tengan mecanismos para mostrar los datos a los usuarios de una forma los más intuitiva posible. Muchas veces es más fácil interpretar por parte del usuario una

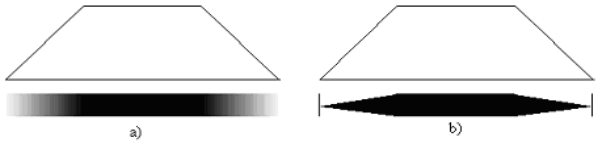


Figura 3.11: Representación gráfica de número difusos

representación mediante gráficas que el uso de valores concretos. Por esto, lo que proponemos es un operador de resumen difuso cuyo resultado es una representación de números difusos de forma gráfica, por que nos basaremos en los resultados que se obtienen al aplicar el operador Resumen Lingüístico. Una vez tengamos el número difuso, para representar los valores según su grado de pertenencia lo podremos hacer de dos formas distintas:

- Combinación de dos colores mediante gradiente. Se trata de asignar un color al grado 1 y otro al grado 0. De esta forma, los valores cuyo grado de pertenencia sea 1 se representarán por el color elegido, al igual que los de grado cero. Los que tengan un valor intermedio se utilizará una combinación de ambos colores en los que participarán según de cercano esté el valor a los correspondientes. El resultado será un degradado entre ambos colores según los valores de pertenencia. En la figura 3.11.a se recoge como sería el aspecto resultante con este enfoque.

- Barra de anchura variable. Esta propuesta se basa en considerar una barra cuya anchura dependerá el valor de pertenencia del valor. Todos los valores que tengan el mismo grado de pertenencia tendrán una anchura igual que variará desde un valor máximo (para los de grado 1) hasta cero para los de grado 0. En la figura 3.11.b puede observar como se haría en este caso.

En la figura 3.12 puede observarse cual sería el aspecto de una gráfica uti-

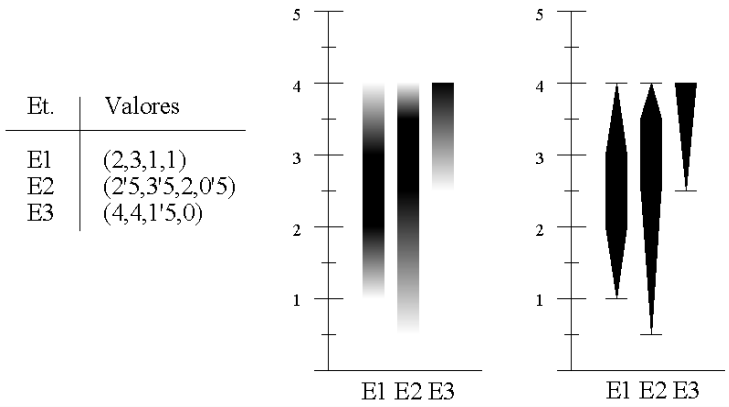


Figura 3.12: Ejemplo de gráfica utilizando ambos enfoques de representación gráfica

lizando ambas manera de representar los número difusos.

En algunos casos puede ser interesante no sólo considerar con valores difusos un único eje sino que los valores a representar estén relacionados con valores difusos en el otro eje. En estos casos, el primer método propuesto (combinación de dos colores) también nos es válido. Lo que tendríamos que hacer es aplicar el mismo esquema en ambos ejes y combinar el resultado utilizando una t-norma para obtener una única mezcla de colores como resultado de las de cada eje. En la figura 3.13 puede observarse un ejemplo en el que se representan números difusos según grupos de edad que han sido definidos mediante números difusos a su vez.

La segunda propuesta (barra de anchura variable) permite a su vez una representación equivalente a la los diagramas de barras en el caso preciso (figura 3.14).

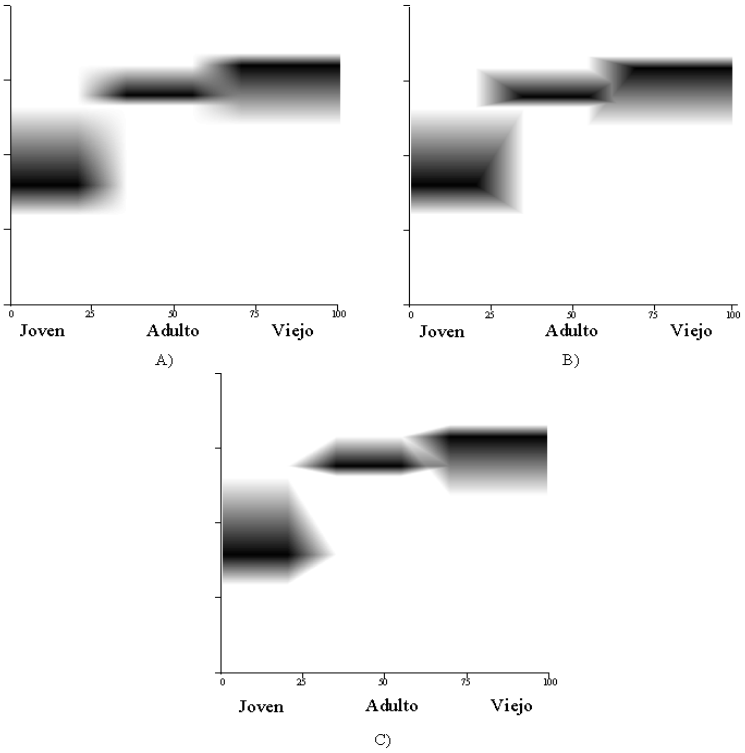


Figura 3.13: Ejemplo de gráficas con ambos ejes difusos según diversas t-normas. A) producto. B) mínimo. C) Lukasiewicz

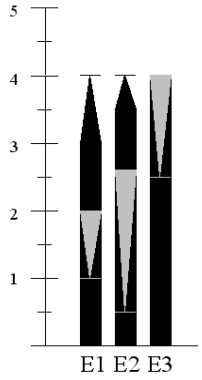


Figura 3.14: Diagrama de barras para el caso difuso

3.7. Ejemplo

Una vez que hemos presentado el modelo multidimensional difuso y el uso de jerarquías lingüísticas sobre este, veamos como se opera sobre ellos utilizando un ejemplo simple. Comenzaremos presentando el problema y el esquema multidimensional asociado. Luego veremos los elementos que tenemos que decidir para poder trasladarlo a nuestra propuesta. El última paso será resolver algunas consultas sobre estas, utilizando tanto el enfoque preciso como difuso con y sin jerarquías lingüísticas para poner de manifiesto las diferencias tanto a nivel de diseño como de operación.

En la figura 3.15 viene recogido el ejemplo de esquema multidimensional que presentamos en la sección 3.2.1.4. Este intentaba modelar el análisis de las causas de las quejas realizadas por los clientes. La información que poseía la compañía se ha completado con dos fuentes externas. Una de ellas ha sido la opinión de expertos referentes a la gravedad de las causa de la queja. El nivel Gravedad en la dimensión Causa contempla esta información. Por otro lado se ha introducido una clasificación de los proveedores atendiendo a su calidad. Esta información se

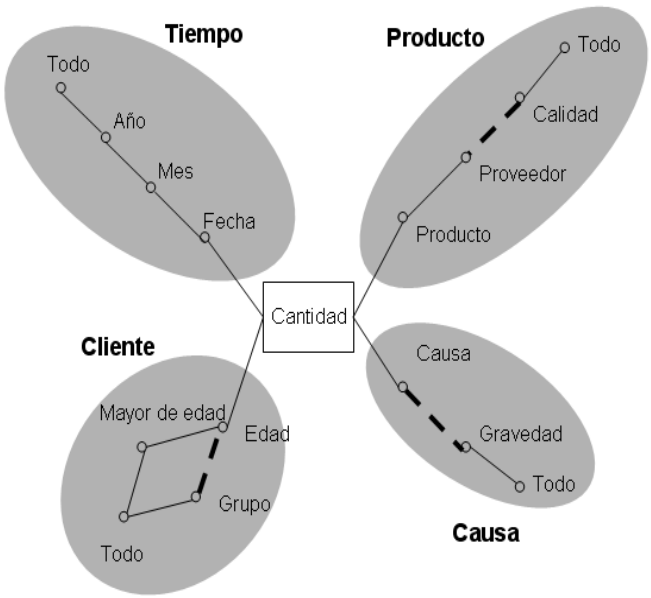


Figura 3.15: Ejemplo de esquema multidimensional

ha podido obtener a través de Internet según diferentes ranking encontrados. El nivel Calidad en la dimensión Producto refleja estos datos. En la figura, las líneas continuas representan relaciones tipo crisp entre los niveles. Las discontinuas las difusas, donde un elemento puede estar relacionado con los elementos un nivel superior con valores difusos.

La estructura del DataCubo que representa este esquema sería:

$$C_{Quejas} = (\{Cliente, Producto, Tiempo, Causa\}, \{Cantidad\} \cup \emptyset, \Omega, A) \tag{3.24}$$

donde las dimensiones están definidas como:

$$\begin{gathered} Cliente = (\{Edad, Mayor\ Edad, Grupo, Todos\}, \leq_{Cliente}, Edad, Todos) \\ Producto = (\{Producto, Proveedor, Calidad, Todos\}, \leq_{Producto}, Producto, Todos) \\ Tiempo = (\{Fecha, Mes, Ano, Todos\}, \leq_{Tiempo}, Fecha, Todos) \\ Causa = (\{Causa, Gravedad, Todos\}, \leq_{Causa}, Todos) \end{gathered} \tag{3.25}$$

El único elemento que nos quedaría por definir sería la relación A entre dimensiones y hechos. En este caso, la relación tendría la forma

$$A : Edad \times Producto \times Fecha \times Causa \rightarrow \{Cantidad\} \cup \emptyset \tag{3.26}$$

Los valores que consideramos para los niveles de las dimensiones son:

- Dimensión Causa:
  - Causa = {rotura, problema eléctrico, daño leve, falta pieza}=$l_{\perp}$
  - Gravedad = {leve, grave}
  - Todos = {todo} = $l_{\top}$
- Dimensión Tiempo:
  - Fecha = {1-Dic-02,...,28-Feb-03} = $l_{\perp}$
  - Mes = {Dic-02, Ene-03, Feb-03}
  - Año = {2002, 2003}
  - Todos = {todo} = $l_{\top}$
- Dimensión Producto:
  - Producto = {radio, TV, discman, video} = $l_{\perp}$
  - Proveedor = {Proveedor 1, Proveedor 2}

- Calidad = {bueno, malo}
- Todos = {todo} = $l_{\top}$

- Dimensión Cliente:
  - Edad = {1,...,100} = $l_{\perp}$
  - Mayor Edad = {Sí, No}
  - Grupo = {joven, adulto, viejo}
  - Todos = {todo} = $l_{\top}$

Estos valores serán comunes tanto al modelo preciso como al difuso con y sin etiquetas lingüísticas en las relaciones jerárquicas. Veremos para cada cómo se definen las relaciones.

Modelo Preciso

Para este modelo diremos los conjuntos que definen cada uno de los nombre o etiquetas que existen en cada nivel.

- Dimensión Causa:
  - Gravedad: leve = {rotura, daño leve}, grave = {problema eléctrico, falta pieza}
  - Todos: todo = leve $\bigcup$ grave = {rotura, daño leve, problema eléctrico, falta pieza}

- Dimensión Tiempo:
  - Mes: Dic-02 = { 1-Dic-02, ..., 31-Dic-02}, Ene-03 = {1-Ene-03,...,31-Ene-03}, Feb-03 ={1-Feb-03,...,28-Feb-03}
  - Año: 2002 = Dic-02 = { 1-Dic-02, ..., 31-Dic-02}, 2003 = Ene-03 $\bigcup$ Feb-03 = {1-Ene-03,...,28-Feb-03}

  - Todos: todo = 2002 $\bigcup$ 2003 =$\{$ 1-Dic-02,...,28-Feb-03$\}$

- Dimensión Producto:

  - Proveedor: Proveedor 1 = $\{$radio, discman$\}$, Proveedor 2 = $\{$TV, video$\}$
  - Calidad: bueno =Proveedor 1= $\{$radio, discman$\}$, malo = Proveedor 2 = $\{$TV, video$\}$
  - Todos: todo = bueno $\bigcup$ malo = $\{$radio, discman, TV, video$\}$

- Dimensión Cliente:

  - Mayor Edad: Sí = $\{$18,...,100$\}$, No = $\{$1,...,17$\}$
  - Grupo: joven = $\{$1,...,25$\}$, adulto = $\{$26,...,64$\}$, viejo = $\{$65,...,100$\}$
  - Todos: todo = Sí $\bigcup$ No = joven $\bigcup$ adulto $\bigcup$ viejo = $\{$1,...,100$\}$

Para aplicar las operaciones que requieran agregación de los hechos utilizaremos los operadores habituales (máximo, mínimo, etc.) en aritmética clásica. Utilizaremos gráficas de los resultados para presentarlos de forma más intuitiva.

Modelo difuso sin jerarquías lingüísticas

En este caso, sólo vamos a reflejar aquellas relaciones que son distintas al caso preciso. Las relaciones que se verían modificados serían la de parentesco entre los niveles Gravedad y Causa (tabla 3.1) en la dimensión Causa, Calidad y Proveedor (tabla 3.2) de la dimensión Producto, y Grupo con Edad (figura 3.6).

Para calcular la *relación de parentesco extendido* necesitamos especificar qué t-norma y t-conorma vamos a utilizar. Para el ejemplo utilizaremos el *mínimo* y el *máximo*, respectivamente. Además, necesitaremos operadores de agregación para realizar las operaciones roll-up y slice. En este caso utilizaremos la adaptación que hemos comentado de los operadores propuestos por Rundensteiner & Bic ([RB89]).

Causa	Leve	Grave
Rotura	0,8	0,2
Problema eléctrico	0	1
Daño grave	1	0
Falta pieza	0,3	0,7

Cuadro 3.1: $\mu_{Gravedad,Causa}$

Proveedor	Bueno	Malo
Proveedor 1	0,3	0,7
Proveedor 2	0,9	0,2

Cuadro 3.2: $\mu_{Calidad,Proveedor}$

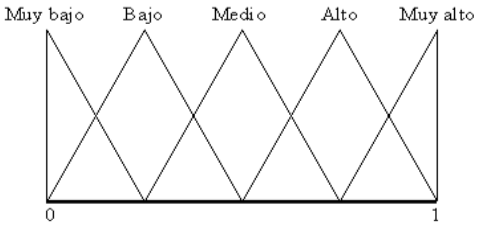


Figura 3.16: Etiquetas lingüísticas

Para mostrar los datos de la manera más intuitiva posible utilizaremos como *vista de usuario* el *Resumen Lingüístico (RL)* propuesto en [BSSV03] y las propuestas gráficas que hemos presentado.

Modelo difuso con jerarquías lingüísticas

En el caso difuso hemos utilizado conceptos difusos en las jerarquías. En el caso de la relación Gravedad y Causa ha sido proporcionada por expertos. En la mayoría de los casos es complicado que un experto proporcione valores concretos, siendo más cómodo la utilización de etiquetas lingüísticas para proporcionar los valores. En este caso, utilizaremos el conjunto recogido en la figura 3.16. Así pues, la relación de parentesco $\mu_{Gravedad,Causa}$ quedaría como la relfejada en la tabla 3.3.

Causa	**Leve**	**Grave**
Rotura	Alto	Bajo
Problema eléctrico	Muy bajo	Muy alto
Daño grave	Muy alto	Muy bajo
Falta pieza	Bajo	Alto

Cuadro 3.3: $\mu_{Gravedad,Causa}$

En este modelo utilizaremos el operador de agregación A_β^{OM}, basado en el método de ordenación propuesto por Delgado et al. ([DVV98]) y un valor $\beta = 0,9$, para el cálculo de la *relación de parentesco extendido* en el caso de la dimensión Producto. En cuanto a la función D_f emplearemos el método de defuzificación *media de los máximos*. El resto de elementos (operadores de agregación y vistas de usuario) utilizaremos los mismos que en el caso difuso sin jerarquía lingüística.

Para el ejemplo supondremos que los datos con los que podemos trabajar proceden de dos fuentes distintas (Figura 3.17). Una de ellas es confiable y los datos se pueden ajustar perfectamente al esquema que hemos propuesto, debido a que son procedentes de bases de datos internas a la empresa. Además podemos utilizar los datos que nos facilita una distribuidora respecto a nuestros productos. En este caso, los datos no son tan fiables como los internos. los motivos pueden ser varios: desconocemos cómo son tomados, qué criterios siguen y/o la veracidad de los mismos. La empresa que nos los provee es de cierta confianza pero tenemos que tener en cuenta que queremos utilizar sus datos para decidir el rumbo de la nuestra, por lo que no es una cuestión sencilla. En el modelo clásico podemos optar por dos soluciones, como ya hemos comentado.

En el caso de utilizar el modelo difuso este problema se minimiza en parte. En el caso clásico las opciones implican o pérdida de información o la utilización

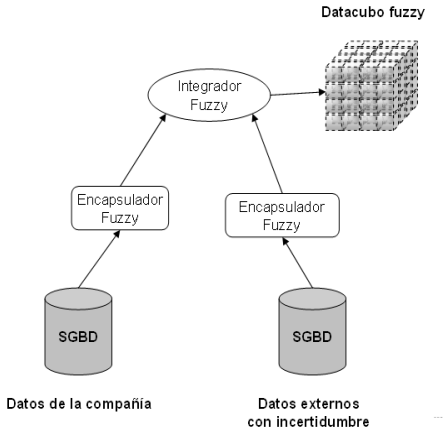


Figura 3.17: Estructura del sistema para el ejemplo

de información no fiable. Con esta aproximación podemos utilizar estos datos en el análisis pero limitando su influencia para que la posible distorsión que puedan añadir esté controlada. Esto lo hacemos estableciendo un valor alfa de los hechos que contemplen esta información externa en el intervalo [0,1]. Cuanto más cercano de 0 le asignemos, menor será su influencia y menos se tendrán en cuenta en el análisis. Si se acerca a 1 el valor, el caso será el contrario. Para el ejemplo hemos utilizado un valor de alfa 0,7. Es decir, la información la consideramos fiable pero no al nivel de la interna de nuestra empresa. En la tabla 3.4 y 3.5 vienen recogidos los hechos, internos e externos respectivamente, que utilizaremos

En el caso preciso, consideraremos dos DataCubos distintos en los ejemplos: uno sin considerar los datos externos, y otro considerándolos todos.

Ya tendríamos todos los elementos necesarios para poder aplicar nuestro modelo al ejemplo propuesto. Para ver el funcionamiento, realizaremos varias consultas sobre los datos propuestos. Pero antes de comentar los resultados, tenemos que tener en cuenta las diferencias existentes entre los enfoques. El modelado del

n	fecha	Edad	Causa	Producto	Cantidad	Alfa
1	02/01/2003	20	Rotura	Video	1	1
2	10/01/2003	40	Rotura	TV	2	1
3	14/01/2003	50	falta pieza	Video	1	1
4	18/01/2003	72	falta pieza	TV	3	1
5	26/01/2003	23	Rotura	TV	1	1
6	30/01/2003	32	falta pieza	Video	2	1
7	03/02/2003	20	Daño leve	TV	2	1
8	11/02/2003	40	Rotura	Video	1	1
9	15/02/2003	50	Problema eléctrico	Video	2	1
10	23/02/2003	80	Problema eléctrico	TV	2	1
11	27/02/2003	23	Rotura	TV	3	1
12	03/03/2004	20	Daño leve	Discman	5	1
13	15/03/2004	50	Daño leve	Discman	4	1
14	19/03/2004	72	Rotura	Radio	1	1
15	23/03/2004	80	falta pieza	Radio	2	1
16	31/03/2004	32	Rotura	Radio	3	1
17	04/04/2004	20	Rotura	Discman	3	1
18	08/04/2004	19	Rotura	Radio	2	1
19	12/04/2004	40	Problema eléctrico	Radio	4	1
20	16/04/2004	50	Problema eléctrico	TV	2	1
21	20/04/2004	72	falta pieza	Video	4	1
22	24/04/2004	80	Problema eléctrico	Video	1	1
23	04/05/2004	20	Daño leve	Discman	3	1
24	08/05/2004	19	falta pieza	TV	3	1
25	16/05/2004	50	Daño leve	TV	3	1
26	20/05/2004	72	Daño leve	TV	1	1
27	24/05/2004	80	falta pieza	Video	2	1
28	30/05/2004	32	Rotura	Radio	4	1
29	03/06/2004	20	Rotura	Discman	4	1
30	07/06/2004	19	Daño leve	Discman	3	1
31	15/06/2004	50	Rotura	Video	2	1
32	19/06/2004	72	Daño leve	TV	3	1
33	23/06/2004	80	Daño leve	Video	4	1
34	01/07/2004	32	Daño leve	Video	4	1

Cuadro 3.4: Hechos ejemplo sobre el esquema multidimensional internos a la compañía

n	fecha	Edad	Causa	Producto	Cantidad	Alfa
1	06/01/2003	19	Rotura	Video	2	0,7
2	22/01/2003	80	falta pieza	TV	1	0,7
3	07/02/2003	19	falta pieza	Video	4	0,7
4	19/02/2003	72	falta pieza	Video	3	0,7
5	28/02/2004	32	Rotura	TV	2	0,7
6	07/03/2004	19	falta pieza	Discman	2	0,7
7	11/03/2004	40	Daño leve	Discman	3	0,7
8	27/03/2004	23	Rotura	Discman	1	0,7
9	28/04/2004	23	Rotura	TV	5	0,7
10	30/04/2004	32	Daño leve	Discman	1	0,7
11	12/05/2004	40	Rotura	TV	2	0,7
12	28/05/2004	23	Daño leve	Radio	2	0,7
13	11/06/2004	40	Problema eléctrico	TV	1	0,7
14	27/06/2004	23	Problema eléctrico	Discman	1	0,7

Cuadro 3.5: Hechos ejemplo sobre el esquema multidimensional procedentes de la fuente con incertidumbre

problema que hacemos en el caso difuso es distinto al preciso. Como ya hemos comentado, la relación entre los elementos de los niveles Gravedad y Causa, en la dimensión Causa, y Calidad y Proveedor, en la dimensión Producto, están modelados de forma distinta. En el caso preciso, estamos ante relaciones normales: un elemento estará sólo relacionado otro del nivel superior (relación uno-a-mucho). En el caso difuso, estas relaciones se han extendido de manera que un mismo elemento puede estar relacionado con múltiples elementos de los niveles superiores, como es el caso de Proveedor 1 en el nivel Proveedor está relacionado tanto con Bueno y Malo en el nivel Calidad. Esta relación en particular hemos comentado que suponemos que se ha obtenido de fuentes externas (como puede ser internet) para incorporarla al modelo. Probablemente se han utilizado diferentes estudios sobre los proveedores que tenemos para obtener esta información, por lo que sería extraño que obtuviéramos un resultado tan concreto como necesita el caso preciso.

En la relación entre los niveles Gravedad y Causa de la dimensión Causa la información la han proporcionado expertos. En el caso preciso hemos forzado a que nos den una única opción para cada valor. En algunos casos esto será fácil para un experto, pero habrá otros en los que no será tan directo clasificar una causa según su gravedad dado que este concepto es por definición subjetivo. En el caso difuso damos al experto más libertad para la definición de la relación, permitiendo dar un valor en el intervalo [0,1] para cada una de las causas para cada valor de gravedad que consideramos. Sin embargo, para un experto diferenciar entre un valor 0,3 y 0,4 será muy complicado. Le será más intuitivo calificar la relación entre los conceptos diciendo que están o muy relacionados, o poco relacionados, o nada relacionados. Esto lo permitimos en el caso de utilizar etiquetas lingüísticas para la definición de las jerarquías. Cabe destacar que al utilizarla no estamos dando un valor concreto sino un *rango de valores* (en realidad un conjunto difuso) por lo que introducimos imprecisión que se reflejará en los resultados de la consulta.

	Calidad			
	Sin fuente con imprecisión		Con fuente con imprecisión	
Gravedad	Bueno	Malo	Bueno	Malo
Leve	27	32	38	39
Grave	22	6	31	9

Cuadro 3.6: Resultados de la consulta 1 en el caso preciso

3.7.1. Consulta 1

Supongamos la siguiente consulta:

"número de quejas según la gravedad de las mismas y la calidad de los proveedores de los productos"

Para realizar el análisis tendríamos que aplicar *Roll-up* sobre la dimensión *Causa*, el nivel *Gravedad*, y la dimensión *Producto* en el nivel *Calidad* utilizando el operador de agregación *Suma*.

Los resultados obtenidos para la consulta viene recogidos en la tabla 3.6 y figura 3.18 para el caso crisp, la tabla 3.7 y figura 3.19 para el difuso sin jerarquía lingüística, y la tabla 3.8 y figura 3.20 en el caso de utilizarla.

A la vista de la gráfica de la Figura 3.18 podemos ver cómo, de considerar o no los datos de la fuente externa, se obtienen diferentes resultados. Cabe destacar que en cada caso estaríamos trabajando con cubos distintos sobre los que hemos aplicado la misma consulta. En la gráfica de la Figura 3.19 podemos ver cual ha sido el resultado en el caso difuso. En este último caso lo que hemos conseguido ha sido el obtener una visión global de los datos pero sin desvirtuarlos, ya que lo procedentes de la fuente con incertidumbre tienen menor importancia que los internos. En el caso de uso de jerarquías lingüística vemos que los resultados son

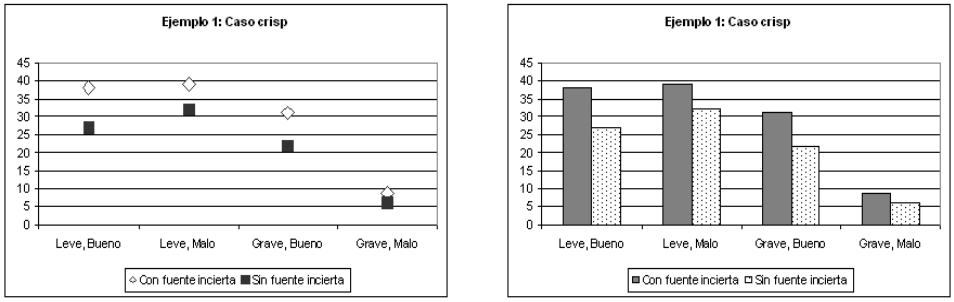
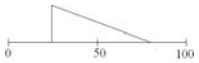


Figura 3.18: Resultado de la consulta 1 en el caso preciso

Gravedad	**Calidad**		
	Bueno		
	C'	C'_{RL}	
Leve	{1’0/23’0, 0’8/27’0, 0’7/38’0, 0’3/49’0, 0’2/94’0}	(23’05,23’05,0’04,65’08) ”Mayor que 23”	
Grave	{1’0/16’0, 0’7/31’0, 0’2/60’0}	(16’29,16’29,0’25,40’75) ”Mayor que 16”	
	Malo		
	C'	C'_{RL}	
Leve	{1’0/21’0, 0’8/32’0, 0’7/39’0, 0’3/49’0, 0’2/94’0}	(21’11,21’11,0’0,43’03) ”Mayor que 21’11”	
Grave	{1’0/2’0, 0’7/9’0, 0’2/23’0, 0’3/49’0, 0’2/94’0}	(2’01,2’01,0’01,28’82) ”Mayor que 2”	

Cuadro 3.7: Resultados de la consulta 1 en el caso difuso

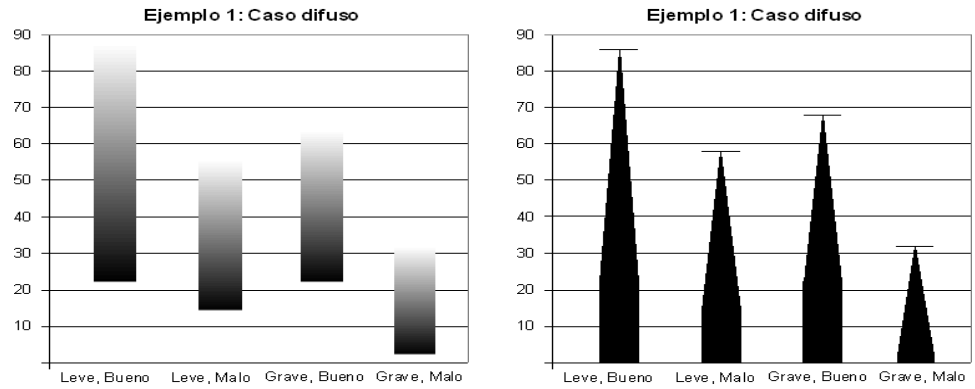


Figura 3.19: Resultado de la consulta 1 en el caso difuso

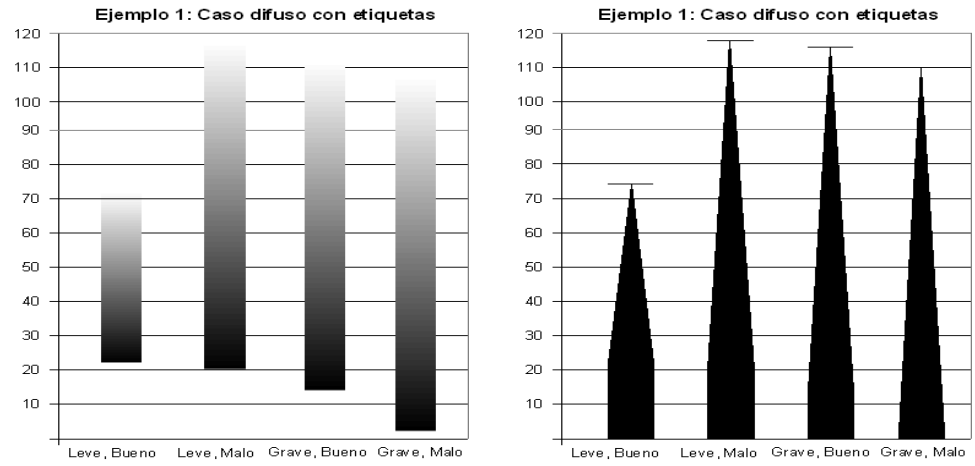
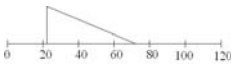
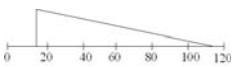
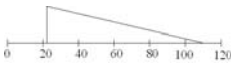
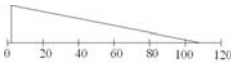


Figura 3.20: Resultado de la consulta 1 en el caso difuso con jerarquías lingüísticas

Gravedad	**Calidad**		
	Bueno		
	C'	C'_{RL}	
Leve	{1'0/23, 0'73/32, 0'72/34, 0'56/40, 0'53/45, 0'45/66, 0'43/77, 0'42/81, 0'41/97, 0'407/100, 0'405/104, 0'38/108, 0'37/110, 0'35/112, 0'32/113}	(23'0,23'0,0'0,59'7) "Mayor que 23"	
Grave	{1'0/16, 0'81/22, 0'73/26, 0'71/31, 0'45/33, 0'43/37, 0'42/40, 0'4/42, 0'39/53, 0'36/56, 0'27/79, 0'24/88, 0'21/109, 0'18/113 }	(16'0, 16'0, 0'0, 97'0) "Mayor que 16"	
	Malo		
	C'	C'_{RL}	
Leve	{1'0/21, 0'85/32, 0'73/36, 0'71/39, 0'56/43, 0'53/45, 0'41/47, 0'38/48, 0'27/71, 0'24/80, 0'225/82, 0'22/88, 0'20/104, 0'19/109, 0'17/113 }	(21'4,21'4,0'348,87'1) "Mayor que 21'4"	
Grave	{1'0/2, 0'87/6, 0'76/7, 0'72/9, 0'61/20, 0'55/23, 0'48/60, 0'45/66, 0'42/74, 0'39/79, 0'36/102, 0'33/104, 0'3/113}	(2'00,2'00,0'00,105'3) "Mayor que 2"	

Cuadro 3.8: Resultados de la consulta 1 con jerarquías lingüísticas

similares a los del caso difuso simple. La diferencia radica en que el soporte de los conjuntos difusos resultados son mayores. Esto se debe a esa introducción de imprecisión en los cálculos al utilizar la etiquetas lingüísticas, que se traduce en la consideración de más hechos pero con menor certeza menor , lo que se traduce en menor influencia.

3.7.2. Consulta 2

La consulta que queremos realizar es obtener el

"número de quejas realizadas por jóvenes en cada mes"

Las operaciones que tenemos que realizar son:

1. *Dice* sobre la dimensión cliente con condición $\beta(x) ='' \ x \ es \ Joven''$ en el nivel *Grupo*.
2. Seguido de un *roll-up* en las dimensiones *Tiempo*, nivel *Mes*, y *Cliente*, y nivel *Grupo*.

En esta consulta no mostraremos el caso de utilizar el modelo con jerarquías lingüísticas, dado que las dimensiones implicadas no utilizan etiquetas en las relaciones y el resultado sería el mismo que el difuso simple.

Para esta consulta tendremos que tener en cuenta que en el caso de los grupos de edades existen diferencias de definición del modelo preciso con el difuso (figura 3.21). En el caso preciso, el grupo *Joven* está representado mediante un intervalo de edades ([0,25]). En el caso difuso se ha utilizado un conjunto difuso, más cercana al usuario y que evita el efecto frontera (dos elementos contiguos pertenezcan a clases distintas).

El resultado viene recogida en las tablas 3.9, para el caso preciso, y 3.10, para el caso difuso. La representación gráfica de los mismos se puede ver en las figuras

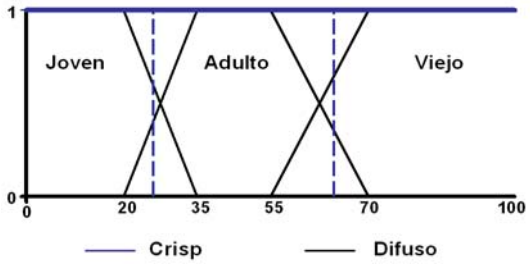


Figura 3.21: Grupos de edades en el modelo Preciso Vs. Difuso

3.22 y 3.23. Cómo puede verse, en el caso discreto considerar o no los datos externos influye de forma bastante importante en el resultado de la consulta. Es más, ambos casos presentan comportamientos distintos sin que se mantenga en ningún caso la misma tendencia en los resultados. Así que, si tenemos en cuenta los datos externos obtendríamos un posible análisis, mientras que si no, la enterpretación de los datos sería distinta.

En el caso difuso vemos como se aprecia una tendencia clara en los datos. En la Figura 3.24 se muestra de forma explícita cual sería. La inclusión de los datos externos a la empresa no ha desvirtuado los internos y han servido para enriquecer el análisis. Sin embargo, como hemos controlado su influencia lo que tenemos es que carecen de la importancia tal como para que el ruido que puedan incluir en el análisis impida realizar una interpretación guiada principalmente por los datos internos a la empresa. Aún así, estos datos han servido para matizar los resultados y hacer más evidente la tendencia que se observa en los datos.

3.8. Conclusiones

En este capítulo hemos presentado un modelo multidimensional capaz de modelar imprecisión e incertidumbre mediante la aplicación de la Teoría de los

Mes	Valor	
	Sin fuente con imprecisión	Con fuente con imprecisión
Enero	2	4
Febrero	5	9
Marzo	5	8
Abril	5	10
Mayo	6	8
Junio	7	8

Cuadro 3.9: Resultados de la consulta 2 en el caso preciso

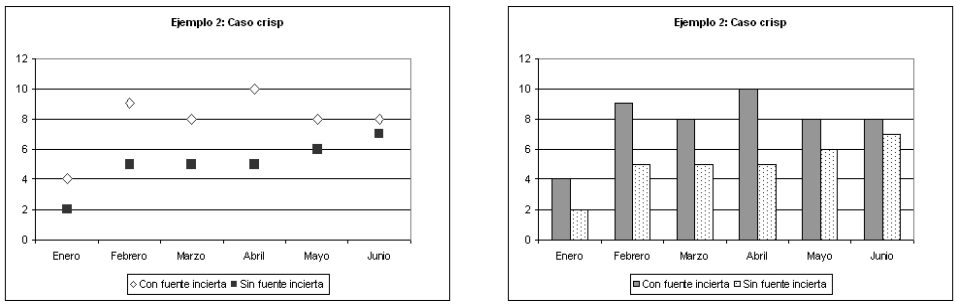
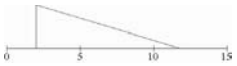


Figura 3.22: Resultado de la consulta 2 en el caso preciso

Mes	Mes	
Mes	C'	C'_{RL}
Enero	{1'0/1'0, 0'8/2'0, 0'7/4'0, 0'2/6'0},1	(1,1,0,4'8) *"Algo mayor que 1"* 
Febrero	{1'0/2'0, 0'8/5'0, 0'7/9'0, 0'2/11'0},1	(2,2,0,8'8) *"Algo mayor que 2"* 
Marzo	{1'0/5'0, 0'7/8'0, 0'2/11'0},1	(5,5,0,6) *"Algo mayor que 5"* 
Abril	{1'0/5'0, 0'7/10'0, 0'2/11'0},1	(5,5,0,4'6) *"Algo mayor que 5"* 
Mayo	{1'0/6'0, 0'7/8'0, 0'2/12'0},1	(6,6,0,5'6) *"Algo mayor que 6"* 
Junio	{1'0/7'0, 0'7/8'0, 0'2/12'0},1	(7,7,0,4'3) *"Algo mayor que 7"* 

Cuadro 3.10: Resultados de la consulta 2 en el caso difuso

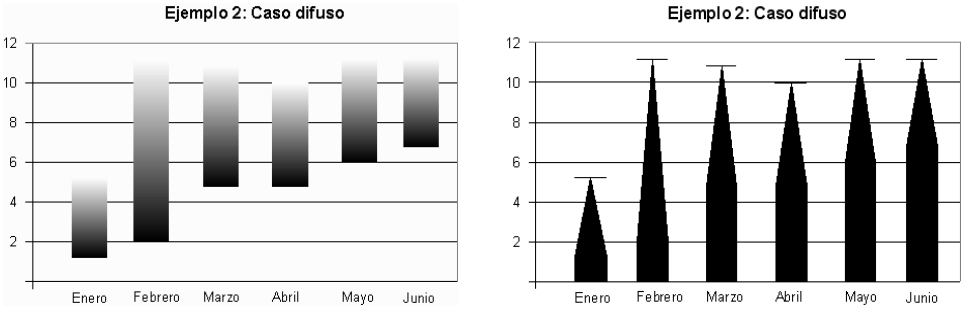


Figura 3.23: Resultado de la consulta dos en el caso difuso

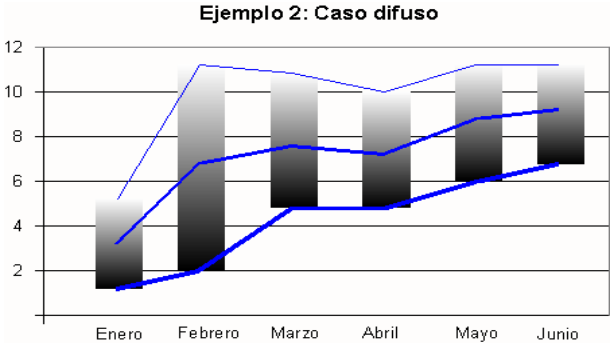


Figura 3.24: Resultado de la consulta dos en el caso difuso marcando la tendencia

Conjuntos Difusos. Gracias a ella, hemos podido modelar la imprecisión y la incertidumbre tanto en los hechos como en las dimensiones.

Para facilitar la integración de información proporcionada por expertos, hemos extendido las relaciones jerárquicas de tal forma que se puedan utilizar expresiones lingüísticas en lugar de valores concretos. Para mantener la funcionalidad del sistema, hemos desarrollado un operador (A_β^{OM}) para el cálculo de las relaciones entre niveles no consecutivos.

El manejo de la imprecisión en los hechos ayuda en la integración de información proveniente de diversas fuentes en las que no todas sin igualmente confiables. La capacidad de modelar las dimensiones con relaciones difusas y lingüísticas ayuda a modelar dominios complejos no bien definidos, en los que parte de las relaciones vengan dadas por parte de expertos.

Toda la complejidad nacida del uso de la lógica difusa se ha ocultado de cara a el usuario final mediante el uso de *vistas de usuario*, de forma que el resultado mostrado sea lo más intuitivo posible.

Respecto a otros modelos anteriores, el más desarrollado es el propuesto por Laurent y que comentado en el capítulo anterior. Este modelo tenía diversas carencias. Con respecto a su complejidad, la definición de dos jerarquías distintas en cada dimensión (una para particiones difusas y otras para relaciones) y la no ocultación de todos los mecanismos puede hacerlo complejo de cara a un usuario.

En el primer punto, en el caso de definir las relaciones difusas, en el modelado se ha de establecer los valores que relacionan cada par de valores en la dimensión, lo que hace que sea complicado su modificación o la inclusión de nuevos conceptos por parte de los usuarios. En nuestro modelo se ha establecido una única relación jerárquica en cada dimensión y la relación entre los valores se obtiene de forma automática mediante la utilización de las relaciones entre los niveles adyacentes. De esta forma, si se añade un nuevo nivel para modelar un nuevo concepto, sólo se ha de definir las relaciones con los elementos del nivel sobre el que

se define y el resto de obtiene de forma automática.

Además, valorar una relación en el intervalo [0,1] para un usuario no siempre es fácil. En el modelo de Laurent no existe otra posibilidad. Nuestro modelo dala posibilidad de utilizar etiquetas lingüísticas.

Toda la complejidad del modelo en el caso de Laurent no se oculta de cara a el usuario, lo que puede hacerlo poco intuitivo. Nuestro modelo lo hemos dotado de mecanismos para ocultar la complejidad, incluyendo mecanismos de representación gráfica.

Capítulo 4

Extracción de reglas de asociación sobre DataCubos difusos

Cualquier tecnología los bastante avanzada es indistinguible de la magia

TERCERA LEY DE CLARKE
Ley de Murphy

4.1. Introducción

Los modelos multidimensional están orientados a realizar análisis para la extracción de conocimiento. Los servidores OLAP permiten realizar dicho proceso de forma interactiva. Sería interesante que este proceso de extracción de conocimien-

to se pudiera realizar de forma automática. De esta forma, el usuario obtendrá información más elaborada. Estas técnicas han sido extendidas para ser utilizadas sobre DataCubos, agrupándose bajo el nombre de *OLAP mining* u *OLAM* (on-line analytical mining).

Una vez que hemos presentado el modelos multidimensional, en este capítulo presentaremos una técnica de minería de datos sobre el modelo basado en reglas de asociación. En esta sección realizaremos una breve introducción a la minería de datos y a su aplicación sobre datacubos. Posteriormente presentaremos nuestra propuesta.

4.1.1. Minería de datos: reglas de asociación

4.1.1.1. Conceptos generales

Dentro de las bases de datos, la extracción de conocimiento se conoce como *Data Mining* o *minería de datos*. Esta se defino como *el proceso de extracción no trivial de información implícita, previamente desconocida y potencialmente útil utilizando los datos almacenados en una base de datos* ([PSF91]) .

Existen múltiples técnicas de minería de datos (agrupamiento [GR02], árboles de decisión, etc.). Para una revisión consultar [CHY96, Man97]. Dado que propondremos un algoritmo de extracción de reglas de asociación sobre nuestro modelo de datacubos difuso, nos centraremos en las técnicas relativas a las reglas de asociación.

Dada una base de transacciones, es interesante descubrir asociaciones importantes entre elementos tales que la presencia de algunos elementos en una transacción implicará la presencia de otros elementos en la misma transacción. Agrawal *et al.* ([AIS93]) formalizaron el problema de la extracción de reglas de asociación. Sea $\mathcal{I} = \{i_1, i_2, ..., i_m\}$ un conjunto de literales, llamados elementos. Sea D un conjunto de transacciones, donde cada transacción T es un conjunto de elementos

tal que $T \subseteq \mathcal{I}$. Una transacción T contiene un conjunto de elementos X si y sólo si $X \subseteq T$. Con estos conceptos ya podemos introducir la definición de una *regla de asociación*.

Definición 4.1 *Sea $\mathcal{I} = \{i_1, i_2, ..., i_m\}$ un conjunto de literales y D un conjunto de transacciones definidas sobre $\mathcal{I}$. Una regla de asociación es una implicación de la forma $X \rightarrow Y$, donde $X \subset \mathcal{I}$, $Y \subset \mathcal{I}$ y $X \cap Y = \emptyset$.*

Una regla $X \rightarrow Y$ tendrá en D una *confianza* con valor c si el $c\,\%$ de las transacciones en D que contienen X también contienen Y. La regla tendrá un *soporte* con valor s en el conjunto de transacciones D si el $s\,\%$ de las transacciones en D contienen a los elementos $X \cup Y$.

La confianza da una medida de la calidad de la regla (la fuerza de la implicación). En 4.2.2.1 hablaremos de esta medida con mayor detalle. Normalmente se desea encontrar asociaciones entre elementos con un soporte razonablemente alto. Las reglas con confianza alta y soporte alto son denominadas *reglas fuertes* ([AIS93, PS91]).

La extracción de reglas de asociación se ha descompuesto normalmente en dos pasos ([AIS93, AS94]):

- Descubrir los *conjuntos de elementos frecuentes*, es decir, los conjuntos de elementos con soporte superior a un umbral prefijado.
- Utilizar los conjuntos de elementos frecuentes para generar reglas de asociación.

El primer punto es el que consume mayores recursos de tiempo y espacio. La generación de reglas una vez se tienen los conjuntos de elementos frecuentes es simple. Para la obtención de estos conjuntos se han propuesto múltiples algoritmo intentando mejorar su eficiencia. Ejemplos de estos son el algoritmo Apriori,

AprioriTID y AprioriHybrid (todos presentados en [AS94]), DHP (utiliza tablas hash para acelerar la localización de candidatos a conjuntos frecuentes, [PCY95]), SETM (pensado para trabajar directamente sobre bases de datos relacionales, [HS93]), DIC (saber si un conjunto de elementos es frecuente sin tener que contar todas las transacciones en las que aparece, [BMUT97]), Partition (realiza intersecciones de conjuntos para calcular soportes, [SON95]), FP-growth (realiza un preprocesamiento para obtener una representación muy compacta de las transacciones mediante un tipo de árbol denominado *FP-tree*, [HPY00]) o Eclat (calcula el soporte realizando intersecciones de conjuntos que aborta en cuanto puede asegurar que no se alcanzará el soporte mínimo, [ZPOL97]). Una comparación entre algunos de los algoritmos se puede encontrar en [HGN00].

4.1.1.2. Extracción de reglas a múltiples niveles

En algunos casos, los datos pueden presentar una jerarquía de conceptos. Estas jerarquía pueden surgir mediante la combinación de atributos en las bases de datos (p.e. tener una jerarquía de las fechas agrupándolas en meses y años) o dadas de forma explícita por expertos para el proceso. Las relaciones entre elementos es posible que no aparezcan a los niveles más bajos de las jerarquía y sí en niveles más altos. La extracción de reglas de asociación a múltiples niveles o diferentes niveles conceptuales se ha estudiado.

Dado que en los DataCubos también trabajamos con jerarquías de conceptos, estas técnicas resultan interesantes a la hora de presentar los problemas de trabajar sobre múltiples niveles.

Srikant y Agrawall ([SA95]) propusieron tres algoritmos para la extracción de reglas múltiples niveles (una extensión directa del Apriori considerando todos los elementos, Cumulate y EstMerge). Los mecanismos propuestos, menos el básico, realizaban una generación de reglas a todos los niveles y la aceptación de las reglas se hace utilizando una *medida de interés* que considera si existe una regla

a un nivel conceptual más alto que la incluya y esta no aporta más conocimiento. Veamos cómo definen esta medida: supongamos una regla $X \rightarrow Y$, siendo $Z = X \bigcup Y = \{z_1, ..., z_n\}$ y un ancestro (regla definida a menor nivel de detalle que la incluya) $\widehat{X} \rightarrow \widehat{Y}$, siendo $\widehat{Z} = \widehat{X} \bigcup \widehat{Y} = \{\widehat{z}_1, ..., \widehat{z}_n\}$ donde $\widehat{z}_i$ es un ancestro de z_i. La regla $X \rightarrow Y$ se considera interesante respecto a $\widehat{X} \rightarrow \widehat{Y}$ si su soporte y confianza es R veces superior a los valores esperados según este ancestro, siendo R un valor dado por el usuario. Estos valores esperados se calculan como

$$E_{\widehat{Z}}[Pr(Z)] = \frac{Pr(z_1)}{Pr(\widehat{z}_1)} \times ... \times \frac{Pr(z_n)}{Pr(\widehat{z}_n)} \times Pr(\widehat{Z}) \tag{4.1}$$

para el soporte esperado y, siendo $Y = \{y_1, ..., y_n\}$ e $\widehat{Y} = \{\widehat{y}_1, ..., \widehat{y}_n\}$ donde $\widehat{y}_i$ es un ancestro de y_i, se calcula la confianza esperada como

$$E_{\widehat{X} \rightarrow \widehat{Y}}[Pr(Y|Z)] = \frac{Pr(y_1)}{Pr(\widehat{y}_1)} \times ... \times \frac{Pr(y_n)}{Pr(\widehat{y}_n)} \times Pr(\widehat{Y}|\widehat{X}) \tag{4.2}$$

Las reglas que satisfazcan esta condición son denominadas $R - interesantes$.

Utilizan un único umbral para el soporte de los conjuntos frecuentes independientemente del nivel al que se definan.

Han y Fu ([HF95]) proponen varios métodos para obtener reglas a múltiples niveles. Realizan una búsqueda de conjuntos de elementos frecuentes descendiendo en la jerarquía (*top-down*). De esta forma, comienzan generando conjuntos de elementos frecuentes al nivel más alto. Una vez obtenidos, buscan conjuntos frecuentes a niveles más bajos si están agrupados por conjuntos frecuentes a niveles más altos. De esta forma, un conjunto frecuente se define como aquel cuyo soporte es superior a un umbral y todos sus ancestros son también frecuentes.

Otra diferencia con la propuesta anterior es que consideran diferentes umbrales de soporte para cada nivel conceptual. Estos métodos trabajan con conjuntos de elementos que están todos al mismo nivel conceptual.

Shen y Shen ([SS98]) proponen un algoritmo para la generación de reglas a todos los niveles de abstracción de una jerarquía asociada a una base de datos.

Considera que un conjunto de elementos pueda estar definida a diferentes niveles de abstracción y utilizar varias taxonomías. Yen ([Yen00]) propone otro método basado en la construcción de un grafo que describe las asociaciones entre los elementos a cualquier nivel.

Este método sólo realiza un lectura de la base de datos para construir el grafo, y posteriormente utiliza éste para obtener los conjuntos frecuentes.

Lui y Chung ([LC00]) propusieron un algoritmo con una filosofía de generalización (*bottom-up*) en lugar de especialización (*top-down*) a la hora de calcular los conjuntos de elementos frecuentes. De esta manera comienza trabajando con conjuntos al más bajo nivel. Si un conjunto no tiene un soporte superior al umbral lo que se hace es generalizarlo. De esta forma, se obtiene un nuevo conjunto de elementos resultado de esta generalización. Estos conjuntos son considerados de nuevo y si su soporte supera el umbral pasarán a ser conjuntos frecuentes.

Para el umbral de soporte propone un mecanismo para el cálculo a diferentes niveles basado en el *grado de generalidad* de los elementos. Este grado se define según la jerarquía utilizada. Suponiendo una jerarquía, el grado de generalidad del un elemento será la relación entre el número elementos al nivel más bajo que agrupa, y el número de elementos total de este nivel. La fórmula para su cálculo sería

$$g(a) = \frac{\text{Número de elementos agrupados por } a}{\text{Número de elementos del nivel más bajo}} \tag{4.3}$$

Para un conjunto de elementos, el grado de generalidad se define como el mínimo de los grados de los elementos que lo componen. Usando este grado, se calcula el umbral para cada conjunto de elementos generalizados a considerar.

Recientemente, las reglas de asociación han sido extendidas para ser utilizadas sobre bases de transacciones difusas ([DMSV03]). Hong *et al.* ([HLC03]) proponen un algoritmo para obtener reglas a múltiples niveles incorporando información cuantitativa mediante etiquetas lingüísticas.

En general, los procesos de extracción de reglas de asociación tiene el proble-

ma de que se suelen obtener un número alto que puede hacer complicado su interpretación. Si se realiza a múltiples niveles, el número de reglas generadas puede ser aún mayor. En la mayoría de las propuestas anteriores se incluyen mecanismo para reducir el número de reglas obtenidas. En algunos casos se hace mediante la selección de los conjuntos frecuentes a considerar ([SA95, HF95, LC00, SS98]) o mediante mecanismos de eliminación de reglas redundantes (reglas que están cubiertas por otras, [SA95]).

Otro enfoque seguido viene de la mano de la *inducción orientada por atributos*[1] o AOI. En algunos casos puede ser interesante trabajar con datos a mayor nivel de detalle del representado. La AOI lo que propone es realizar una generalización de los datos. Para ello se utilizan técnicas como la eliminación de atributos, utilizar niveles superiores en la jerarquía de conceptos, etc. Estos datos generalizados se pueden utilizar como punto de partida para la obtención de relaciones entre los elementos.

Utilizando este esquema, Han *et al.* ([HCC93]) propusieron un mecanismo de extracción de reglas. Lo que propone es realizar una generalización de las transacciones a contemplar de forma que se obtenga un número menor o igual a un número establecido como umbral. Para ello, realiza una generalización de elementos mediante el uso de una jerarquía de conceptos. Una vez generalizado, obtiene reglas partiendo de este nuevo conjunto. Este mismo enfoque lo aplicaron Muyeba y Keane ([MK00]) en su algoritmo COMPARE.

4.1.2. OLAP Mining

Con el uso cada vez más extendido de los data warehouses, se hace necesario el desarrollo de técnicas de minería de datos que trabajan sobre los modelos multidimensionales que se utilizan. Según Han ([Han97]) las razones para realizarlo

[1] attribute-oriented induction

serían tres:

- Las herramientas de extracción de datos necesitan trabajar sobre datos integrados, consistentes y sobre los que previamente se ha aplicado una limpieza ([FPSSU96]). Un Data Warehouse se construye realizando este tipo de preprocesamiento sobre los datos. Por esto, sería una buena fuente para obtener los datos en el análisis.

- Los usuarios habitualmente requieren moverse sobre los datos de la base de datos, seleccionar porciones de los datos y analizarlos a diferentes niveles de abstracción. Los sistemas OLAP permiten realizar este movimiento de forma flexible. La integración de técnicas de minería de datos dará mecanismos para ver las relaciones entre los datos manteniendo la flexibilidad de cara al usuario.

- Para los usuario a veces es difícil predecir previamente que tipo de conocimiento quieren obtener. Integrando OLAP con diferentes métodos de minería de datos, se da la libertad al usuario de seleccionar diferentes técnicas de minería de datos.

En los últimos años se han propuesto técnicas de minería de datos adaptadas para ser aplicadas sobre datacubos. Algunos ejemplos pueden encontrarse en [HCC+97, NH94, EKS97, HCC98]. Como en el caso de la sección anterior nos centraremos en las técnicas propuestas de extracción de reglas de asociación.

Una primera aproximación se puede encontrar en [KHC97]. En este trabajo los autores proponen la extracción de reglas utilizando una estructura de datacubos. El modelo multidimensional presentado es simple: contempla que cada atributo sea una dimensión y el datacubo tenga pre-calculados el resultado de todas las posibles combinaciones de atributos. En cada una almacenaría el número de

transacciones que hay para esa combinación. No define jerarquías sobre las dimensiones ni mecanismos de agrupamiento. De esta manera, obtener el soporte de un conjunto de elementos se traduce en buscarla en el datacubo. Al proceso se le añaden meta-reglas que reduzcan el espacio de búsqueda total a aquel que resulte interesante.

En [Zhu98], Zhu propone la extracción de reglas de asociación sobre datacubos. Diferencia entre tres tipos de asociaciones posibles:

- Asociaciones intra-dimensionales. En este caso se buscan asociaciones entre los elementos de una dimensión (*item dimension*), considerando otra dimensión para obtener el soporte de los elementos (*transaction dimension*).

- Asociaciones inter-dimensionales. Las relaciones se buscan entre elementos de diferentes dimensiones.

- Asociaciones híbridas. En este caso las relaciones que se buscan son entre elementos de una misma dimensión y entre elementos de diferentes dimensiones. Se trata de la unión de asociaciones intra-dimensionales e inter-dimensionales. Lo que realizan es buscar conjuntos de elementos frecuentes intra-dimensionales, e inter-dimensionales y fusionarlos.

Para el cálculo de los conjuntos de elementos frecuentes proponen una adaptación del algoritmo Apriori ([SA95]) visto en la sección anterior.

Otras propuestas que se pueden considerar que trabajan sobre bases multidimensionales son aquellas orientadas a extraer lo que se han denominado reglas inter-transaccionales ([LFH00, TLHF03]). En estos casos se consideran los datos junto a los atributos que los contextualizan (p.e. la fecha en que se dieron, el lugar donde ocurren, etc.) que se denominan *dimensiones*.

Normalmente los procesos de extracción de reglas utilizan las relaciones entre elementos que se encuentran en las mismas transacciones (se dan a la vez). En

ocasiones es posible que estos elementos no se den juntos sino que tengan una relación respecto a las dimensiones que los caracterizan. De esta forma se buscarían reglas del estilo *Si las acciones de la compañía A suben el día 1, entonces las acciones de la compañía B bajarán el día 2, pero subirán el 4.*

Sin embargo, aunque definen dimensiones, no pueden considerarse dentro de las técnicas de OLAM dado que no trabajan sobre datacubos.

También se han propuesto técnicas de minería de datos sobre datacubos precisos utilizando técnicas difusas (p.e. árboles de decisión difusos sobre datacubos precisos, [LBMD01, LBMD$^+$00]) y técnicas difusas sobre datacubos difusos ([Lau03a, AK03, KA05]). En este enfoque, la extracción de reglas de asociación sobre datacubos difusos ha sido estudiada por Kaya y Alhajj ([AK03, KA05]). El modelo multidimensional difuso propuesto por los autores se puede consultar en la sección 2.2.5.4. Como ya comentamos entonces, se trata de un modelo con un tratamiento simple de la imprecisión.

La propuesta que realizan consideran la generación de reglas a todos los niveles definidos en el datacubo. Para cada nivel definen un umbral mínimo de soporte para aceptar los elementos de ese nivel. Cuando los conjuntos están formados por elementos de múltiples niveles, el umbral de soporte se considera el mínimo de los asociados a los elementos individuales.

El algoritmo propuesto es una adaptación del algoritmo Apriori previamente comentado. Para considerar una regla como interesante establecen un umbral de confianza y una condición extra para evitar dependencias negativa debido a elementos muy frecuentes. Para detectar estas situaciones proponen la medida siguiente

$$I(X \rightarrow Y) = \frac{Soporte(X \bigcup Y)}{Soporte(X) \times Soporte(Y)}, \tag{4.4}$$

aceptando una regla si $I(X \rightarrow Y) \geq 1$.

Para utilizar esta propuesta se ha de definir un umbral mínimo de soporte para cada nivel considerado. Esto puede implicar que el usuario debe tener un

conocimiento alto del DataCubo como para poder establecer este umbral para considerar frecuente el conjunto de elementos. Además, considera todas las combinaciones de elementos entre dimensiones, de forma que pueden aparecer muchas reglas redundantes y complicar el resultado de cara a el usuario final.

4.2. COGARE: Extracción de reglas guiada por complejidad

En la sección anterior hemos comentado diferentes propuestas para la extracción de reglas de asociación sobre DataCubos tanto precisos como difusos. Nosotros seguiremos un enfoque similar proponiendo un mecanismo de extracción de reglas de asociación sobre el modelo multidimensional difuso presentado en el capítulo anterior.

Los mecanismos anteriormente comentados suelen buscar la extracción del máximo conocimiento existente sobre los datacubos. Por esto, suelen obtener un conjunto muy grande de reglas. Algunos de ellos proponen mecanismos para la eliminación de reglas redundantes, con lo que consiguen reducir en parte el conjunto resultante.

El algoritmo COGARE (*COmplexity Guided Association Rule Extraction*) lo que busca es obtener un resultado que sea la más entendible posible de cara al usuario. Por ello, lo que guiará el proceso será la complejidad frente a la calidad del conjunto. Si permitimos que la calidad sea muy baja, aunque el resultado sea simple, tampoco será útil. Por esto, aunque buscaremos reducir la complejidad, controlaremos la calidad del resultado.

Antes de presentar el algoritmo, necesitaremos introducir las medidas que utilizaremos para controlar la complejidad y la calidad de las reglas.

4.2.1. Medidas de complejidad

Nuestro objetivo es reducir la complejidad del conjunto de reglas obtenido de forma que sea lo más entendible posible de cara al usuario que haya de interpretarlas. Por esto, lo primero que debemos hacer es definir cómo se va a medir la complejidad para poder manejarla en el proceso de extracción.

Siguiendo un enfoque similar al propuesto en [ABP04], identificamos los factor que influyen en la complejidad de un conjunto de reglas de car al usuario. Consideraremos dos factores:

- Número de reglas: cuanto mayor sea el número de reglas que obtengamos, más complicado será de interpretar de cara al usuario. En la siguiente sección presentaremos una medida de complejidad de un conjunto de reglas según su cardinalidad.

- Complejidad de los elementos de las reglas: cuantos más concretos sean los valores que compongan las reglas más información darán pero más complicado será de entender de cara al usuario. Por esto, nos interesará que las reglas utilicen elementos de niveles altos. Para medir el grado de complejidad de un valor introduciremos el concepto de abstracción de un elemento y los extenderemos a una regla y a un conjunto de reglas.

Una vez presentemos medidas para estos factores de complejidad, introduciremos un mecanismo simple de combinarlos.

4.2.1.1. Número de reglas

Como hemos comentado, cuanto mayor sea el número de reglas que obtengamos mayor será la dificultad por parte del usuario para interpretarlo. Para nosotros, el comportamiento de una medida que mida la complejidad según el tamaño del conjunto tendrá el siguiente comportamiento.

Definición 4.2 *Sea C_{NR} una función definida como*

$$C_{NR} : \mathbb{N} \rightarrow [0, 1] \tag{4.5}$$

será una medida de complejidad por número de reglas *si para todo x e y tal que $x > y$ se cumple $C_{NR}(x) \geq C_{NR}(y)$ (monótona creciente).*

Si al aplicar una medida de complejidad de este tipo sobre un conjunto de reglas se obtiene un valor 1, indicará que alcanza la máxima complejidad (será un conjunto con una cardinalidad muy alta). Un valor cercano a 0 implicará que el conjunto de reglas será muy reducido y, por lo tanto, por parte de un usuario será fácil de entender.

Cualquier función que cumpla la definición anterior podrá utilizarse como medida de complejidad en el posterior algoritmo. Una opción será establecer un umbral de forma que cuando el número de reglas sea mayor al umbral, estableceremos una complejidad 1, y si es igual o menor, una complejidad 0. Sin embargo, la complejidad del conjunto de reglas depende del dominio de las dimensiones del datacubo. Obtener 100 reglas puede parecer un conjunto complejo si buscamos relaciones entre 10 elementos. En el caso de que el número de elementos fuera 1000, obtener 100 reglas se puede considerar que no es tan complejo dado el número de relaciones posibles. Por esto, la medida de complejidad queremos que se adapte al tamaño del dominio y mida la complejidad dependiendo del tamaño del problema.

Siguiendo esta idea, proponemos la siguiente medida.

Definición 4.3 *Sea N el número elementos en el dominio de las dimensiones de un datacubo C. El valor de complejidad por tamaño de un conjunto de reglas sobre el datacubo C sería una función $C_{NR} : \mathbb{N} \rightarrow [0, 1]$ donde para un conjunto*

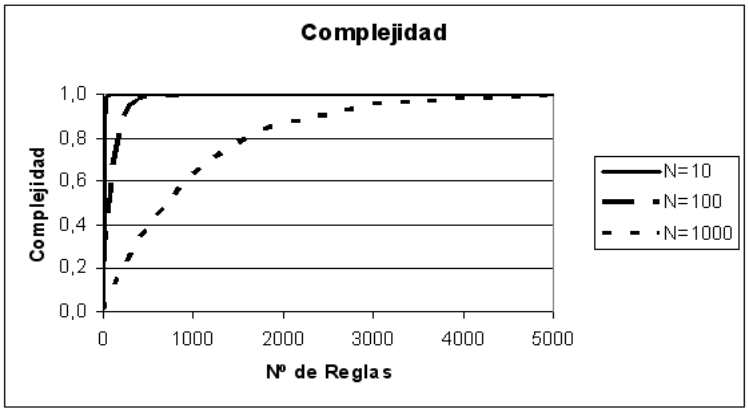


Figura 4.1: Evolución de la complejidad para diferentes tamaños de problemas

C_R el valor sería

$$C_{NR}(C_R) = \begin{cases} 0 & \textit{Si } |C_R| = 0 \\ 1 - e^{-\frac{|CR|}{N}} & \textit{en otro caso} \end{cases} \tag{4.6}$$

Si el conjunto de reglas estuviera formado por 0 reglas, el caso más simple, la medida de complejidad nos daría un valor 0. Conforme el número de reglas crezca, la complejidad lo hará, pero dependiendo del número de elementos considerados. En la Figura 4.1 puede verse como evoluciona la medida conforme aumenta el número de reglas para diferentes tamaños del problema.

Con estas medidas tenemos una manera de medir la complejidad de los resultados obtenidos de forma que se adapte al tamaño del problema.

4.2.1.2. Abstracción

Otro de los factores que influye en la complejidad de las reglas de cara al usuario es el nivel detalle al que estén definidos los elementos que la forman.

Para medir esta complejidad utilizaremos una medida de las abstracción de los elementos. Posteriormente la utilizaremos para medir la abstracción del conjunto de reglas.

Definición 4.4 *Sea D una dimensión. Una función A definida como*

$$A : dom(D) \rightarrow [0, 1] \tag{4.7}$$

será una función de abstracción *si cumple las siguientes propiedades*

1. *Si $c_i \in l_i \wedge c_j \in l_j \wedge l_j \in P_{l_i}$ entonces $A(c_j) \geq A(c_i)$ (la abstracción aumentará conforme subimos en la jerarquía).*
2. *Si $c_i \in l_\perp$ entonces $A(c_i) = 0$ (todos los valores del nivel base tienen abstracción 0).*
3. *Si para un $c_i \in l_i$ se da $\forall c_\perp \in l_\perp \eta_{i\perp}(c_i, c_\perp) = 1$ entonces $A(c_i) = 1$ (si un valor engloba a todos los definidos en el nivel base, su abstracción será 1).*

De la definición podemos ver que estas funciones son crecientes conforme nos elevamos en la jerarquía. Representa que valores más altos en dimensión serán menos concretos que los que se encuentren en los niveles bajos. De la definición podemos ver que forzamos a que el nivel base de la dimensión ($l_\perp$) tendrá, para todos los valores que engloba, una abstracción 0. Parece lógico dado que es el nivel más concreto que manejamos. La última condición establece que si un elemento agrupa a todos los elementos del nivel base con grado 1 (están totalmente relacionados), ese valor será el más abstracto que se pueda utilizar. Por eso, la abstracción de ese elemento será siempre 1. Normalmente ese elemento pertenecerá al nivel nivel $l_\top$.

Veamos algunos ejemplos de funciones para medir la abstracción. Una primera aproximación sería considerar cada nivel como una medida de la abstracción de

cada uno de sus elementos. De esta manera todos los elementos de un nivel tendrían la misma abstracción. En este caso, podríamos definir la abstracción del nivel según el número de elementos que utilice para dividir el dominio, es decir, el número de elementos que se definen a este nivel ($|l_i|$). Cuanto mayor sea el número de elementos, mayor sea la abstracción. De tal forma que presenta una relación inversa entre abstracción y valores en el nivel. Si queremos que el nivel $l_\perp$ tenga valor 0 (como fuerza la definición de función de abstracción) una posible definición sería la siguiente.

Definición 4.5 *A^1 es una función de abstracción definida como $A^1 : dom(D) \rightarrow [0,1]$ cuyo valor para $c_i \in l_i$ será:*

$$A^1(c_i) = \frac{\frac{1}{|l_i|} - \frac{1}{|l_\perp|}}{1 - \frac{1}{|l_\perp|}} \tag{4.8}$$

Como puede verse, en el caso de que $c_i \in l_\perp$, el valor de abstracción será 0. Si el nivel l_i que se considera sólo presenta un único valor, la función considera que se trata del mayor nivel de abstracción posible, por lo que le asigna un valor 1.

Esta función es bastante simple de calcular. Sin embargo, si el número de valores agrupados en diferentes etiquetas de un nivel es muy variable, esta función puede ser contraintutiva. Veamos un ejemplo simple: supongamos que tenemos un nivel que agrupa las edades en si son *mayores de edad* o no. Este nivel tendría dos etiquetas: *No* y *Sí*. En la primera de ellas, en España tendríamos los valores $\{0,...,17\}$, en total serían 18 valores. Por contra, la etiqueta *No* agruparía los valores $\{18,...,100\}$, suponiendo como edad máxima 100. En este valor se contemplarían 83 valores distintos. Si utilizamos la función anterior ambas etiquetas tendrían la misma abstracción cuando la cardinalidad de los conjuntos tiene una relación superior a 4:1. Estaríamos estableciendo niveles de abstracción asociados a cada nivel. Por esto, también podríamos definir una función de abstracción

que contemplara cada valor de forma independiente. La función de *generalidad* utilizada por Lui y Chung ([LC00]) representaría este comportamiento. Su definición adaptada a nuestra estructura multidimensional sería la siguiente

Definición 4.6 *A^2 es una función de abstracción definida como $A^2 : dim(D) \rightarrow [0,1]$ donde para un $c_i \in l_i$ su valor será*

$$A^2(c_i) = \frac{V_{l_i}}{|l_\perp|} \tag{4.9}$$

donde

$$V_{l_i}(c_i) = \begin{cases} \sum\limits_{\forall c_\perp \in l_\perp} \eta_{i\perp}(c_i, c_\perp) & \textit{si } l_i \neq l_\perp \\ 0 & \textit{en otro caso} \end{cases} \tag{4.10}$$

Como puede verse, se utiliza el cardinal difuso simple. Si c_i pertenece al nivel base, su abstracción será 0. Conforme subamos en la jerarquía, las abstracción irá creciendo. Si el elemento agrupa a todos los elementos definidos en el nivel base, por la definición vemos que el valor de su abstracción será 1.

Una vez tenemos la abstracción de cada elemento, necesitamos extenderlo para calcular la abstracción de una regla. Parece lógico pensar que la abstracción vendrá dada por el conjunto de elementos o items que la forman. De esta forma, la abstracción de una regla la obtendremos como la media de la abstracción de cada uno de sus elementos.

Definición 4.7 *Sea R una regla formada por los elementos $I_1, ..., I_N$ y A una función de abstracción. La abstracción asociada a la regla será*

$$A_R = \frac{\sum\limits_{i=1}^{N} A(I_i)}{N} \tag{4.11}$$

Si una regla tuviera una abstracción 0, esto implicaría que todos los elementos que la forman pertenecen a los niveles básicos de sus correspondientes dimensiones. Un valor muy alto, significaría que el caso es el contrario: los valores pertenecen a niveles altos en las jerarquías. Cuanto mayor sea la abstracción de la regla, significa que se emplean conceptos con menor detalle y, supuestamente más fáciles de entender de cara al usuario..

Veamos un ejemplo: si tenemos una regla de la forma

SI [Paciente tiene 23 años] Y [duración es 1 hora] ENTONCES [Gravedad es baja]

esta tendrá una abstracción más bien baja. Si por el contrario presentáramos una regla de la forma

SI [Paciente es Joven] Y [duración en baja] ENTONCES [Gravedad es baja]

Esta regla tendrá una abstracción más alta. De cara al usuario será más entendible dado que se está utilizando información con menor nivel de detalle. Sin embargo, un nivel muy alto de abstracción puede resultar en pérdida de información. Supongamos que tenemos una regla definida con valores del nivel más alto de las dimensiones. Tendría la forma

SI [Todos los pacientes] Y [Todas las duraciones] ENTONCES [Todas gravedades]

y su abstracción tendría un valor 1. Por el contrario, esta regla no da ninguna información dado que incluye a todos los casos posibles.

Una vez definida la abstracción de una regla, debemos obtener una medida de la abstracción aplicada a un conjunto de reglas. Intuitivamente esta dependerá de la abstracción de los reglas que la forman. Lo que haremos será agregar estas abstracciones para obtener un índice global mediante la aplicación de una media. En

este cálculo no todas las reglas tendrán el mismo peso dado que algunas cubrirán más ejemplos que otras de tal forma que el nivel de abstracción representará un mayor conjunto. De esta forma, realizaremos una media ponderada por el soporte de cada una de las reglas.

Definición 4.8 *Sea CR un conjunto de reglas $R_1, ..., R_N$ con soportes asociados $sop(R_1), ..., sop(R_N)$ y A una función de abstracción. La abstracción del conjunto CR será*

$$A_{CR} = \frac{\sum_{i=1}^{N} A_{R_i} \times sop(R_i)}{\sum_{i=1}^{N} sop(R_i)} \tag{4.12}$$

De esta forma, un conjunto de reglas en la que todos sus elementos se definen al nivel más bajo de cada dimensión, su abstracción asociada será 0. Cuanto más alto sea su abstracción indicará que las reglas se definen utilizando valores de niveles más altos. Un valor de 1 implicaría que se han definido al máximo nivel de abstracción posible.

Cuanto mayor sea la abstracción menor será su complejidad para el usuario. Así pues, dada una función de abstracción A, la complejidad asociada a un conjunto de reglas CR será

$$C_A(CR) = 1 - A(CR) \tag{4.13}$$

4.2.1.3. Medida de complejidad global

Una vez que hemos establecido los mecanismos para medir los dos factores para medir la complejidad de un conjunto de reglas, veamos como combinarlas para obtener un medida única. Para ello, dada una función de complejidad para el número de reglas (C_{NR}) y otra para la abstracción C_A, proponemos combinar los

	Perteneciente a C	No perteneciente a C	
Cubiertos por R	f_{rc}	$f_{r\overline{c}}$	f_r
No cubiertos por R	$f_{\overline{r}c}$	$f_{\overline{r}\overline{c}}$	$f_{\overline{r}}$
	f_c	$f_{\overline{c}}$	1

Cuadro 4.1: Tabla de contingencias con valores relativos

dos factores mediante una media ponderada de la forma

$$C_{global}(CR) = \alpha \times C_{NR}(CR) + (1-\alpha)(1 - C_A(CR)) \tag{4.14}$$

4.2.2. Medidas de calidad de reglas

Dado un conjunto de reglas, es interesante tener una medida de la calidad del mismo. La mayor parte de ellas nace dentro del modelado de sistemas mediante reglas. A continuación presentaremos algunas de las medidas habitualmente utilizadas para la calidad de reglas. Posteriormente, dada una medida de calidad para las reglas daremos la expresión para calcular la calidad de un conjunto de reglas.

4.2.2.1. Medidas de calidad clásicas

En esta sección presentaremos muy brevemente algunas de las principales medidas de calidad utilizadas. Para mayor detalle consultar [AC04, DF97, Ped04, TK00]. Muchas medidas de calidad de reglas están basadas en el análisis de la relación entre la regla $R : X \rightarrow C$ y la clase o consecuente de la misma (C) basándose en la tabla de contingencia (Tabla 4.1).

Cobertura y consistencia: La consistencia mide cómo de específica es una regla. La consistencia será alta cuanto mejor cubra la regla la clase que considera. Alcanza el máximo cuando no se dan falsos positivos. La fórmula de esta

medida es

$$Q_{cons}(R) = \frac{f_{rc}}{f_r} = \frac{sop(R)}{sop(A)} \tag{4.15}$$

y toma valores en el intervalo [0,1].

En el ámbito de la extracción de reglas de asociación, a esta medida se conoce mejor como *confianza*. Ha sido utilizada ampliamente para obtener una medida de la calidad de las reglas y servir de poda a la hora de generar reglas.

La cobertura o completitud mide cuanto del dominio del consecuente es cubierto por una regla. Alcanza el máximo (valor 1) cuando la regla engloba a todos los ejemplos que satisfacen el consecuente. Su formulación es la siguiente

$$Q_{cob}(R) = \frac{f_{rc}}{f_r} = \frac{sop(R)}{sop(C)} \tag{4.16}$$

y como en el caso anterior, toma valores entre 0 y 1.

Estos dos indicadores miden dos aspectos importantes de la calidad de las reglas. Sin embargo, si sólo consideramos la cobertura tendremos que las reglas con buena cobertura no sólo englobarán ejemplos pertenecientes al consecuente. La consistencia por sí sola puede llegar a que las reglas sólo cubran unos pocos ejemplos del consecuente, por lo que es posible que se produzca un sobreajuste. En el caso de extracción de reglas de asociación este segundo problema no es tan importante dado que se intenta obtener información y no reproducir el comportamiento del sistema. Sin embargo, veremos algunas de las propuestas que combinan ambas medidas para obtener las ventajas de ambas.

Michalski ([Mic90]) propuso la suma ponderada de ambas medidas según la fórmula siguiente

$$Q_{SP}(R) = w_1 \times Q_{cons}(R) + w_2 \times Q_{cob}(R) \tag{4.17}$$

donde $w_1 = 0{,}5 + \frac{1}{4}Q_{cons}(R)$ y $w_2 = 0{,}5 - \frac{1}{4}Q_{cons}(R)$.

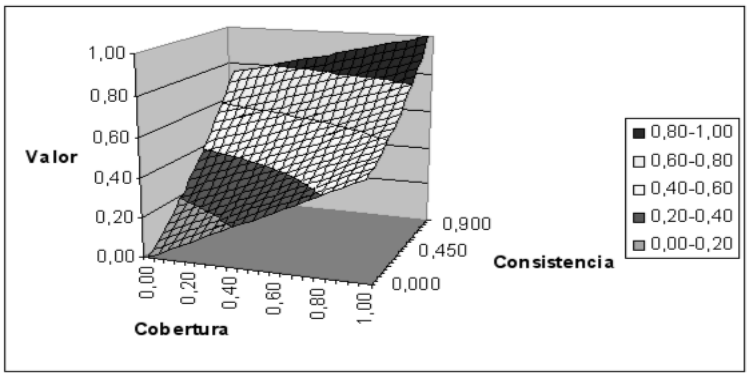


Figura 4.2: Comportamiento de la combinación de consistencia y cobertura propuesta por Michalski

Como puede verse, se utilizan pesos dependientes de la consistencia. De esta forma, cuanto mayor sea la consistencia mayor será la importancia de este factor en la suma (Figura 4.2). Otra propuesta ([BT90]) es la siguiente combinación

$$Q_P(R) = Q_{cons}(R) \times e^{Q_{cob}(R)-1} \tag{4.18}$$

En este caso también se da preferencia a la consistencia de la regla frente la cobertura (Figura 4.3) aunque en menor medida.

Factor de certeza: Esta medida fue propuesta por Shortliffe y Buchanan en 1975. Su formulación es la siguiente

$$Q_{FC}(R) = \begin{cases} \frac{Q_{cons}(R)-f_c}{1-f_c} & \text{si } Q_{cons}(R) > f_c \\ \frac{Q_{cons}(R)-f_c}{f_c} & \text{si } Q_{cons}(R) < f_c \\ 0 & \text{en otro caso} \end{cases} \tag{4.19}$$

Esta medida devuelve un valor en el intervalo [-1,1], de tal forma que cuanto más cercano sea el valor a 1 implica que la observación del antecedente conlleva

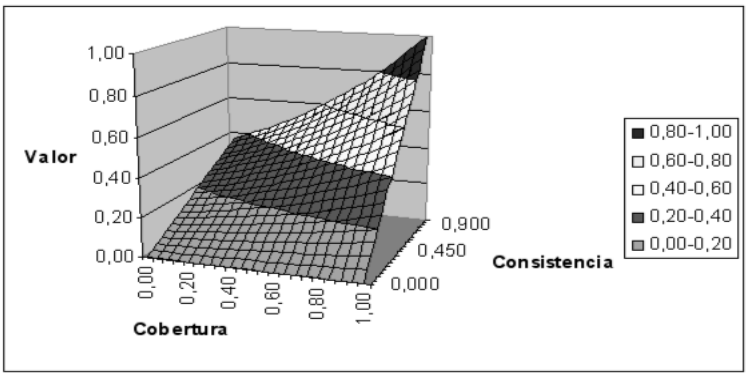


Figura 4.3: Comportamiento de la combinación de consistencia y cobertura propuesta por Brazdil y Torgo

el consecuente (relación directa). Cuanto menor sea el valor (más cercano -1) indica que la observación del antecedente implica la no ocurrencia del consecuente (relación inversa).

Esta medida ha sido utilizada como sustituta de la confianza en la extracción de reglas de asociación por algunos autores ([DMSV03]) debido a que presenta menos problemas que la primera.

Medidas de concordancia: Las medidas de concordancia consideran las diagonal principal de la tabla de consistencia. La calidad de Cohen calcula la diferencia entre la asociación real y la predicha a través de la diagonal. La fórmula correspondiente sería:

$$Q_{Cohen}(R) = \frac{Q_{cons}(R) - f_c}{\frac{1}{2}(1 + \frac{Q_{cons}(R)}{Q_{cob}(R)}) - f_c} \tag{4.20}$$

Devuelve un valor en el intervalo [-1,1], donde los valores negativos representan una relación inversa entre el antecedente y el consecuente.

El estadístico de Coleman mide la relación entre la primera columna y la primera fila de la tabla de consistencia. La fórmula sería:

$$Q_{Coleman}(R) = \frac{Q_{cons}(R) - f_c}{1 - f_c} \tag{4.21}$$

Devuelve también un valor en el intervalo [-1,1] con el mismo significado. Este indicador no tiene en cuenta la cobertura de una regla. Esto significa que no mide los falsos negativos que obtiene la regla.

Para mejorar el comportamiento de ambas medidas, y capturar la mejores características se ambas, se han propuesto ([Bru96]) dos medidas que combinan a las anteriores. Las medidas son:

$$Q_{C1}(R) = C_{Coleman}(R) \times \frac{2 + C_{Cohen}(R)}{3} \tag{4.22}$$

$$Q_{C2}(R) = C_{Coleman}(R) \times \frac{1 + Q_{cob}(R)}{2} \tag{4.23}$$

Estas medidas miden la concordancia y la asociación. Para ello utilizan la totalidad de la tabla de contingencia.

Medida de información: Basándose en la teoría de la información se puede definir la ganancia de información de una regla como:

$$Q_{GI}(R) = -log(f_c) + log\frac{f_{rc}}{f_r} \tag{4.24}$$

donde el logaritmo es en base 2. Sólo es válido si $Q_{cons}(R) \geq f_c$, por lo que no es muy adecuada. No incorpora medida de la completidud o cobertura. Este estadístico da la información que aporta la regla para clasificar correctamente los elementos pertenecientes a la clase (que cumplen el consecuente).

Medida de suficiencia lógica: La medida de la suficiencia lógica de la regla viene dada por la fórmula

$$Q_{SL}(R) = \frac{f_{rc}/f_c}{f_{r\bar{c}}/f_{\bar{c}}} \tag{4.25}$$

Si el valor obtenido es alto, esto significa que la observación del antecedente A, alienta la ocurrencia del consecuente. En el extremo, un valor de infinito implicaría que el antecedente es suficiente para asegurar C desde un punto de vista lógico.

Medida de discriminación: De la recuperación de la información probabilística se ha utilizado la discriminación. Este medida da una idea de la capacidad de una consulta para discriminar entre documentos relevantes y no relevantes. Si se considera una regla como una consulta, los ejemplos que cubre el antecedente y el consecuente serían los documentos relevantes, y los que no cumplan el consecuente, serían los no relevantes. Su formulación sería la siguiente:

$$Q_D(R) = log\frac{f_{rc}/f_{\overline{r}c}}{f_{r\bar{c}}/f_{\overline{rc}}} \tag{4.26}$$

Medidas empíricas: Algunas medidas propuestas no se basan en un análisis formal de la dependencia sino de resultados empíricos. No son fáciles de interpretar pero suelen presentar buenos resultados en los sistemas para los que se propusieron. Un ejemplo de estas medidas en la utilizada en un sistema de inducción de reglas. Su formulación es

$$Q_{IMAFO}(R) = (AC \times E_C) \times 10 \tag{4.27}$$

donde $AC = f_{rc} + f_{\overline{rc}}$ da una medida de la precisión de la regla, y $E_C = e^{\frac{f_{rc}}{f_c}-1}$ mide la cobertura.

En indicador devuelve un valor en el intervalo [0,10].

4.2.2.2. Calidad de un conjunto de reglas

La calidad del conjunto de reglas lo haremos en función de la calidad de cada una de las reglas. Para ello las ponderaremos según el tamaño del espacio que cubran, es decir, aquellas reglas que incluyan mayor número de hechos (soporte mayor) tendrán más peso que aquellas que cubran un espacio pequeño (soporte bajo). Así pues, dada la función para calcular la calidad de cada regla (Q_R), la calidad del conjunto será

$$Q_{CR} = \frac{\sum_{i=1}^{|CR|} Q_R(R_i) \times sop(R_i)}{\sum_{i=1}^{|CR|} sop(R_i)} \tag{4.28}$$

donde $sop(R)$ representa el soporte de la regla R.

4.2.3. Algoritmo

En esta sección presentaremos el algoritmo COGARE. Como ya hemos comentado, el objetivo de éste será obtener un conjunto de reglas que sea lo más entendible de cara a el usuario. Por ello, intentaremos reducir la complejidad (tanto por número como por abstracción) del conjunto obtenido. Para ello, el usuario establecerá un umbral máximo de complejidad del resultado y cuanta calidad está dispuesto a sacrificar para obtenerlo.

Las fases del algoritmo son:

1. En primer lugar el algoritmo extraerá reglas de asociación sobre el datacubo. Buscará relaciones a múltiples niveles entre elementos de diferentes dimensiones.
2. Tras obtener el conjunto de reglas, si este supera la complejidad establecida por el usuario, intentará reducirla generalizando las reglas utilizando para ello las jerarquías de las dimensiones (Figura 4.4).

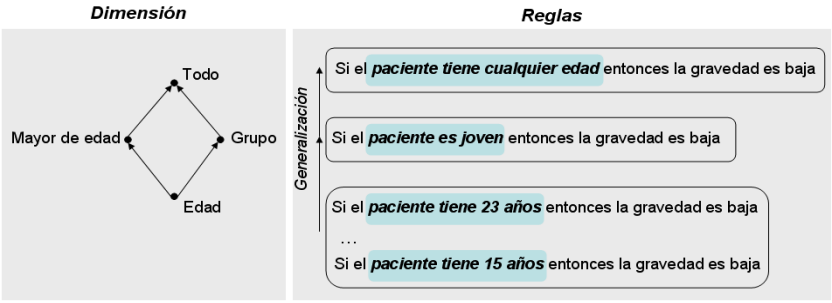


Figura 4.4: Generalización de reglas utilizando las dimensiones

Veamos detenidamente cada una de las fases comentadas.

4.2.3.1. Generación de reglas

En esta fase el algoritmo obtendrá las asociaciones entre los elementos de diferentes dimensiones y a múltiples niveles. Como hemos comentado al hablar de los algoritmos de extracción reglas, se compone de dos pasos:

- Obtener los conjuntos de elementos frecuentes.
- Generar las reglas utilizando los conjuntos obtenidos.

El primer paso buscará conjuntos de elementos frecuentes, es decir, aquellos que tengan un soporte superior o igual a un umbral. Dependiendo de los niveles a los que pertenezcan los elementos este umbral de aceptación será distinto. El usuario establecerá el valor para los niveles base ($l_\perp$). Basándose en este y en la abstracción del conjunto de elementos (I) se obtendrá el valor de umbral para aceptarlo. De esta forma, dada una función de abstracción A, para un conjunto I el umbral que deberá satisfacer el soporte será

$$umbral_I = umbral_{SOP} + (1 - umbral_{SOP}) \times A(I) \tag{4.29}$$

siendo $umbral_{SOP}$ el valor dado por el usuario. Para los conjuntos definidos a los niveles más bajos, dado que tendrán una abstracción 0, el umbral será el dado por el usuario ($umbral_{SOP}$). Conforme el conjunto esté formado por elementos de mayor nivel, mayor será el valor que tengan que satisfacer.

Para obtener los conjuntos seguirá un enfoque similar al utilizado por Lui y Chung ([LC00]). Se trata de una extensión del algoritmo Apriori. De esta forma, para obtener los conjuntos frecuentes de *k* elementos, construiremos los candidatos utilizando los conjuntos frecuentes de *k-1* elementos.

El proceso comenzará considerando como candidatos todos los conjuntos de un elementos construidos utilizando los valores de los niveles base de cada dimensión (siendo C el datacubo el conjunto de candidatos será $C_1 = \bigcup_{\forall D_i \in C} l_{i\perp}$).

Para cada conjunto de los candidatos, se calcula su umbral. Si este es mayor o igual a su umbral asociado, entonces se considera conjunto frecuente. Si no se acepta, los que se hace es generalizarlo y volver a considerarlo. El proceso de generalización (Figura 4.5) consiste es considerar los elementos de los niveles padres que incluyen al no aceptado. Estos conjuntos generalizados pasarían a considerarse nuevos candidatos y se repetiría el proceso (volviendo a generalizarlos si tampoco son frecuentes). Si no se puede generalizar, se deshecha el conjunto.

En la generalización no se consideran los elementos cuya abstracción sea 1 (agrupen a todos los valores del nivel base). Estos elementos producirían reglas de la forma

SI *[Paciente tiene cualquier edad]* entonces ...
SI ... entonces *[Paciente tiene cualquier edad]*

que no aportan información al usuario, y por ello, no se consideran.

Utilizando los conjuntos frecuentes de tamaño 1, se calculan los candidatos a frecuentes de tamaño 2. Para ello, se consideran las posibles parejas de valores tal que cada uno pertenece a una dimensión distinta. El proceso para aceptarlos con

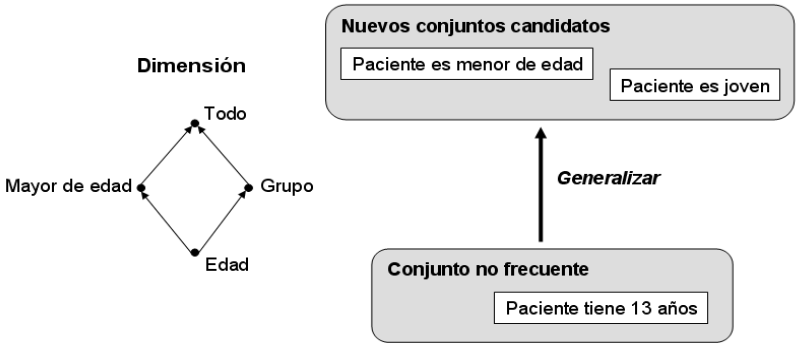


Figura 4.5: Generalización de conjuntos no aceptados

frecuentes es el mismo. En el caso de no ser aceptado, se generaliza de la misma manera pero considerando generalizaciones de cada elemento integrante de forma independiente.

De forma general, lo candidatos de tamaño k se obtiene como la unión de conjuntos frecuentes de tamaño $k-1$ tal que comparten $k-2$ elementos, los no comunes pertenecen a dimensiones distintas, y todos los subconjuntos de $k-1$ elementos son frecuentes (al igual que en el algoritmo Apriori, [AS94]). En el Cuadro 4.2 viene recogido el pseudocódigo del proceso.

Una vez obtenidos los conjuntos de elementos frecuentes construiremos reglas basándose en ellos. Utilizaremos el mismo algoritmo del Apriori, utilizando para podar un umbral de factor de certeza mínimo ($umbral_{CF}$). La utilización del factor de certeza en lugar la confianza se debe a los problemas que puede presentar este segundo indicador ([DMSV03]).

Algoritmo: $ObtenerConjutosFrecuentes(umbral_{SOP}, C)$

- Entradas:
  - $umbral_{SOP}$: Umbral de soporte para aceptar un itemset como frecuente
  - C: DataCubo
- Resultado:
  - Conjuntos elementos frecuentes (C_F)

1. $C_F = \emptyset$
2. $k = 0$
3. Hacer
   - 3.1. $k = k + 1$
   - 3.2. Si $k = 1$ entonces
     - 3.2.1. $C_1 = \bigcup_{\forall D_i \in C} l_{i\perp}$
   - 3.3. En otro caso
     - 3.3.1. $C_k =$ Generar candidatos utilizando C_{k-1}
   - 3.4. $C_{Fk} = \emptyset$
   - 3.5. Mientras $C_k \neq \emptyset$ hacer
     - 3.5.1. I= primer elemento de C_k
     - 3.5.2. $C_k = C_k - \{I\}$
     - 3.5.3. $sop_I =$ calcular soporte de I
     - 3.5.4. Si $sop_I \geq umbral_{SOP} + (1 - umbral_{SOP}) \times A(I)$ entonces
       - 3.5.4.1. $C_F = C_F \bigcup \{I\}$
     - 3.5.5. En otro caso
       - 3.5.5.1. $C_k = C_k \bigcup Generalizar(I)$
   - 3.6. Fin mientras
   - 3.7. Si $k \neq 1$ entonces
     - 3.7.1. $C_F = C_F \bigcup C_{Fk}$
4. Mientras $C_{Fk} \neq \emptyset \wedge k <$ Nº dimensiones
5. Devolver C_F

Cuadro 4.2: Algoritmo de cálculo de conjuntos frecuentes

4.2.3.2. Generalización

El conjunto de reglas encontrado en el paso anterior puede ser muy complejo de cara al usuario. El siguiente paso que haremos será intentar simplificarlo basándonos en las jerarquías definidas en las dimensiones y las relaciones entre sus elementos.

Para reducir la complejidad tendremos que abordar los dos factores que influyen: el número de reglas y la abstracción de los elementos. El algoritmo plantea la reducción aplicando un proceso de generalización de los términos utilizados. Este proceso incide de forma directa sobre el segundo factor, pero también sobre el primero. Veámoslo con un ejemplo: si tenemos las reglas siguientes

Si *[Paciente tiene 13 años]* entonces *[Gravedad es baja]*
Si *[Paciente tiene 20 años]* entonces *[Gravedad es baja]*

si generalizamos sustituyendo 13 años y 20 años por *Paciente es joven*, ambas reglas se fusionarían en una única con la forma

Si *[Paciente es joven]* entonces *[Gravedad es baja]*.

Por esto, reduciendo la complejidad debida a términos muy complejos es de esperar que el número de reglas también disminuya.

Como ya hemos comentado, este proceso de generalización se realizará siempre que el conjunto de reglas resultante del proceso anterior supere el umbral de complejidad dada por el usuario ($umbral_{Complejidad}$). El esquema del proceso de generalización es el recogido en la Figura 4.6.

Lo que se intentará será reducir la complejidad por debajo del umbral dado sin sacrificar para ello la calidad del conjunto de reglas. Si de esta manera no es posible, se intentará reducirla permitiendo una pérdida de calidad, cuyo nivel máximo dará el usuario. A continuación presentamos ambos pasos.

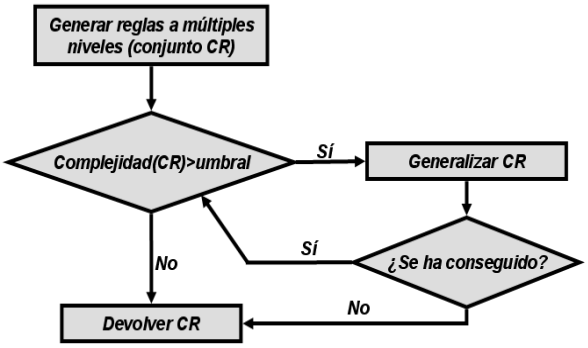


Figura 4.6: Proceso de reducción de complejidad

Sin pérdida de calidad Este primer enfoque fuerza a que se mantenga cuando menos la calidad del conjunto de reglas. Se tratará de un proceso iterativo que intentará generalizar el conjunto de reglas. Para ello, calculará los elementos que generalizan el conjunto actual. El siguiente paso será seleccionar un elemento de estos y aplicar la generalización. Si el conjunto resultado es menos complejo y mantiene la calidad, nos quedaremos con este conjunto y repetiremos el proceso hasta que el conjunto resultado tenga una complejidad menor o igual al umbral. En el caso de que nos se acepte el nuevo conjunto, probaremos con otro de los elementos que generalizan el conjunto y volveremos a aplicar el proceso.

A la hora de seleccionar los elementos para generalizar las reglas, aplicaremos una heurística para la ordenación de los mismos. Parece lógico pensar que aquellos elementos que generalicen un mayor número de reglas sean los primeros en ser probados. Si son susceptibles de generalizar un número alto de reglas, hace pensar que algunos de ellas puedan quedar reducidas a una misma. Además, la abstracción global del conjunto de reglas tenderá a verse aumentada. Sin embargo, es posible que estas generalizaciones no sean aceptadas (la calidad del conjunto

de reglas se vea reducida) y estos elementos vuelvan a ser considerados en futuras generalizaciones. Dado que están relacionados con un número alto de reglas, se tendrá que realizar un número considerable de cálculos (los nuevos soportes) por lo que será costoso en tiempo. Si reiteradamente estos elementos no son aceptados, estaremos consumiendo tiempo de forma inútil. Por esto, proponemos que conforme vayan utilizándose estos elementos y sean rechazados, su prioridad se vea reducida. Para ellos ordenaremos según un peso asociado a cada elemento dependiente del número de reglas que generaliza y el número de ocasiones en que ha sido utilizado sin éxito. Para el cálculo de dicho peso proponemos la siguiente fórmula:

$$Peso_I = \text{Nº reglas generalizadas} \times \frac{1}{(\text{Nº veces probado } I) \times \beta + 1} \qquad (4.30)$$

donde $\beta \in [0, \infty)$ controlaría la penalización de un elemento respecto al número de ocasiones probado sin éxito. Un valor 0 significaría que no se penalizaría (de forma que siempre se ordenarían por el número de reglas que podrían generalizar). A mayor valor de β mayor sería la penalización asociada a generalizaciones sin éxito.

Si tras probar con todos los valores, ninguno consigue reducir la complejidad hasta el umbral, el método terminará devolviendo el último conjunto de reglas aceptado. En el cuadro 4.3 viene recogido el algoritmo. Para aplicarlo se deberá escoger las funciones para calcular la complejidad y la calidad del conjunto de reglas.

Con pérdida de calidad En el caso de generalizar permitiendo pérdida de calidad el esquema será el mismo. El cambio radicará en el criterio de aceptación de un nuevo conjunto de reglas. En el caso anterior, forzábamos a que no existiera pérdida de calidad. En este enfoque tendremos en cuenta la calidad del conjunto así como la pérdida de calidad y de complejidad que se produzca. De tal forma,

Algoritmo: $Generalizar_{SP}(CR, umbral, \beta)$

- Entradas:
  - CR: Conjunto de reglas a generalizar
  - $umbral$: valor en [0,1] con el valor máximo de complejidad buscado
  - β: valor en $[0, +\infty]$ con la penalización para los elementos fallidos al generalizar.
- Resultado:
  - Conjunto de reglas generalizado

1. Mientras $Complejidad(CR) > umbral$ hacer
   - 1.1. G_{CR}=Conjunto de items que generalizan CR ordenados usando β
   - 1.2. Hacer
     - 1.2.1. Si $G_{CR} = \emptyset$ entonces
       - 1.2.1.1. Devolver CR
     - 1.2.2. I_G=primer elemento de G_R
     - 1.2.3. $G_{CR} = G_{CR} - \{I_G\}$
     - 1.2.4. CR'=generalizar CR utilizando I_G
   - 1.3. Mientras $(Q(CR') < Q(CR)) \vee (Complejidad(CR') > Complejidad(CR))$
   - 1.4. $CR = CR'$
2. Fin mientras
3. Devolver CR

Cuadro 4.3: Algoritmo de generalización sin pérdida de calidad

para aceptar un conjunto se tendrá que dar:

1. Para aceptar un conjunto forzaremos que la pérdida de calidad que se produzca merezca la pena respecto a la complejidad que se elimine. Esto se traducirá en que aceptaremos un nuevo conjunto si la pérdida de calidad es menor a un porcentaje ($\gamma \in [0, +\infty)$) de la complejidad que se pierde:

$$PerdidaCalidad < \gamma \times PerdidaComplejidad \tag{4.31}$$

2. En cualquier caso, aunque la reducción de complejidad sea alta, no aceptaremos un conjunto si su calidad se reduce por debajo de un umbral de calidad calculado como $\delta \times MejorCalidad$, donde $\delta \in [0, 1]$ es un parámetro dado por el usuario. Este nos da cuanta calidad está dispuesto a sacrificar en el proceso para reducir la complejidad.

De esta forma, el algoritmo propuesto sería el recogido en el cuadro 4.4.

4.2.3.3. Proceso completo

Una vez terminado de presentar las diferentes fases del algoritmo, comentaremos el proceso completo. Como puede verse, se trata de una búsqueda de reglas con un enfoque *bottom-up* o de generalización. De esta forma, partimos de los elementos definidos a los niveles más bajos de las dimensiones y posteriormente generalizamos para buscar las relaciones y reducir la complejidad. Hemos escogido este enfoque en contrate a la especialización (*top-down*) principalmente por eficiencia. Si realizamos una aproximación por especialización comenzaremos calculando el soporte de conjuntos de elementos definidos a niveles altos. Por esto, cabe esperar que el número de hechos con los que estén relacionados sea alto. Las agregaciones de valores cuando se considera imprecisión son bastante más costosas en estos casos dado que se contempla un manor número de

Algoritmo: $Generalizar_{CP}(CR, umbral, \beta, \delta, \gamma)$

- Entradas:
  - CR: Conjunto de reglas a generalizar
  - $umbral$: valor en [0,1] con el valor máximo de complejidad buscado
  - β: valor en $[0, +\infty]$ con la penalización para los elementos fallidos al generalizar.
  - δ: valor en [0,1] con el porcentaje de calidad a conservar
  - γ: valor en [0,1] con el porcentaje de pérdida de calidad respecto a la reducción de complejidad para aceptar un conjunto.
- Resultado:
  - Conjunto de reglas generalizado

1. $MejorCalidad = Calidad(CR)$
2. Mientras $Complejidad(CR) > umbral$ hacer
   - 2.1. G_{CR}=Conjunto de items que generalizan CR ordenados usando β
   - 2.2. Hacer
     - 2.2.1. Si $G_{CR} = \emptyset$ entonces
       - 2.2.1.1. Devolver CR
     - 2.2.2. I_G=primer elemento de G_R
     - 2.2.3. $G_{CR} = G_{CR} - \{I_G\}$
     - 2.2.4. CR'=generalizar CR utilizando I_G
     - 2.2.5. PérdidaCalidad = $(Q(CR) - Q(CR'))/Q(CR)$
     - 2.2.6. PérdidaComplejidad = $(Complejidad(CR) - Complejidad(CR'))/Complejidad(CR)$
   - 2.3. Mientras $(PerdidaCalidad > \gamma \times PerdidaComplejidad) \vee (Calidad(CR') < \delta \times MejorCalidad)$
   - 2.4. $CR = CR'$
3. Fin mientras
4. Devolver CR

Cuadro 4.4: Algoritmo de generalización con pérdida de calidad

elementos. En la generalización, partimos de elementos a niveles muy bajos, por lo que al calcular el soporte consideraremos pocos hechos, lo que se traduce un un menor tiempo de cómputo. Conforme generalicemos, el soporte será más costoso de calcular, pero en muy pocas ocasiones llegaremos a los niveles de partida del caso de la especialización.

La diferentes fases del algoritmo COGARE se enlazan según se recoge en el Cuadro 4.5.

El algoritmo aplicado sobre DataCubos difuso permite obtener información en forma de reglas más entendibles para el usuario. Esto se debe a dos factores:

- En las dimensiones definidas en los DataCubos se pueden utilizar conceptos más intuitivos para el usuario y modelarlos mediante la lógica difusa. Por esto, se pueden obtener reglas que utilicen estos conceptos por lo que serán más cercanas al espacio cognitivo del usuario.

- El algoritmo COGARE busca la reducción de la complejidad reduciendo el número de reglas y utilizando conceptos de más alto nivel. Por esto, de cara al usuario el conjunto obtenido será más reducido y utilizando conceptos de más alto nivel más fáciles de entender.

4.3. Experimentos

Tras presentar el algoritmo, vamos a probarlo utilizando para ello diferentes DataCubos construidos sobre datos reales. Comenzaremos presentando los parámetros usados y el mecanismo para el cálculo del soporte. Posteriormente comentaremos los resultados obtenidos y un análisis de tiempos.

Algoritmo: $COGARE(C, umbral_{Complejidad}, umbral_{SOP}, umbral_{CF}, \delta)$

- Entradas:
  - C: DataCubo sobre el que aplicar el algoritmo
  - $umbral_{Complejidad}$: valor en [0,1] con el valor máximo de complejidad buscado
  - $umbral_{SOP}$: valor en [0,1] con el valor de soporte para aceptar un conjunto de elementos
  - $umbral_{CF}$: valor en [0,1] con el valor de factor de certeza para aceptar una regla
  - β: valor en $[0, +\infty]$ penalización para los elementos fallidos al generalizar.
  - γ: valor en [0,1] con el porcentaje de reducción de calidad respecto a la mejora de complejidad para aceptar un conjunto en la generalización con pérdida.
  - δ: valor en [0,1] con el porcentaje de calidad a conservar
- Resultado:
  - Conjunto de reglas

1. C_F=$ObtenerConjuntosFrecuentes(umbral_{SOP}, C)$
2. $CR = GenerarReglas(C_F, umbral_{CF})$
3. Si $Complejidad(CR) > umbral_{Complejidad}$ entonces
   3.1. CR=$Generalizar_{SP}(CR, umbral_{Complejidad}, \beta)$
4. Si $Complejidad(CR) > umbral_{Complejidad}$ entonces
   4.1. CR=$Generalizar_{CP}(CR, umbral_{Complejidad}, \beta, \delta, \gamma)$
5. Devolver CR

Cuadro 4.5: Algoritmo COGARE

4.3.1. Parámetros de los procesos

Para la experimentación hemos utilizado tres dominios distintos: datos médicos ($C_{\text{Médico}}$), contables ($C_{\text{Contables}}$) y del censo (C_{Censo}). Los DataCubos que se han construido sobre ellos se pueden consultar en el capítulo 5. Veamos los valores escogidos para los parámetros necesarios:

- $umbral_{Complejidad}$: para estudiar la capacidad par reducir la complejidad, estableceremos un umbral de complejidad 0. De esta forma, el proceso intentará realizar la máxima reducción posible.

- $umbral_{SOP}$: en cada problema de forma empírica hemos establecido un umbral distinto. Los valores utilizamos son

  - $C_{\text{Médico}}$: umbral $0{,}01$.
  - $C_{\text{Contables}}$: umbral $0{,}01$.
  - C_{Censo}: umbral $0{,}15$.

- $umbral_{CF}$: en todos los casos se ha utilizado un umbral de factor de certeza de $0{,}2$ para aceptar una regla.

- β: dado lo costoso que resulta calcular los soportes de los conjuntos de elementos, aquellos elementos que generalicen un número alto de reglas implicará calcular un número altos de soportes. Por esto, si un elemento falla al generalizar le estableceremos una penalización alta. Este hecho explica la elección un valor $\beta = 10$, escogido de forma empírica.

- γ: utilizaremos un valor $\gamma = 1{,}2$, por lo que aceptaremos un nuevo conjunto de reglas si la reducción de la calidad es como mucho el 120 % de la reducción de la complejidad.

- δ: permitiremos que se siga reduciendo la complejidad mientras la calidad no baje por debajo del 60 % de la mejor obtenida ($\delta = 0{,}6$).

Para medir la complejidad del conjunto utilizaremos la medida por número de reglas C_{NR} (presentada en la Sección 4.2.1.1) y la medida de abstracción A^2 (Sección 4.2.1.2).

En el caso la calidad de las reglas utilizaremos todas las medidas clásicas que hemos presentado en la Sección 4.2.2: cobertura (Q_{cob}), consistencia o confianza (Q_{cons}), suma ponderada de las anteriores (Q_{SP}), producto de cobertura y consistencia (Q_P), factor de certeza (Q_{CF}), Cohen (Q_{Cohen}), Coleman ($Q_{Coleman}$), las combinaciones propuestas por Bruha (Q_{C1} y Q_{C2}), ganancia de información (Q_{GI}), suficiencia lógica (Q_{SL}), discriminación (Q_D), y la medida propuesta en IMAFO (Q_{IMAFO}) donde también consideramos la medida de precisión que utilizan (Q_{AC}).

Además de los DataCubos difusos, consideraremos una versión precisa de la misma (estableciendo a 1 todas las relaciones con valores mayores o iguales a $0{,}5$, y cero las restantes). De esta manera compararemos los resultados aplicados sobre el modelo difuso y preciso y veremos si el primero aporta ventajas en el proceso de extracción de reglas.

Así pues, tendremos 2 DataCubos por cada dominio y 14 ejecuciones (una por cada medida de calidad) sobre cada uno (84 experimentos en total).

4.3.2. Cálculo del soporte

A la hora de medir el soporte se nos presenta el problema de cómo obtener el número de transacciones involucradas. Si miramos las medidas de los DataCubos que vamos a utilizar vemos que se ha definido una (*número*) que en todos los casos nos da el número de pacientes, empresas o personas que comparten coordenadas. Una transacción será cada una de los registros de partida y no el número de hechos

del DataCubo. Por esto, utilizaremos este hecho (mejor dicho la suma de éste) para calcular el soporte de un conjunto de elementos. De esta manera, un hecho con un valor 4 en la medida *número* implicaría que ese hecho representa a cuatro transacciones para calcular el soporte.

En los DataCubos precisos, utilizaremos la suma habitual para calcular el soporte. Para los difusos vamos a utilizar la misma aproximación propuesta en [DMSV03]. En este trabajo se propone utilizar la evaluación de sentencias cuantificadas para obtener el soporte en el caso difuso. Una sentencia cuantificada es una expresión de la forma

"Q de F son G"

donde F y G son conjuntos difusos y Q es un cuantificador lingüístico relativo. Para evaluar la sentencia cuantificada se utiliza el cuantificador GD ([DSV99]).

Como cuantificador relativo se utiliza $Q(x) = x$ dado que es fácil comprobar que la conjunción del mecanismo de evaluación y el cuantificador se comportan de forma coherente en el caso preciso ([DMSV03]). La elección de éste enfoque se debe a dos razones:

- Es fácil de calcular a partir de consultas utilizando los operadores propuestos para las consultas sobre los DataCubos difusos.
- Es un esquema eficiente para su cálculo.

4.3.3. Resultados

En este punto vamos a presentar los resultados obtenidos de los experimentos propuestos. Mostraremos los resultados agrupados por dominios y por DataCubos (preciso y difuso). Mostraremos los resultados tanto finales como los obtenidos en los pasos intermedios (generación de reglas, generalización sin pérdida de calidad

Medida calidad	Nº reglas	Complejidad	Calidad
Q_{CF}	106	0,25041103	0,5812684
Q_{GI}	106	0,25041103	0,0651961
Q_{P}	106	0,25041103	0,3967705
Q_{SP}	106	0,25041103	0,7692262
Q_{SL}	106	0,25041103	0,6611954
Q_{D}	106	0,25041103	0,8286231
Q_{cob}	106	0,25041103	0,8668966
Q_{cons}	106	0,25041103	0,5110079
$Q_{Coleman}$	106	0,25041103	0,7906342
Q_{Cohen}	106	0,25041103	0,6613348
Q_{AC}	106	0,25041103	0,6420718
Q_{C1}	106	0,25041103	0,4430986
Q_{C2}	106	0,25041103	0,7775496
Q_{IMAFO}	106	0,25041103	0,4472607

Cuadro 4.6: Resultados tras la generación de reglas en $C_{\text{Médico}}$ preciso

y el resultado final tras la generalización permitiendo pérdida de calidad). Los resultados para cada dominio vienen recogidos en las siguientes tablas, a saber, para el dominio de datos médico las tablas 4.6 a 4.13, el dominio de datos contables las tablas 4.14 a 4.21, y para los datos censales las tablas 4.22 a 4.29.

Medida calidad	Nº reglas	Complejidad	Calidad
Q_{CF}	106	0,25041103	0,5812684
Q_{GI}	97	0,22071731	0,0941390
Q_{P}	97	0,17654178	0,4842291
Q_{SP}	99	0,22559968	0,7733217
Q_{SL}	104	0,24446377	0,6630120
Q_{D}	104	0,24503176	0,8288022
Q_{cob}	106	0,25041103	0,8668966
Q_{cons}	86	0,15374571	0,5662591
$Q_{Coleman}$	106	0,25041103	0,7906342
Q_{Cohen}	74	0,22700647	0,6671687
Q_{AC}	104	0,24503176	0,6434762
Q_{C1}	104	0,24503176	0,4446466
Q_{C2}	106	0,25041103	0,7775496
Q_{IMAFO}	99	0,22559968	0,4598711

Cuadro 4.7: Resultados tras la generalización sin pérdida en $C_{\text{Médico}}$ preciso

Medida calidad	Nº reglas	Complejidad	Calidad
Q_{CF}	66	0,16571520	0,4716935
Q_{GI}	86	0,20474544	0,0907575
Q_{P}	61	0,13881620	0,4794883
Q_{SP}	60	0,13674732	0,7663548
Q_{SL}	87	0,18449190	0,5617908
Q_{D}	60	0,13674732	0,7020550
Q_{cob}	59	0,13447234	0,8141326
Q_{cons}	61	0,13881620	0,5950424
$Q_{Coleman}$	59	0,13767093	0,6641629
Q_{Cohen}	60	0,13477111	0,5773560
Q_{AC}	60	0,13674732	0,5214880
Q_{C1}	87	0,18741933	0,3594554
Q_{C2}	59	0,13447234	0,6579800
Q_{IMAFO}	60	0,13674732	0,3801841

Cuadro 4.8: Resultados tras la generalización con pérdida en $C_{\text{Médico}}$ preciso

Medida calidad	Nº reglas	Complejidad	Calidad
Q_{CF}	37,74 %	33,82 %	-18,85 %
Q_{GI}	18,87 %	18,24 %	39,21 %
Q_{P}	42,45 %	44,56 %	20,85 %
Q_{SP}	43,40 %	45,39 %	-0,37 %
Q_{SL}	17,92 %	26,32 %	-15,03 %
Q_{D}	43,40 %	45,39 %	-15,27 %
Q_{cob}	44,34 %	46,30 %	-6,09 %
Q_{cons}	42,45 %	44,56 %	16,44 %
$Q_{Coleman}$	44,34 %	45,02 %	-16,00 %
Q_{Cohen}	43,40 %	46,18 %	-12,70 %
Q_{AC}	43,40 %	45,39 %	-18,78 %
Q_{C1}	17,92 %	25,16 %	-18,88 %
Q_{C2}	44,34 %	46,30 %	-15,38 %
Q_{IMAFO}	43,40 %	45,39 %	-15,00 %
Media	37,67 %	39,86 %	-5,42 %
Máximo	44,34 %	46,30 %	39,21 %
Mínimo	17,92 %	18,24 %	-18,88 %

Cuadro 4.9: Mejora tras aplicar la generalización sobre $C_{\text{Médico}}$ preciso

Medida calidad	Nº reglas	Complejidad	Calidad
Q_{CF}	110	0,24962756	0,50641390
Q_{GI}	110	0,24962756	0,10366419
Q_{P}	110	0,24962756	0,38894758
Q_{SP}	110	0,24962756	0,77083840
Q_{SL}	110	0,24962756	0,57864344
Q_{D}	110	0,24962756	0,80457443
Q_{cob}	110	0,24962756	0,87114453
Q_{cons}	110	0,24962756	0,50789744
$Q_{Coleman}$	110	0,24962756	0,75320697
Q_{Cohen}	110	0,24962756	0,64280623
Q_{AC}	110	0,24962756	0,62598780
Q_{C1}	110	0,24962756	0,38273010
Q_{C2}	110	0,24962756	0,74109584
Q_{IMAFO}	110	0,24962756	0,43607897

Cuadro 4.10: Resultados tras la generación de reglas en $C_{\text{Médico}}$ difuso

Medida calidad	Nº reglas	Complejidad	Calidad
Q_{CF}	108	0,24603620	0,52499163
Q_{GI}	98	0,21370108	0,12223510
Q_{P}	93	0,16314700	0,52132510
Q_{SP}	92	0,16323379	0,78299785
Q_{SL}	96	0,23648076	0,60538300
Q_{D}	107	0,24493448	0,81416875
Q_{cob}	110	0,24962756	0,87114453
Q_{cons}	93	0,16314700	0,59643400
$Q_{Coleman}$	108	0,24603620	0,76249590
Q_{Cohen}	86	0,22797054	0,65640590
Q_{AC}	107	0,24493448	0,63976310
Q_{C1}	108	0,24603620	0,39845392
Q_{C2}	108	0,24667181	0,74971650
Q_{IMAFO}	98	0,21370108	0,46841866

Cuadro 4.11: Resultados tras la generalización sin pérdida en $C_{\text{Médico}}$ difuso

Medida calidad	Nº reglas	Complejidad	Calidad
Q_{CF}	69	0,16622776	0,42979680
Q_{GI}	82	0,15074593	0,10007311
Q_{P}	64	0,13616583	0,48918802
Q_{SP}	63	0,13404284	0,76807630
Q_{SL}	78	0,17085886	0,49395138
Q_{D}	67	0,13601287	0,68966080
Q_{cob}	64	0,13243702	0,82744044
Q_{cons}	64	0,13616583	0,59550940
$Q_{Coleman}$	58	0,12648809	0,63012123
Q_{Cohen}	67	0,13398269	0,57940370
Q_{AC}	67	0,13601287	0,53123250
Q_{C1}	74	0,17453670	0,32973290
Q_{C2}	64	0,13243702	0,63851980
Q_{IMAFO}	65	0,13522878	0,37757920

Cuadro 4.12: Resultados tras la generalización con pérdida en $C_{\text{Médico}}$ difuso

Medida calidad	Nº reglas	Complejidad	Calidad
Q_{CF}	37,27 %	33,41 %	-15,13 %
Q_{GI}	25,45 %	39,61 %	-3,46 %
Q_{P}	41,82 %	45,45 %	25,77 %
Q_{SP}	42,73 %	46,30 %	-0,36 %
Q_{SL}	29,09 %	31,55 %	-14,64 %
Q_{D}	39,09 %	45,51 %	-14,28 %
Q_{cob}	41,82 %	46,95 %	-5,02 %
Q_{cons}	41,82 %	45,45 %	17,25 %
$Q_{Coleman}$	47,27 %	49,33 %	-16,34 %
Q_{Cohen}	39,09 %	46,33 %	-9,86 %
Q_{AC}	39,09 %	45,51 %	-15,14 %
Q_{C1}	32,73 %	30,08 %	-13,85 %
Q_{C2}	41,82 %	46,95 %	-13,84 %
Q_{IMAFO}	40,91 %	45,83 %	-13,41 %
Media	38,57 %	42,73 %	-6,59 %
Máximo	47,27 %	49,33 %	25,77 %
Mínimo	25,45 %	30,08 %	-16,34 %

Cuadro 4.13: Mejora tras aplicar la generalización sobre $C_{\text{Médico}}$ difuso

Medida calidad	Nº reglas	Complejidad	Calidad
Q_{CF}	1265	0,80670774	0,57889515
Q_{GI}	1265	0,80670774	0,03913903
Q_{P}	1265	0,80670774	0,08947705
Q_{SP}	1265	0,80670774	0,65581900
Q_{SL}	1265	0,80670774	0,59897380
Q_{D}	1265	0,80670774	0,79077804
Q_{cob}	1265	0,80670774	0,85340923
Q_{cons}	1265	0,80670774	0,14401528
$Q_{Coleman}$	1265	0,80670774	0,78944755
Q_{Cohen}	1265	0,80670774	0,60228460
Q_{AC}	1265	0,80670774	0,37699038
Q_{C1}	1265	0,80670774	0,41736212
Q_{C2}	1265	0,80670774	0,77561830
Q_{IMAFO}	1265	0,80670774	0,18219034

Cuadro 4.14: Resultados tras la generación de reglas en $C_{\text{Contables}}$ preciso

Medida calidad	Nº reglas	Complejidad	Calidad
Q_{CF}	1242	0,79932940	0,57914160
Q_{GI}	1162	0,76066950	0,04654224
Q_{P}	780	0,62142134	0,15025650
Q_{SP}	1122	0,75435746	0,66001457
Q_{SL}	1242	0,79949920	0,60180330
Q_{D}	1265	0,80653490	0,79087317
Q_{cob}	1228	0,78892934	0,85557060
Q_{cons}	759	0,61243796	0,26458920
$Q_{Coleman}$	1242	0,79932940	0,78957080
Q_{Cohen}	1011	0,71704290	0,60681444
Q_{AC}	780	0,62142134	0,42308253
Q_{C1}	1242	0,79945886	0,41761643
Q_{C2}	1265	0,80640140	0,77574503
Q_{IMAFO}	780	0,62142134	0,22008882

Cuadro 4.15: Resultados tras la generalización sin pérdida en $C_{\text{Contables}}$ preciso

Medida calidad	Nº reglas	Complejidad	Calidad
Q_{CF}	880	0,67041300	0,48014420
Q_{GI}	917	0,67929730	0,04312349
Q_{P}	658	0,56890480	0,14586228
Q_{SP}	645	0,56213737	0,62508650
Q_{SL}	880	0,67052870	0,53053340
Q_{D}	645	0,56213737	0,66105640
Q_{cob}	645	0,56213737	0,77078474
Q_{cons}	650	0,56469345	0,25979197
$Q_{Coleman}$	645	0,56213737	0,65460473
Q_{Cohen}	645	0,56216466	0,59299480
Q_{AC}	645	0,56233126	0,41581970
Q_{C1}	889	0,67456890	0,34592450
Q_{C2}	645	0,56213737	0,64707035
Q_{IMAFO}	650	0,56469345	0,21342246

Cuadro 4.16: Resultados tras la generalización con pérdida en $C_{\text{Contables}}$ preciso

Medida calidad	Nº reglas	Complejidad	Calidad
Q_{CF}	29,15 %	16,13 %	-17,06 %
Q_{GI}	21,08 %	10,70 %	10,18 %
Q_{P}	15,64 %	8,45 %	63,02 %
Q_{SP}	42,51 %	25,48 %	-4,69 %
Q_{SL}	29,15 %	16,13 %	-11,43 %
Q_{D}	49,01 %	30,30 %	-16,40 %
Q_{cob}	47,48 %	28,75 %	-9,68 %
Q_{cons}	14,36 %	7,80 %	80,39 %
$Q_{Coleman}$	48,07 %	29,67 %	-17,08 %
Q_{Cohen}	36,20 %	21,60 %	-1,54 %
Q_{AC}	17,31 %	9,51 %	10,30 %
Q_{C1}	28,42 %	15,62 %	-17,12 %
Q_{C2}	49,01 %	30,29 %	-16,57 %
Q_{IMAFO}	16,67 %	9,13 %	17,14 %
Media	31,72 %	18,54 %	4,96 %
Máximo	49,01 %	30,30 %	80,39 %
Mínimo	14,36 %	7,80 %	-17,12 %

Cuadro 4.17: Mejora tras aplicar la generalización sobre $C_{\text{Contables}}$ preciso

Medida calidad	Nº reglas	Complejidad	Calidad
Q_{CF}	1265	0,80670774	0,57889515
Q_{GI}	1265	0,80670774	0,03913903
Q_{P}	1265	0,80670774	0,08947705
Q_{SP}	1265	0,80670774	0,65581900
Q_{SL}	1265	0,80670774	0,59897380
Q_{D}	1265	0,80670774	0,79077804
Q_{cob}	1265	0,80670774	0,85340923
Q_{cons}	1265	0,80670774	0,14401528
$Q_{Coleman}$	1265	0,80670774	0,78944755
Q_{Cohen}	1265	0,80670774	0,60228460
Q_{AC}	1265	0,80670774	0,37699038
Q_{C1}	1265	0,80670774	0,41736212
Q_{C2}	1265	0,80670774	0,77561830
Q_{IMAFO}	1265	0,80670774	0,18219034

Cuadro 4.18: Resultados tras la generación de reglas en $C_{\text{Contables}}$ difuso

Medida calidad	Nº reglas	Complejidad	Calidad
Q_{CF}	1265	0,80640140	0,57912180
Q_{GI}	947	0,69552374	0,05214428
Q_{P}	581	0,53700024	0,14553821
Q_{SP}	905	0,67208236	0,66799240
Q_{SL}	1242	0,80097340	0,60106970
Q_{D}	1265	0,80653490	0,79087317
Q_{cob}	1207	0,78057150	0,85513680
Q_{cons}	581	0,53700024	0,26495674
$Q_{Coleman}$	1265	0,80640140	0,78956090
Q_{Cohen}	858	0,66251266	0,60845417
Q_{AC}	588	0,54115690	0,42555040
Q_{C1}	1265	0,80653490	0,41748407
Q_{C2}	1265	0,80640140	0,77574503
Q_{IMAFO}	588	0,54115690	0,21894516

Cuadro 4.19: Resultados tras la generalización sin pérdida en $C_{\text{Contables}}$ difuso

Medida calidad	Nº reglas	Complejidad	Calidad
Q_{CF}	838	0,65225250	0,47783625
Q_{GI}	853	0,66218390	0,05117523
Q_{P}	581	0,53700024	0,14553821
Q_{SP}	567	0,52895564	0,64226663
Q_{SL}	660	0,57479860	0,48142200
Q_{D}	567	0,52968320	0,67893220
Q_{cob}	582	0,53475630	0,79497770
Q_{cons}	573	0,53256130	0,26248390
$Q_{Coleman}$	567	0,52895564	0,67246480
Q_{Cohen}	567	0,52971270	0,59739935
Q_{AC}	568	0,52999960	0,42297304
Q_{C1}	838	0,65225250	0,34261343
Q_{C2}	567	0,52895564	0,66260624
Q_{IMAFO}	573	0,53256130	0,21713972

Cuadro 4.20: Resultados tras la generalización con pérdida en $C_{\text{Contables}}$ difuso

Medida calidad	Nº reglas	Complejidad	Calidad
Q_{CF}	33,75 %	19,15 %	-17,46 %
Q_{GI}	32,57 %	17,92 %	30,75 %
Q_{P}	54,07 %	33,43 %	62,65 %
Q_{SP}	55,18 %	34,43 %	-2,07 %
Q_{SL}	47,83 %	28,75 %	-19,63 %
Q_{D}	55,18 %	34,34 %	-14,14 %
Q_{cob}	53,99 %	33,71 %	-6,85 %
Q_{cons}	54,70 %	33,98 %	82,26 %
$Q_{Coleman}$	55,18 %	34,43 %	-14,82 %
Q_{Cohen}	55,18 %	34,34 %	-0,81 %
Q_{AC}	55,10 %	34,30 %	12,20 %
Q_{C1}	33,75 %	19,15 %	-17,91 %
Q_{C2}	55,18 %	34,43 %	-14,57 %
Q_{IMAFO}	54,70 %	33,98 %	19,18 %
Media	49,74 %	30,45 %	7,06 %
Máximo	55,18 %	34,43 %	82,26 %
Mínimo	32,57 %	17,92 %	-19,63 %

Cuadro 4.21: Mejora tras aplicar la generalización sobre $C_{\text{Contables}}$ difuso

Medida calidad	Nº reglas	Complejidad	Calidad
Q_{CF}	542	0,5660638	0,5622270
Q_{GI}	542	0,5660638	0,5107154
Q_{P}	542	0,5660638	0,5443431
Q_{SP}	542	0,5660638	0,7716786
Q_{SL}	542	0,5660638	0,6942259
Q_{D}	542	0,5660638	0,8457655
Q_{cob}	542	0,5660638	0,7954575
Q_{cons}	542	0,5660638	0,6808877
$Q_{Coleman}$	542	0,5660638	0,7811135
Q_{Cohen}	542	0,5660638	0,7956100
Q_{AC}	542	0,5660638	0,7112396
Q_{C1}	542	0,5660638	0,4986896
Q_{C2}	542	0,5660638	0,7575944
Q_{IMAFO}	542	0,5660638	0,5520869

Cuadro 4.22: Resultados tras la generación de reglas en C_{Censo} preciso

Medida calidad	Nº reglas	Complejidad	Calidad
Q_{CF}	471	0,5235664	0,5883291
Q_{GI}	471	0,5235664	0,5719318
Q_{P}	526	0,5427836	0,5661074
Q_{SP}	471	0,5044676	0,7958164
Q_{SL}	487	0,5430365	0,7134301
Q_{D}	487	0,5018543	0,8488071
Q_{cob}	471	0,5049198	0,8257290
Q_{cons}	526	0,5336881	0,7064086
$Q_{Coleman}$	471	0,5235664	0,7941564
Q_{Cohen}	471	0,5235664	0,8094239
Q_{AC}	471	0,5128989	0,7285597
Q_{C1}	471	0,5235664	0,5314555
Q_{C2}	471	0,5235664	0,7706867
Q_{IMAFO}	471	0,5235664	0,5771512

Cuadro 4.23: Resultados tras la generalización sin pérdida en C_{Censo} preciso

Medida calidad	Nº reglas	Complejidad	Calidad
Q_{CF}	378	0,3872042	0,5506121
Q_{GI}	378	0,3940104	0,5402204
Q_{P}	330	0,3237454	0,4759803
Q_{SP}	330	0,3174483	0,7449263
Q_{SL}	378	0,3766919	0,5757355
Q_{D}	330	0,3263264	0,7960270
Q_{cob}	330	0,3174483	0,7750619
Q_{cons}	330	0,3174483	0,6422902
$Q_{Coleman}$	330	0,3260262	0,7161411
Q_{Cohen}	330	0,3260262	0,7255325
Q_{AC}	330	0,3174483	0,6378748
Q_{C1}	378	0,3872042	0,4908171
Q_{C2}	330	0,3260262	0,6947261
Q_{IMAFO}	330	0,3254407	0,4625765

Cuadro 4.24: Resultados tras la generalización con pérdida en C_{Censo} preciso

Medida calidad	Nº reglas	Complejidad	Calidad
Q_{CF}	30,26 %	31,60 %	-2,07 %
Q_{GI}	30,26 %	30,39 %	5,78 %
Q_{P}	39,11 %	42,81 %	-12,56 %
Q_{SP}	39,11 %	43,92 %	-3,47 %
Q_{SL}	30,26 %	33,45 %	-17,07 %
Q_{D}	39,11 %	42,35 %	-5,88 %
Q_{cob}	39,11 %	43,92 %	-2,56 %
Q_{cons}	39,11 %	43,92 %	-5,67 %
$Q_{Coleman}$	39,11 %	42,40 %	-8,32 %
Q_{Cohen}	39,11 %	42,40 %	-8,81 %
Q_{AC}	39,11 %	43,92 %	-10,32 %
Q_{C1}	30,26 %	31,60 %	-1,58 %
Q_{C2}	39,11 %	42,40 %	-8,30 %
Q_{IMAFO}	39,11 %	42,51 %	-16,21 %
Media	36,58 %	39,83 %	-6,93 %
Máximo	39,11 %	43,92 %	5,78 %
Mínimo	30,26 %	30,39 %	-17,07 %

Cuadro 4.25: Mejora tras aplicar la generalización sobre C_{Censo} preciso

Medida calidad	Nº reglas	Complejidad	Calidad
Q_{CF}	529	0,5678742	0,5939430
Q_{GI}	529	0,5678742	0,5783234
Q_{P}	529	0,5678742	0,5457484
Q_{SP}	529	0,5678742	0,7845218
Q_{SL}	529	0,5678742	0,7233468
Q_{D}	529	0,5678742	0,8470481
Q_{cob}	529	0,5678742	0,8152627
Q_{cons}	529	0,5678742	0,6791264
$Q_{Coleman}$	529	0,5678742	0,7969715
Q_{Cohen}	529	0,5678742	0,8104868
Q_{AC}	529	0,5678742	0,7255757
Q_{C1}	529	0,5678742	0,5352445
Q_{C2}	529	0,5678742	0,7732873

Cuadro 4.26: Resultados tras la generación de reglas en C_{Censo} difuso

Medida calidad	Nº reglas	Complejidad	Calidad
Q_{CF}	513	0,5430480	0,6114758
Q_{GI}	467	0,5249555	0,6085516
Q_{P}	502	0,5302344	0,5731663
Q_{SP}	502	0,5207723	0,8080809
Q_{SL}	483	0,5447211	0,7326140
Q_{D}	529	0,5180513	0,8513395
Q_{cob}	502	0,5207723	0,8388534
Q_{cons}	502	0,5370002	0,7023768
$Q_{Coleman}$	513	0,5430480	0,8057305
Q_{Cohen}	467	0,5249555	0,8196657
Q_{AC}	502	0,5310505	0,7452773
Q_{C1}	513	0,5430480	0,5564904
Q_{C2}	513	0,5430480	0,7822885
Q_{IMAFO}	502	0,5310505	0,5933619

Cuadro 4.27: Resultados tras la generalización sin pérdida en C_{Censo} difuso

Medida calidad	Nº reglas	Complejidad	Calidad
Q_{CF}	311	0,3209981	0,5121795
Q_{GI}	311	0,3202355	0,5007925
Q_{P}	311	0,3202355	0,5081036
Q_{SP}	311	0,3127072	0,7739045
Q_{SL}	311	0,3209981	0,6400797
Q_{D}	311	0,3209981	0,8116171
Q_{cob}	311	0,3127072	0,8093344
Q_{cons}	311	0,3128058	0,6665680
$Q_{Coleman}$	311	0,3209981	0,7547395
Q_{Cohen}	311	0,3196942	0,7680693
Q_{AC}	311	0,3196942	0,6865367
Q_{C1}	311	0,3209981	0,4477699
Q_{C2}	311	0,3209981	0,7314003
Q_{IMAFO}	311	0,3202355	0,5203288

Cuadro 4.28: Resultados tras la generalización con pérdida en C_{Censo} difuso

Medida calidad	Nº reglas	Complejidad	Calidad
Q_{CF}	41,21 %	43,47 %	-13,77 %
Q_{GI}	41,21 %	43,61 %	-13,41 %
Q_{P}	41,21 %	43,61 %	-6,90 %
Q_{SP}	41,21 %	44,93 %	-1,35 %
Q_{SL}	41,21 %	43,47 %	-11,51 %
Q_{D}	41,21 %	43,47 %	-4,18 %
Q_{cob}	41,21 %	44,93 %	-0,73 %
Q_{cons}	41,21 %	44,92 %	-1,85 %
$Q_{Coleman}$	41,21 %	43,47 %	-5,30 %
Q_{Cohen}	41,21 %	43,70 %	-5,23 %
Q_{AC}	41,21 %	43,70 %	-5,38 %
Q_{C1}	41,21 %	43,47 %	-16,34 %
Q_{C2}	41,21 %	43,47 %	-5,42 %
Q_{IMAFO}	41,21 %	43,61 %	-8,37 %
Media	41,21 %	43,85 %	-7,12 %
Máximo	41,21 %	44,93 %	-0,73 %
Mínimo	41,21 %	43,47 %	-16,34 %

Cuadro 4.29: Mejora tras aplicar la generalización sobre C_{Censo} difuso

En la tabla 4.30 se recoge un resumen de los resultados obtenidos. En ella se presentan los valores medios obtenidos en cada una de las etapas que hemos considerado. La columna *Significativo* muestra el resultado de aplicar una prueba de diferencias de medias basada en la prueba de Student. Se ha aplicado considerando observaciones por pares y 2 colas, dado que no teníamos ninguna suposición previa. Comentaremos cada dominio de forma independiente y posteriormente sobre el comportamiento general.

Resultados sobre $C_{\text{Médico}}$: Lo primero que podemos destacar al estudiar los resultados es ver que en ningún caso se ha llegado al umbral de calidad establecido respecto a la calidad del conjunto de partida. En media, la reducción ha sido del 5,42 % en el caso preciso y 6,6 % en el difuso, aunque esta diferencia entre ambos enfoques no es estadísticamente significativa. Además, se produce el hecho de que en la mayoría de las casos al realizar la generalización sin pérdidas se mejora la calidad del conjunto de partida (Figura 4.7).

En cuanto a la complejidad tampoco existen diferencias significativas entre ambos DataCubos, estando la reducción entorno al 40 %. Sin embargo, cabe destacar que, aunque la diferencia entre las complejidades obtenidas no sea significativa, las reglas obtenidas en el caso difuso serán más inteligibles de cara al usuario al utilizar conceptos más cercanos a su espacio de razonamiento. Como ejemplo de este hecho, en los experimentos se ha encontrado la siguiente regla tanto en el caso difuso como en el preciso:

SI [Causa es enf. del aparato respiratorio], [la duración es normal] y [es del área metropolitada] ENTONCES [No se requieren implantes],

con soporte $0{,}12423447$ y factor de certeza $0{,}9843216$ pera el caso difuso y valores $0{,}10903524$ y $0{,}96863496$ para el preciso.

Veamos en qué se traduce esta regla en cada caso. En el preciso, la duración

Medida	Preciso	Difuso	Significativo
Generalización sin pérdida			
Complejidad media	0,229315271	0,221832727	0,1201
Calidad media	0,617233886	0,608138139	0,2948
Generalización con pérdida			
Complejidad media	0,150598591	0,142953078	0,0645
Calidad media	0,545852952	0,534306113	0,096
Mejora total			
Reducción del nº de reglas	37,67 %	38,57 %	0,5799
Reducción de complejidad media	39,86 %	42,73 %	0,0794
Ganancia de calidad media	-5,42 %	-6,59 %	0,7213

Comparativa de resultados entre $C_{\text{Médico}}$ preciso y difuso

Medida	Preciso	Difuso	Significativo
Generalización sin pérdida			
Complejidad media	0,736303881	0,700017996	0,0053
Calidad media	0,512979231	0,513755202	0,3564
Generalización con pérdida			
Complejidad media	0,594877312	0,561044933	0,00001
Calidad media	0,456158537	0,46070205	0,3534
Mejora total			
Reducción del nº de reglas	43,31 %	49,74 %	0,00001
Reducción de complejidad media	26,26 %	30,45 %	0,00001
Ganancia de calidad media	4,96 %	7,06 %	0,2152

Comparativa de resultados entre $C_{\text{Contables}}$ preciso y difuso

Medida	Preciso	Difuso	Significativo
Generalización sin pérdida			
Complejidad media	0,522043809	0,532553978	0,0018
Calidad media	0,701999491	0,716376602	0,0001
Generalización con pérdida			
Complejidad media	0,340606766	0,318878836	0,0149
Calidad media	0,630608697	0,652244546	0,0417
Mejora total			
Reducción del nº de reglas	36,58 %	41,21 %	0,011
Reducción de complejidad media	39,83 %	43,85 %	0,0116
Reducción de calidad media	-6,93 %	-7,12 %	0,9329

Comparativa de resultados entre C_{Censo} preciso y difuso

Cuadro 4.30: Comparativa de resultados

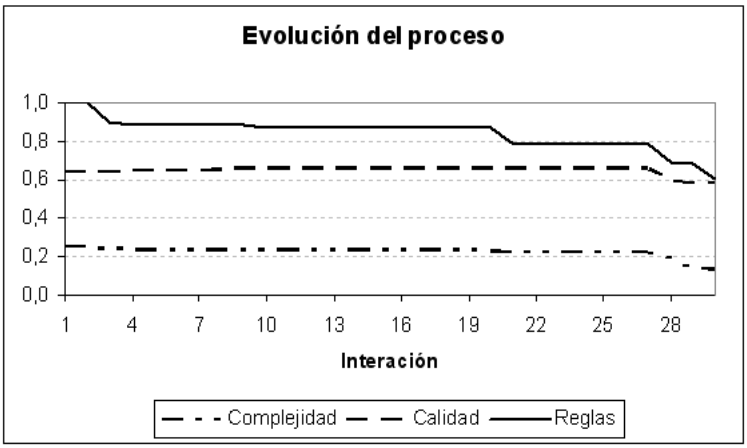


Figura 4.7: Evolución del proceso sobre C_{Medico} difuso utilizando la medida de calidad Q_{Cohen}. El valor para el número de reglas representa el porcentaje sobre el número inicial

se ha definido como el intervalo [0,1] y el área metropolitana como un conjunto de localidades. Por esto, el significado real de la regla será

SI [Causa es enf. del aparato respiratorio], [la duración está en [0,1]] y [es de Armilla, Otura, Jun, ...] ENTONCES [No se requieren implantes]

De esta manera si una intervención durara hora y media ya no satisfacería el antecedente de la regla y no sería válida. Pero para nosotros una hora y media y una hora son similares y consideraremos que en este caso se cumpliría la regla pero quizá con menor confianza.

En el caso difuso, tanto la duración como el área de influencia modelada con el área metropolitana se han definido mediante conjuntos difusos (véase Sección 5.1). Estos se han establecido de forma que representaran el concepto de forma más cercana al usuario. De esta forma, se podría interpretar la regla de un modo

más directo dado que los conceptos que utiliza tienen una correspondencia más cercana con los que utilizamos en nuestro espacio cognitivo.

Además, observando los soportes correspondientes vemos que en el caso difuso la regla es más representativa (12 % de las transacciones frente al 10 %) y sin pérdida de precisión, incluso mejorándola ligeramente. Así pues, los conceptos difusos utilizados en la regla se adaptan mejor al dominio de los datos subyacentes.

En cuanto a la calidad, ambos enfoques presentan un comportamiento muy similar, sin apreciarse diferencias significativas.

Resultados sobre $C_{\textbf{Contables}}$: En el caso anterior hemos visto que no se llega al umbral de calidad establecido. En los DataCubos contables también se presenta este hecho, pero llegando incluso a obtenerse en media un resultado mejor al de partida tras la generalización con pérdida. En el caso del DataCubo difuso utilizando la medida de calidad consistencia/confianza se llega a mejorar la calidad tras el proceso en un 82 % (la evolución en este experimento se recoge en la Figura 4.8).

A la vista de la tabla 4.30 podemos ver que en este caso si existe una diferencia significativa en cuanto a la complejidad de los resultados significativa. En todos los pasos a tenor de los resultados intermedios y finales se observa que siempre el modelo difuso presenta una menor complejidad y además se consigue una reducción de la misma superior (30,5 % frente al 26,3 %). A este hecho hay que añadirle que el modelo difuso consigue una mayor reducción del número de reglas finales, reduciendo casi a la mitad el conjunto frente al 43,3 % del preciso.

En cuestiones de calidad podemos ver que ambos enfoques de modelado obtienen resultados similares sin diferencias significativas. Así pues, el modelo difuso consigue la misma calidad que el preciso pero consiguiendo una mayor reducción de la complejidad y número de reglas.

Si tenemos en cuenta que en el difuso utilizamos conceptos más cercanos al

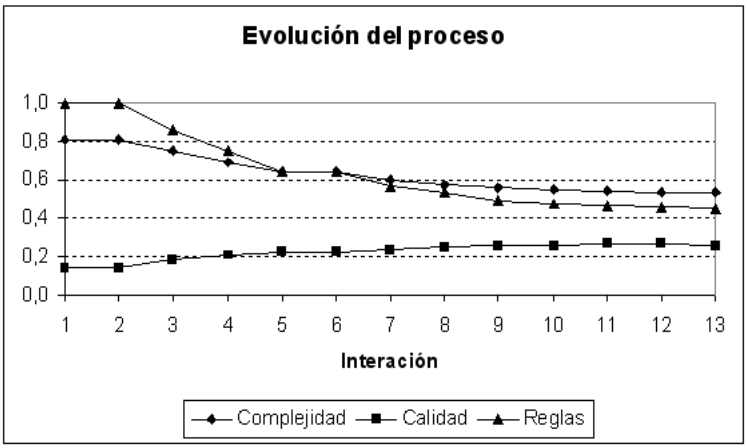


Figura 4.8: Mejor evolución del proceso sobre $C_{\text{Contables}}$ difuso utilizando la medida de calidad Q_{cons}. El valor para el número de reglas representa el porcentaje sobre el número inicial

usuario, unido a la reducción de complejidad, tenemos que este enfoque obtiene mejores resultados de cara a la interpretabilidad.

Resultados sobre C_{Censo}: Se vuelve a cumplir el mantenimiento de la calidad final de los conjuntos de reglas por encima del umbral. En estos DataCubos, tanto en el preciso como el difuso, se produce un mejora de la calidad según todas las medidas consideradas al realizar la generalización sin permitir pérdida. Un ejemplo de la evolución de las medidas durante el proceso se muestra en la Figura 4.9.

En este problema, al realizar la reducción de complejidad sin pérdida de calidad el modelo preciso se nos presenta mejor. Sin embargo, en el siguiente paso no consigue tanta reducción como el difuso. En cuanto a complejidad total y reducción de la misma el difuso se presenta mejor en el proceso global. En cuanto

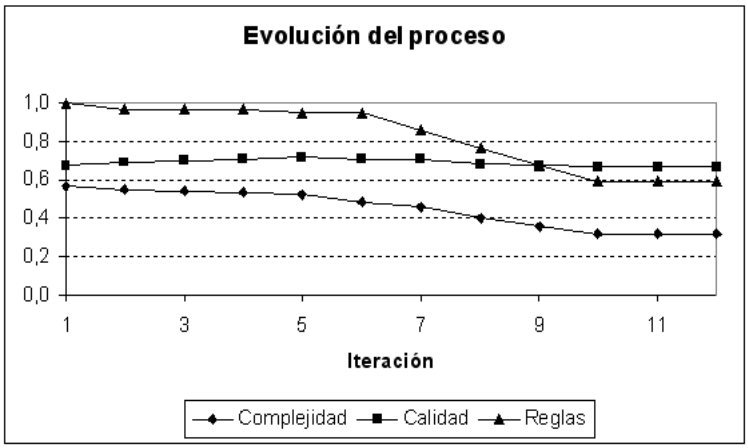


Figura 4.9: Evolución del proceso sobre C_{Censo} difuso utilizando la medida de calidad Q_{cons}. El valor para el número de reglas representa el porcentaje sobre el número inicial

al número de reglas que se consiguen eliminar, el difuso consigue una reducción entorno al 41 % mientras el preciso se queda en un 36,5 %.

Hablando de la calidad, en este problema se cambia la tónica de los anteriores. Mientras en los anteriores los dos modelos propuestos tenían una calidad comparable en todas las etapas, en este dominio el modelo difuso presenta mejoras significativas desde el mismo momento de la generación de reglas.

General: Observando los tres experimentos en conjunto, podemos ver que en todos ellos el proceso de generalización sin pérdida de calidad suele conseguir una mejora de los resultados. De esta forma, al aplicar la generalización en la que ya si aceptamos reducir la calidad, se parte de un umbral más alto para detener el proceso. Esto hace que el resultado final nunca llegue a alcanzar una gran pérdida de calidad.

En todos los casos se ha conseguido una reducción del número de reglas cercana al 40 % (destacando el modelo difuso en el DataCubo con datos contables que ha estado muy cerca del 50 %) y una reducción de complejidad oscilando desde el 26 % al casi 44 %.

En general podemos decir que la modelización difusa de los DataCubos ha resultado mejor a la hora de la reducción de complejidad. En el caso de $C_{\text{Médico}}$ ambos han tenido comportamiento similares, salvo la utilización de conceptos más intuitivos en el caso difuso ya comentado. En el resto de los dominios, el modelo difuso se ha mostrado mejor, superando siempre al preciso en reducción de complejidad, y cuando menos obteniendo una calidad similar al preciso. En este sentido, el DataCubo C_{Censo} difuso se ha comportado mejor en todos los aspectos (complejidad y calidad) respecto al preciso.

Para poder analizar el comportamiento de cada medida de calidad necesitaríamos realizar experimentos sobre un número mayor de DataCubos de forma que los resultados fueran significativos. Sin embargo, comentaremos los rasgos observados durante los experimentos, aunque teniendo presente este hecho.

Si nos fijamos en la complejidad del conjunto resultado, en todas las medidas salvo con el factor de certeza (C_{CF}) y la precisión utilizada por IMAFO (C_{AC}) el proceso sobre el DataCubo difuso ha obtenido un resultado con menor valor que el preciso. En estas dos, dependiendo del dominio el valor menor se ha ido alternando.

Observando la calidad del conjunto de reglas resultado, tenemos que el factor de certeza y la combinación Q_{C1} siempre han obtenido mayores valores para los modelos precisos. Por contra, las medidas Q_{SP}, Q_{cob}, Q_{cons}, Q_{Cohen} y Q_{AC} se han mostrado con valores superiores en todos los casos para los DataCubos difusos.

La reducción media de complejidad obtenida para cada medida se recoge en la Tabla 4.31. Como puede verse, en el caso preciso la medida que ha obtenido la

Medida	Preciso	Difuso	Medida	Preciso	Difuso
Q_{CF}	27,44 %	32,01 %	$Q_{Coleman}$	39,49 %	41,45 %
Q_{GI}	21,47 %	33,71 %	Q_{cons}	39,25 %	42,41 %
Q_P	38,95 %	40,83 %	Q_{Cohen}	39,63 %	41,46 %
Q_{SP}	39,88 %	41,89 %	Q_{AC}	39,87 %	41,17 %
Q_{SL}	25,55 %	34,59 %	Q_{C1}	24,38 %	30,90 %
Q_D	39,35 %	41,11 %	Q_{C2}	39,67 %	41,62 %
Q_{cob}	40,18 %	41,86 %	Q_{IMAFO}	39,30 %	41,14 %

Cuadro 4.31: Reducción media obtenida según la medida de calidad utilizada

mayor reducción ha sido Q_{cob} (40,18 %), y la menor Q_{GI} (21,47 %). Estas medidas han tenido un comportamiento similar en el caso difuso pero no corresponden a la mayor o menor. En el caso difuso las medidas son otras: la mayor reducción corresponde a la medida Q_{cons} (42,41 %)y la menor Q_{C1} (30,90 %). De forma general se puede hacer notar que cuando una medida ha sido buena en un caso también presenta una buena reducción en el DataCubo análogo, al igual que aquellas que han obtenido una reducción menor.

Considerando ambos casos, parece que la medida que ha conseguido una mayor reducción de complejidad es la cobertura. Esta medida ha obtenido la máxima reducción en el preciso y la tercera, muy cercana a las primeras, en el caso difuso. El caso de la medida de calidad Q_{cons} aunque en el caso difuso ha obtenido la máxima reducción, en el preciso tiene siete medidas que han conseguido mayor reducción.

4.3.4. Análisis de tiempos

Los experimentos han sido ejecutados sobre plataformas muy diferentes y utilizando las posibilidades de distribución de cálculo que aporta el sistema imple-

	Preciso	**Difuso**
Conjuntos frecuentes	1,25	1,28
Generación de reglas	0,34	0,34
Total	1'60	1'62

Cuadro 4.32: Tiempos expresado en horas para la generación de reglas sobre los DataCubos $C_{\text{Contables}}$ preciso y difuso

mentado[2]. De esta forma, hablar de tiempos a escala total se hace imposible.

Para abordar el problema del tiempo hemos ejecutado un problema por completo en una misma plataforma. Aunque el análisis no sea extrapolable sí nos dará una idea del comportamiento del proceso en líneas generales.

Todas las pruebas utilizando los DataCubos sobre datos contables se han ejecutado sobre una plataforma formada por un ordenador tipo PC con un procesador Pentium IV a 3,2 Ghz y tecnología HT con 1 GB de memoria RAM, funcionando bajo Windows XP Professional. En cada DataCubo considerado, el utilizar una medida u otra de calidad no influye para la generación inicial de reglas, por lo tanto, estas se han generado una única vez y el resultado se ha utilizado de partida para las 14 medidas de calidad consideradas. El tiempo de generación de reglas se encuentra desglosado en la Tabla 4.32. Los tiempos empleados en la generalización para cada una de las medidas se recoge en la Tabla 4.33.

Considerando los tiempos medios para la generalización, la distribución de tiempos en cada fase del proceso se recoge en la Tabla 4.34. Como puede verse, en este problema la generación de reglas consume entorno al 82 % del tiempo to-

[2]Los experimentos se han lanzado utilizando por separado un ordenador Origin 3800 con 64 procesadores R14000A a 600 MHz (utilizando un máximo de 14 procesadores en paralelo), un ordenador tipo PC trabajando de forma independiente y varios PC's de diferentes velocidades situados en Jaén y Granada trabajando conectados a través de Internet mediante túneles SSH con diferentes incidencias en las comunicaciones.

Medida calidad	**Preciso**			**Difuso**		
	Sin pérdida	Con pérdida	Total	Sin pérdida	Con pérdida	Total
Q_{CF}	0,330	0,066	0,396	0,366	0,206	0,572
Q_{GI}	0,158	0,103	0,261	0,149	0,050	0,199
Q_{P}	0,143	0,059	0,202	0,140	0,008	0,148
Q_{SP}	0,449	0,059	0,508	0,263	0,035	0,298
Q_{SL}	0,276	0,165	0,441	0,318	0,102	0,421
Q_{D}	0,235	0,072	0,307	0,292	0,066	0,358
Q_{cob}	0,492	0,068	0,560	0,717	0,062	0,779
Q_{cons}	0,159	0,057	0,216	0,140	0,014	0,154
$Q_{Coleman}$	0,331	0,087	0,418	0,368	0,065	0,433
Q_{Cohen}	0,179	0,051	0,229	0,181	0,037	0,218
Q_{AC}	0,143	0,070	0,213	0,140	0,029	0,169
Q_{C1}	0,268	0,211	0,479	0,289	0,263	0,552
Q_{C2}	0,301	0,072	0,373	0,368	0,065	0,433
Q_{IMAFO}	0,143	0,064	0,208	0,140	0,020	0,160
Media	0,258	0,086	0,344	0,276	0,073	0,350

Cuadro 4.33: Tiempos expresado en horas para la generalización sobre los DataCubos $C_{\text{Contables}}$ preciso y difuso

Fase	Preciso		Difuso	
	Tiempo	Porcentaje	Tiempo	Porcentaje
Conjuntos frecuentes	1,25	64,51 %	1,28	64,78 %
Generación de reglas	0,34	17,78 %	0,34	17,49 %
Total generación reglas	1’60	82,29 %	1’62	82,27 %
Generalización sin pérdida	0,26	13,28 %	0,28	14,02 %
Generalización con pérdida	0,09	4,43 %	0,07	3,71 %
Total generalización	0,34	17,71 %	0,35	17,73 %
Total	1,94	100,00 %	1,97	100,00 %

Cuadro 4.34: Tiempos expresados en horas para el proceso completo sobre los DataCubos $C_{\text{Contables}}$ preciso y difuso

tal del proceso (65 % para la obtención de los conjuntos frecuentes y 17 % en la generación de reglas) tanto en el caso preciso como difuso. La posterior generalización del resultado supone tan solo un 18 % del tiempo (algo menos de la quinta parte).

4.4. Conclusiones

El algoritmo COGARE que hemos presentado funciona bajo la filosofía de obtener un resultado con una complejidad controlada de cara a el usuario. Para conseguirlo se utilizan las relaciones jerárquicas definidas en la dimensiones. Este hecho unido al uso de conceptos modelados de forma más cercana al usuario, utilizando un modelo multidimensional difuso, ayudan a reducir la complejidad final.

El método se ha probado sobre datos obtenidos de dominios reales. Los resultados obtenidos muestran que, con una sobrecarga en el tiempo entorno al 20 %, se

llega a conseguir una reducción de la complejidad entorno al 40 %. Si nos fijamos exclusivamente en el número de reglas, el número se reduce hasta en un 50 %. En los experimentos se ha comprobado que de forma general la utilización de conceptos difusos en las jerarquías ayuda en el proceso de reducción de complejidad (sin contar que las reglas resultantes son más inteligibles para el usuario al utilizar representaciones más intuitivas).

Como puede verse, los resultados obtenidos son muy prometedores e indican un buen funcionamiento del algoritmo. Uno de los siguientes pasos será probar a fondo el comportamiento del algoritmo mediante una experimentación mayor.

Capítulo 5

Ejemplo de Esquemas Multidimensionales Difusos

No importa qué es lo que vaya mal, siempre hay alguien que ya lo sabía

LEY DE EVAN Y BJORN
Ley de Murphy

En esta sección presentaremos tres ejemplos de modelos multidimensionales difusos construidos sobre tres dominios distintos. Estos modelos han sido los utilizados en la Sección para probar el algoritmo COGARE.

En cada caso presentaremos los dominios sobre los que se definen, la estructura de las dimensiones con las jerarquías definidas y los hechos considerados. Posteriormente presentaremos ejemplos de algunas consultas realizadas sobre los datacubos resultados.

Comenzaremos presentando un ejemplo construido sobre datos facilitados por

el Servicio de Informática del Hospital Clínico de Granada sobre intervenciones quirúrgicas. El siguiente datacubo se construirá utilizando datos de empresas recopilados por Asexor S.A. sobre empresas españolas y si han presentado quiebra financiera o no. El último ejemplo se construirá utilizando la base de datos *Adult* del repositorio de datos para procesos de extracción de conocimiento de la Universidad de California[1].

5.1. DataCubo sobre datos médicos

En este primer datacubo trabajaremos con datos procedentes del Hospital Clínico de Granada sobre intervenciones no suspendidas que se han realizada entre los años 2002 y 2004 de pacientes provenientes de la provincia de Granada (50.185 registros).

En el siguiente punto presentaremos la estructura del DataCubo construido, indicando la jerarquía de cada dimensión. Posteriormente veremos algunos ejemplos de consultas sobre el mismo.

5.1.1. Estructura del DataCubo

En la Figura 5.1 se recoge el modelo multidimensional que hemos construido sobre los datos médicos. Como puede verse, se han definido 6 dimensiones con diferentes relaciones difusas entre algunos niveles. Para los cálculos de las relaciones de parentesco extendido utilizaremos el mínimo y el máximo como t-norma y t-conorma respectivamente, a menos que al hablar de la dimensión digamos lo contrario. Veamos la estructura en detalle de cada dimensión.

[1]URL: http://kdd.ics.uci.edu/

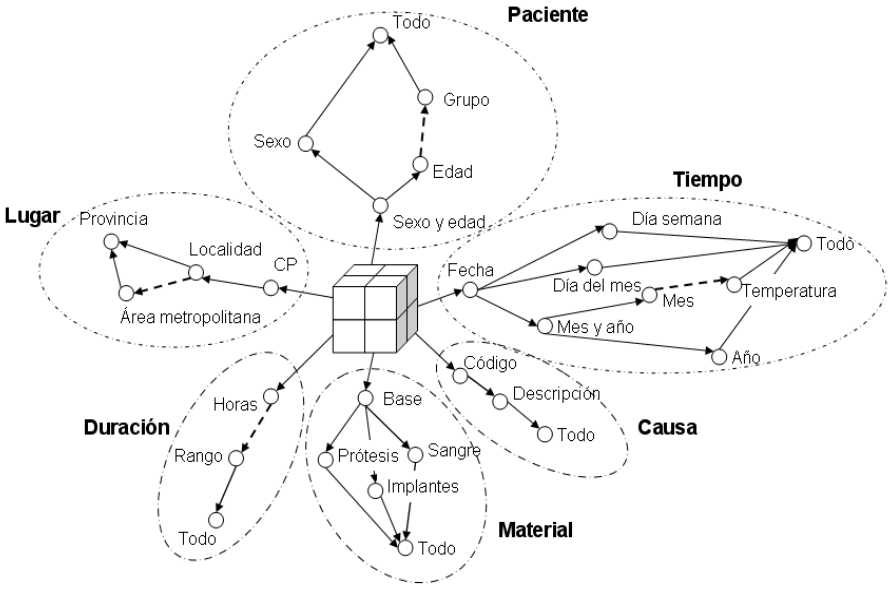


Figura 5.1: DataCubo sobre datos médicos

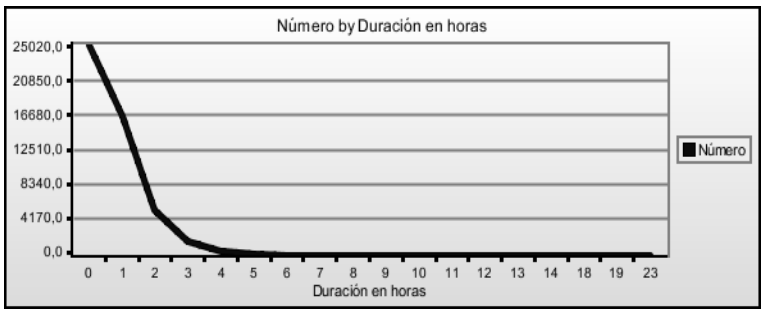


Figura 5.2: Distribución de las intervenciones según la duración

5.1.1.1. Dimensión *Duración*

Uno de las variables que utilizaremos para analizar las intervenciones será el tiempo que ha necesitado la intervención. El nivel base de esta dimensión nos dará el número de horas de cada intervención. Sobre este nivel definimos un nivel que clasifica la duración según sea *normal*, *larga* o *muy larga*. Para definir estos conceptos utilizaremos intervalos difusos que se aproxima más al espacio de razonamiento del usuario que pueda utilizar el DataCubo. Para la definición de las conceptos nos hemos fijado en la distribución de las operaciones según su duración (Figura 5.2).

A la vista de la distribución hemos definido las etiquetas tal y como se recogen en la Figura 5.3. De esta forma, la dimensión resultante tiene la estructura siguiente:

$$Duración = (\{Horas, Rango, Todo\}, \leq_{Duración}, Horas, Todo),$$

donde $\leq_{Duración}$ establece la ordenación de los niveles tal y como se recoge en la Figura 5.1.

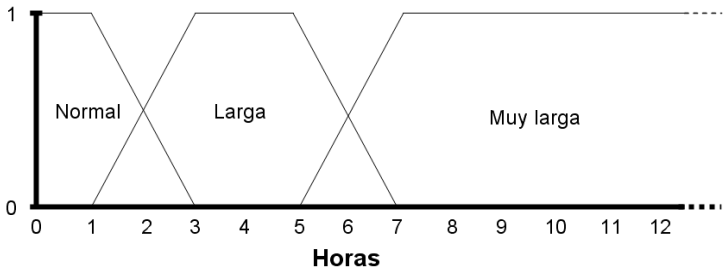


Figura 5.3: Valores y agrupación del nivel *Rango* en la dimensión *Duración*

5.1.1.2. Dimensión *Tiempo*

Esta dimensión sitúa las intervenciones según la fecha en la que se realizaron. El nivel base de la dimensión (*Fecha*) se define según el día de realización. Sobre este nivel definimos las niveles siguientes:

- *Día de la semana*: Este nivel agrupa las fechas según al día de la semana al que corresponda (lunes, martes, miércoles, jueves, viernes, sábado y domingo).

- *Día del mes*: en este caso, las fechas se agrupan según el día del mes que sea (1 a 31).

- *Mes y año*: Los agrupa según el mes y año al que pertenezca (p.e. Enero del 2000, Febrero del 2003, etc.). A su vez, este nivel se agrupa según el mes concreto y el año al que pertenezca (niveles *Mes* y *Año* respectivamente). Sobre el nivel *Mes* se ha definido otro llamado *Temperatura* que intenta agruparlos según sean meses *fríos*, *cálidos* o *templados*. La asignación de un mes a una categoría a otra no es directa. Por ello, hemos preferido definirlo utilizando conjuntos difusos (Figura 5.4).

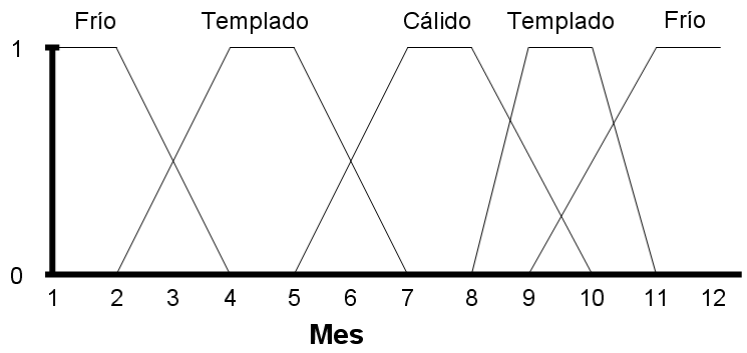


Figura 5.4: Clasificación de meses según su temperatura

Sobre estos niveles se define uno que los agrupa a todos (*Todo*). Así pues, la estructura de la dimensión traducida a nuestra estructura multidimensional será

Tiempo =($\{$*Fecha, Día de la semana, Día del mes, Mes y año, Temperatura, Año, Todo*$\}$*,*$\leq_{Tiempo}$*,Fecha,Todo)*

5.1.1.3. Dimensión *Paciente*

A la hora de tener en cuenta el paciente sometido a la intervención, los definiremos al nivel de detalle de edad y sexo. Esta conjunción será el nivel base de la dimensión *Paciente*. Serán valores formados por pares (p.e. 23 años y mujer, 50 años y hombre, etc.). Sobre el base añadiremos los siguientes niveles:

- *Sexo*: Parece lógico que una primera división se haga en función de si son hombres o mujeres.
- *Edad*: E igualmente parece lógico hacerlo por la edad del paciente. Los valores de este nivel los agruparemos según los pacientes sean *Jóvenes*, *Adultos* o *Mayores*. Estos valores se definirán en el nivel *Grupo* utilizando

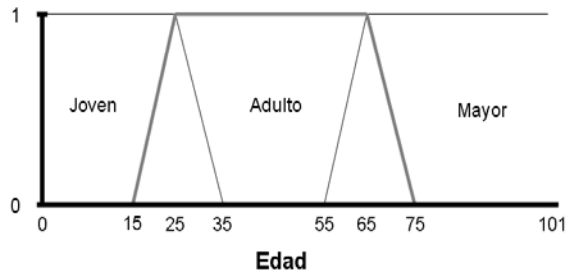


Figura 5.5: Grupos de edades

conceptos difusos, más cercanos a como lo utilizamos los humanos. Las etiquetas definidas viene recogidas en la Figura 5.5.

La estructura resultante de la dimensión sería

Paciente =($\{$Sexo y edad, Sexo, Edad, Grupo, Todo$\}$,$\leq_{Paciente}$,Sexo y edad,Todo)

5.1.1.4. Dimensión *Material*

Esta dimensión modela las necesidades especiales de la intervención. En ella se caracterizan las intervenciones atendiendo a si necesitaron *sangre*, *implantes* y/o *prótesis*. El nivel base modela la unión de las tres características (p.e. un valor del nivel será *la intervención necesitó sangre, no necesitó implantes y sí necesitó prótesis*).

Sobre este nivel hemos establecido tres que consideran cada necesidad por separado. Así pues, la estructura final de la dimensión será

Material =($\{$Base, Sangre, Implantes, Prótesis, Todo$\}$,$\leq_{Material}$,Base,Todo)

5.1.1.5. Dimensión *Lugar*

Otra característica que modelamos del paciente es el lugar de procedencia del mismo. Como hemos comentado nos centramos en pacientes provenientes de la provincia de Granada.

El nivel base que se ha considerado es el código postal del lugar de procedencia. Sobre este, hemos definido la localidad a la que pertenece. La información de los códigos postales asociados a cada localidad han sido obtenidos de la Sociedad Estatal de Correos y Telégrafos S.A.[2].

El nivel más alto de la dimensión es *Provincia*. En el resto de dimensiones hemos definido este nivel con el nombre *Todo* dado que agrupaba a todos los elementos. En este caso, dado que sólo consideramos los pacientes provenientes de la provincia de Granada, este nivel tendrá un única valor que agrupará a todos los demás. Por lo tanto su papel es el mismo.

Entre el nivel localidad y provincia hemos definido un nivel intermedio que hemos llamado *área metropolitana*. En este nivel pretendemos definir el conceptos de ser una localidad bajo el área de influencia más directa de la capital. El nombre que le hemos dado se debe a que nos hemos basado en la agrupación que se ha realizada por parte del Consorcio de Transportes del Área de Granada[3] considerando el área metropolitana.

Esta relación la hemos definido utilizando una relación difusa. Este se debe a que no todas las localidades se pueden considerar igualmente cercanas a la capital. De hecho, el propio Consorcio de Transporte define tres zonas según sea la distancia a Granada capital (Figura 5.6).

Para definir el grado en que una localidad pertenece al área de influencia hemos asignado a cada zona un grado de pertenencia. De esta forma, las localidades pertenecientes a la zona A tendrán un grado de pertenencia 1; los pertenecientes

[2]URL: http://www.correos.es

[3]URL: http://www.consorciotransportes-granada.com

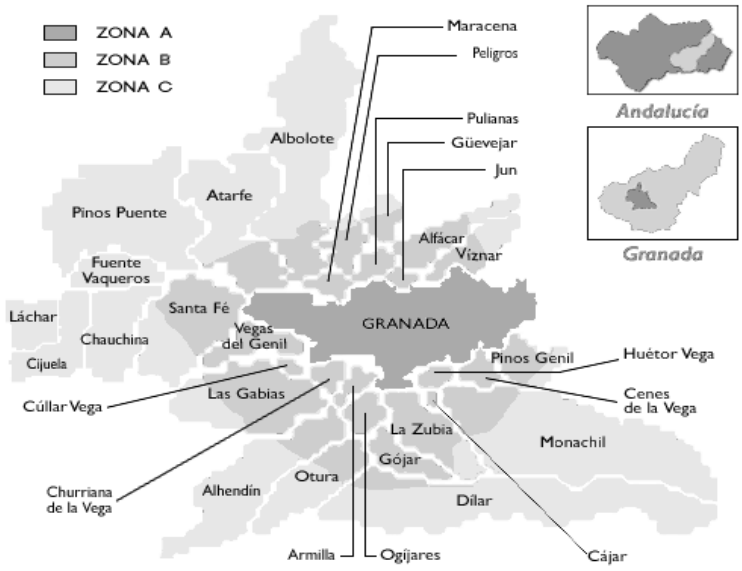


Figura 5.6: Área metropolitana de Granada

Localidad	$\mu_{\textit{Área metropolitana}}$	**Localidad**	$\mu_{\textit{Área metropolitana}}$
Albolote	0,70	Huétor Vega	0,75
Alfacar	0,65	Jun	0,75
Alhendin	0,55	La Zubia	0,70
Armilla	0,75	Láchar	0,50
Atarfe	0,68	Las Gabias	0,68
Cájar	0,75	Maracena	0,75
Cenes de la Vega	0,75	Monachil	0,73
Cijuela	0,50	Ogíjares	0,75
Cúllar Vega	0,75	Otura	0,68
Chauchina	0,50	Peligros	0,73
Churriana de la Vega	0,75	Pinos Genil	0,70
Dílar	0,73	Pinos Puente	0,50
Fuente Vaqueros	0,50	Pulianas	0,75
Gójar	0,75	Santa Fe	0,63
Granada	1,00	Vegas de Genil	0,75
Güevejar	0,65	Víznar	0,63

Cuadro 5.1: Pertenencia de las localidades a el área metropolitana de Granada

a la zona B, un grado $0{,}75$; y los de la zona C un grado $0{,}5$. Sin embargo, existen localidades cuyo término municipal se encuentra entre varias zonas. En estos casos lo que hemos considerado es una ponderación según el tamaño del término que se encuentra en cada zona. De esta forma, la pertenencia de las localidades al área de influencia de la capital considerado es el recogido en la tabla 5.1.

La traducción de la dimensión presentada a la estructura multidimensional difusa propuesta sería

$$\textit{Lugar} = (\{\textit{C.P., Localidad, Área metropolitana, Provincia}\}, \leq_{Lugar}, \textit{C.P., Provincia})$$

Código	Grupo
Q01	Enf.infecciosas y parasitarias
Q02	Tumores
Q03	Enf. endocrinas, nutricionales, metabólicas e inmunitarias
Q04	Enf. de la sangre y órganos hematopoyeticos
Q05	Trastornos mentales
Q06	Enf. del sistema nervioso y los órganos de los sentidos
Q07	Enf. cardiovasculares
Q08	Enf. del aparato respiratorio
Q09	Enf. del aparato digestivo

Cuadro 5.2: Código y grupos según al O.M.S.

5.1.1.6. Dimensión *Causa*

Otra de las características de las intervenciones es la causa. Para ello utilizamos los código propuestos por la Organización Mundial de la Salud (O.M.S.)[4]. Esta dimensión contiene información muy interesante donde se podría poner de manifiesto muchos de los problemas que se presentan en los modelos multidimensionales clásicos. Lamentablemente, para poder modelarlo en toda su extensión se necesitaría la colaboración de médicos especialistas, a lo que no hemos tenido acceso. Por ello, esta dimensión ha sido modelado de forma simple, contemplando el código de la causa de la intervención y uno de los grandes grupos a los que pertenece (Tabla 5.2).

La estructura de la dimensión resultante es

$$Causa = (\{Código, Descripción, Todo\}, \leq_{Causa}, Código, Todo)$$

5.1.1.7. Medidas

La medida que se ha considerado ha sido el *número* de intervenciones que comparten los valores que definen los niveles base de los dimensiones.

[4]URL: http://www.who.int/es/

5.1.1.8. DataCubo

La estructura completa de este DataCubo sería

$$C_{Médico} = (\{Duración, Tiempo, Paciente, Material, Lugar, Causa\}, \{Número\} \bigcup \emptyset, \Omega, A),$$

donde A es la relación que asocia a cada hecho las coordenadas que lo caracterizan en el espacio que definen las dimensiones. A continuación presentaremos algunas consultas de ejemplo sobre esta estructura.

5.1.2. Consultas

En la Figura 5.7 puede verse el número de intervenciones dependiendo de si se producen en meses cálidos, templados o fríos. A la vista, parece que en los meses en los que más intervenciones se producen son en los templados, mientras que en los cálidos es en los que menos.

También podemos saber si dependiendo del grupo de edad la duración de las intervenciones varía. En la Figura 5.8 se puede ver el resultado de la consulta. A la vista del resultado, podemos ver que la mayor parte de las intervenciones se realizan sobre personas *adultas*. La distribución de las intervenciones según su duración es similar en los tres grupos de edades, salvo que en el caso de los *adultos* en el que la relación de número de intervenciones con duración*normal* respecto a las demás es sensiblemente superior.

La distribución de intervenciones según la duración y si los pacientes provienen de una una localidad perteneciente al área metropolitana de Granada viene recogida en la tabla 5.3). Para mostrar los hechos resultantes hemos utilizado el *resumen lingüístico*. Como puede verse, no existen grandes diferencias por la procedencia del paciente.

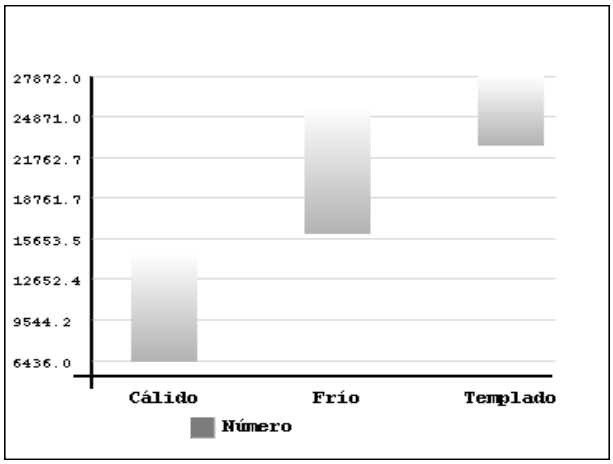


Figura 5.7: Intervenciones según temperatura de los meses

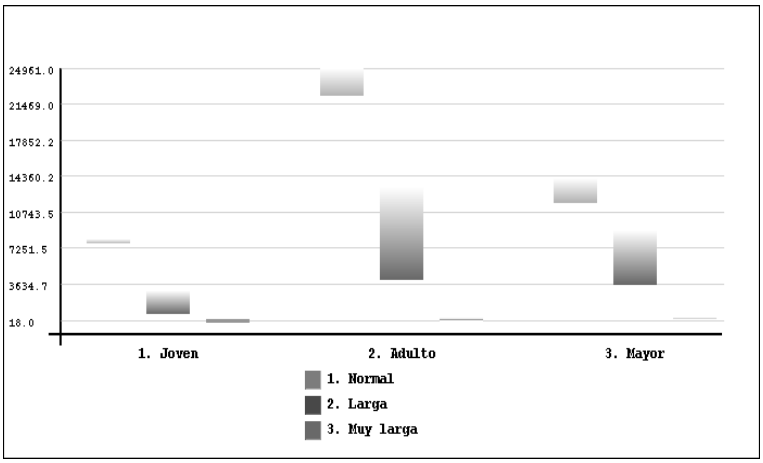


Figura 5.8: Intervenciones según grupos de edades y duración

Duración	Área metropolitana	Intervenciones
1. Normal	No	Mayor que 14727.0
1. Normal	Sí	Mayor que 13005.0
2. Larga	No	Mayor que 3233.0
2. Larga	Sí	Mayor que 2777.0
3. Muy larga	No	Mayor que 116.0
3. Muy larga	Sí	Mayor que 100.0

Cuadro 5.3: Intervenciones según la duración y la procedencia de los pacientes

5.2. DataCubo sobre datos contables

El ejemplo de aplicación de la técnica se ha realizado sobre la base de la información proporcionada por la empresa Axesor, S.A.[5]. Se consideraron 872 empresas pertenecientes a tres sectores de actividad, comercial, industrial y servicios, en función de la Clasificación Nacional de Actividades Económicas. Asimismo, en cada sector de actividad se diferenció entre empresas quebradas y no quebradas en función de la legislación vigente en el año 2001. Las variables económico-financieras que se han estudiado son la rentabilidad económica de explotación, el fondo de maniobra, y el coste del endeudamiento, y los valores que se muestran se refieren a los obtenidos durante los ejercicios 1998 a 2000.

La rentabilidad económica de explotación mide el rendimiento de las inversiones relacionadas con la actividad típica de la empresa y se calcula relacionando el resultado neto de la explotación con las inversiones de la explotación. El fondo de maniobra es una variable de tipo financiero indicativa del nivel de solvencia empresarial. Se calcula comparando la diferencia entre activo y pasivo circulantes con las ventas empresariales. El coste de endeudamiento mide el tipo medio de los recursos ajenos.

[5]URL: http://www.axesor.es

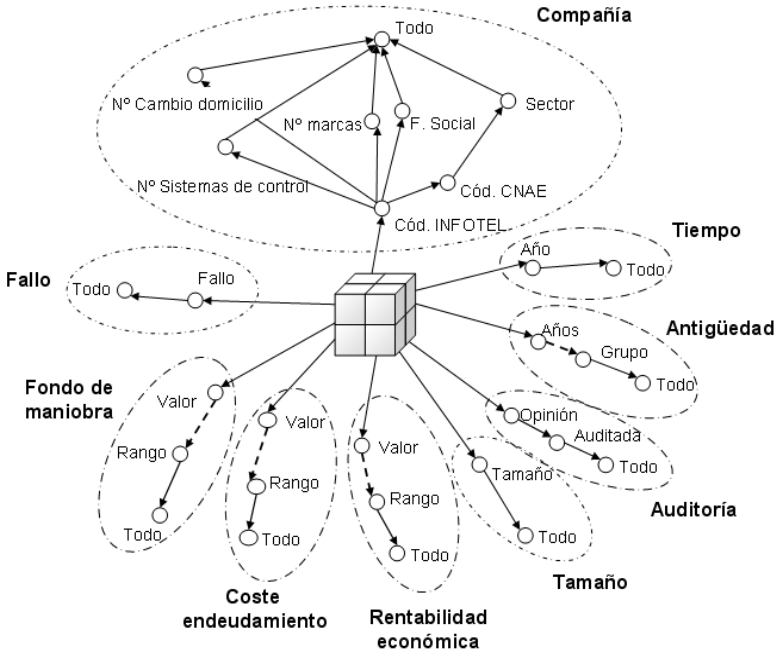


Figura 5.9: DataCubo sobre los datos contables

5.2.1. Estructura del DataCubo

En la Figura 5.9 se recoge la estructura del datacubo construido. Para realizarlo, hemos definido 9 dimensiones. En todos los casos se ha utilizado el mínimo y el máximo como t-norma y t-conorma para calcular las *relaciones de parentesco extendido*. Veamos detalladamente cada una de las dimensiones.

5.2.1.1. Dimension *Tiempo*

El nivel máximo de detalle de esta dimensión, es decir, el nivel base son los años. Como hemos comentado los valores considerados son 1998, 1999 y 2000. Sobre este nivel sólo se define el nivel que agrupa a todos los valores. Con esto, la estructura de la dimensión sería

$$Tiempo = (\{Año, Todo\}, \leq_{Tiempo}, Año, Todo),$$

donde $\leq_{Tiempo}$ define las relación jerárquica entre los niveles como

$$Año \leq_{Tiempo} Año,$$
$$Año \leq_{Tiempo} Todo,$$
$$Todo \leq_{Tiempo} Todo$$

5.2.1.2. Dimensión *Fallo*

El datacubo se ha definido para estudiar la relación de algunas variables financieras con respecto a la quiebra de la empresa. Una de nuestras dimensiones dará la información de esa quiebra. La denominaremos fallo, y el nivel base será *Fallo* con dos valores: *Sí*, cuando la empresa haya quebrado, y *No*, en caso contrario. Sobre este se definiría el nivel *Todo*.

La estructura resultante es:

$$Fallo = (\{Fallo, Todo\}, \leq_{Fallo}, Fallo, Todo).$$

donde $\leq_{Fallo}$ define las relación jerárquica entre los niveles como

$$Fallo \leq_{Fallo} Fallo,$$
$$Fallo \leq_{Fallo} Todo,$$
$$Todo \leq_{Fallo} Todo$$

Sector	Códigos CNAE incluidos
Industrial	151-410
Constructoras	450-454
Comerciales	501-527
Inmobiliarias	701-703
Servicios	651-990 (menos inmobiliarias)
Transportes	602-650

Cuadro 5.4: Sectores según códigos CNAE

5.2.1.3. Dimensión *Compañía*

Esta dimensión modela los datos referentes a las compañías. Hemos utilizado el código INFOTEL para definir el nivel base. Sobre éste, hemos utilizado el código CNAE para clasificar las empresas según el sector al que pertenecen. Para ello hemos utilizado los tres primeros números del código. Los sectores considerados (industrial, comercial, constructoras, inmobiliarias, servicios, transportes, y fuera de estos sectores) se han agrupado según estos códigos tal y como se recoge en la tabla 5.4. El resto de niveles contienen información referente al número de sistemas de control, número de cambios de dirección social, número de marcas registradas y la forma social de la empresa. La estructura resultante es

$$Compañía = (\{INFOTEL, CNAE, N^o \text{ sistemas de control}, N^o \text{ cambios}, N^o \text{ marcas}, \text{forma social}, \text{sector}, Todo\}, \leq_{Compañía}, INFOTEL, Todo),$$

5.2.1.4. Dimensión *Antigüedad*

El nivel base de esta dimensión clasifica a las empresas según el número de años que lleva la empresa en funcionamiento. Sobre este nivel definimos una agrupación de los años para caracterizar a las empresas según sean *muy jóvenes*,

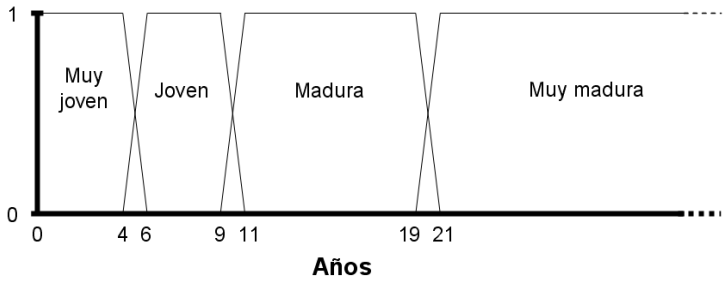


Figura 5.10: Valores y agrupación del nivel *Grupo* en la dimensión *Antigüedad*

jóvenes, *maduras* o *muy maduras*. Los humanos manejamos estos conceptos no como intervalos con fronteras precisas, sino que entendemos que una empresa sea más o menos madura en su actividad. Por eso, hemos utilizado conjuntos difusos para definir la relación de estos conceptos con los años concretos. En la Figura 5.10 se recoge la agrupación y los valores para la *relación de parentesco* que hemos utilizado.

La estructura de la dimensión estaría formada por tres niveles (los dos comentados y el nivel *Todo* que agrupa a los anteriores):

$$Antigüedad = (\{Años, Grupo, Todo\}, \leq_{Antigüedad}, Años, Todo).$$

5.2.1.5. Dimensión *Rentabilidad económica*

Para definir esta dimensión, hemos utilizado los valores observados para la variable financiera rentabilidad económica. Sobre estos valores hemos definido un nuevo nivel (*Rango*) para agrupar los valores en cinco categorías para facilitar el análisis. Para el usuario es más intuitivo el uso de valores categóricos (p.e. medio, bajo, alto, etc.) que valores concretos (e.g. rentabilidad con valor 6.51) a la hora de realizar consultas. Normalmente esta agrupación se realiza definiendo in-

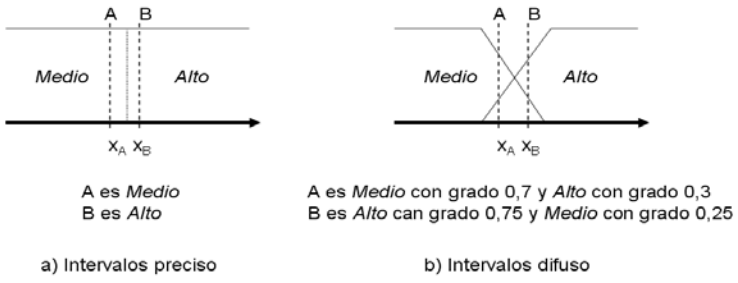


Figura 5.11: Problema de bordes precisos con valores muy cercanos

tervalos sobre los valores. Si utilizamos intervalos precisos se nos puede presentar un problema: dos valores muy cercanos pertenezcan a intervalos distintos (Figura 5.11). Este problema se reduce si suavizamos los bordes de los intervalos.

Las cinco categorías que hemos defino utilizando valores difusos son: *Muy bajo*, *Bajo*, *Medio*, *Alto* y *Muy alto*. Los valores agrupados se han definido en función del valor medio de la variables rentabilidad económica en el conjunto total de datos que disponíamos. De esta manera, los valores cercanos a este valor pertenecerán a la categoría *Medio*, y conforme se alejen de este valor entrarán en las categorías *Bajo* y *Alto*, si están por debajo de este valor medio o por encima, etc. Para los límites hemos utilizado los valores medio, máximo y mínimo de la variable. Cada intervalo [mínimo,media] y [media,máximo] se ha dividido en cinco intervalos de tamaño w_1 y w_2 respectivamente. Utilizando estos valores, se han definido las categorías como se recoge en la Figura 5.12.

La estructura de la dimensión será:

Rentabilidad Económica=($\{$*Valor, Rango, Todo*$\}$*,*$\leq_{RE}$*,Valor,Todo),*

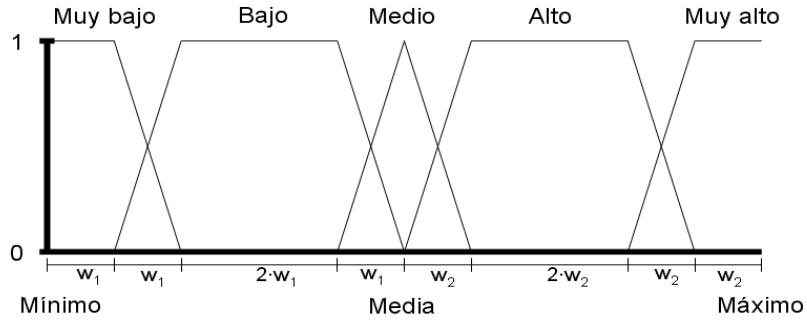


Figura 5.12: Valores y agrupación de los niveles *Rango*

5.2.1.6. Dimensión *Fondo de maniobra*

Esta dimensión se ha modelado igual que la anterior. El nivel base corresponde a los valores de las muestras y sobre éstos se han definido cinco categorías de forma difusa. Esta división se ha realizado mediante el mismo esquema de la *Rentabilidad económica*. Como es este caso, la estructura resultantes es:

Fondo de maniobra=($\{$*Valor, Rango, Todo*$\}$*,*$\leq_{FM}$*,Valor,Todo),*

5.2.1.7. Dimensión *Coste de endeudamiento*

Se trata de otra variable financiera como las anteriores y se ha modelado de la misma forma. La estructura de esta dimensión será la misma:

Coste de endeudamiento=($\{$*Valor, Rango, Todo*$\}$*,*$\leq_{CE}$*,Valor,Todo),*

5.2.1.8. Dimensión *Tamaño*

Esta dimensión clasifica las empresas según sean pequeñas, medianas o grandes empresas. Se ha considerado como una dimensión diferente a *Compañía* dado que

algunas de las empresas consideradas han cambiado de consideración durante los tres años de estudio. La estructura de esta dimensión traducida a nuestro modelo multidimensional sería:

$$Tamaño=(\{Tamaño,\ Todo\},\leq_{Tamaño},Tamaño,Todo),$$

5.2.1.9. Dimensión *Auditoría*

Otra de las variables para estudiar la quiebra de las empresas es si han realizado o no auditoría y, el caso de haberla aplicado, el resultado de la misma.

El nivel base de esta dimensión estaría formada por el nivel *Opinión* formado por los valores *Favorable*, *Denegada*, *Salvedades* y *No realizada*, en el caso de que no se haya realizado. Sobre este nivel, se define *Auditada* con valores *Sí* y *No* para agrupar los valores según provengan de haber realizado la auditoría (los tres primeros valores) o no (valor *No realizada*). La estructura de la dimensión sería:

$$Auditoría=(\{Opinión,\ Auditada,\ Todo\},\leq_{Auditoría},Opinión,Todo),$$

5.2.1.10. Medidas

Las medidas consideradas para definir los hechos han sido la rentabilidad económica, fondo de maniobra, coste de endeudamiento y el número de empresas que comporten el valor de todos las dimensiones que los definen.

Como puede verse, hemos utilizado las variables financieras tanto para definir hechos como dimensiones. Esto se debe a que queremos poder realizar análisis sobre las relaciones entre ellas (p.e. rentabilidad económica media según la categoría de fondo de maniobra que presenten, etc.).

A todos los hechos se le ha asignado un valor α=1 dado que todos provienen de la misma fuente, que es de confianza.

5.2.1.11. DataCubo

Vistas las dimensiones y las medidas, podemos presentar la estructura del DataCubo definido. Eta sería:

$C_{Contable}$ = (\{*Tiempo, Fallo, Compañía, Rentabilidad económica, Fondo de maniobra, Coste endeudamiento, Antigüedad, Tamaño, Auditoría*\},
\{*Rentabilidad económica, Coste de endeudamiento, Fondo de maniobra, Número*\}$\bigcup \emptyset$,Ω,*A*),

donde A es la relación que asocia a cada hecho las coordenadas que lo caracterizan en el espacio que definen las dimensiones. A continuación presentaremos algunas consultas de ejemplo sobre esta estructura.

5.2.2. Consultas

En la Figura 5.13 puede verse un ejemplo de consulta sobre el datacubo construido. En esta gráfica se muestra el resultado de consultar el número de empresas según el sector al que pertenecen (se ha restringido a comercial, industrial y servicios) y el valor de fondo de maniobra que presenten (según las categorías comentadas).

Como puede verse, las categorías *medio* y *bajo* presentan una imprecisión mayor en el resultado. Este hechos se debe a que existe un número alto de empresas con valores de fondo de maniobra entre las dos categorías (tiene un valor algo menor que el valor medio). Si los valores se distribuyeran cerca de los dos bordes el resultado sería análogo, salvo que las dos categorías contiguas presentarían una imprecisión alta. En el caso de utilizar categoría precisas este hecho no se pondría de manifiesto. Se podrían dar dos posibles salidas: una de las categorías presentaría valores más alto que la otra (significaría que la mayoría caerían en esta categoría) o que ambas tengan valores similares (se distribuyen entre las dos categorías).

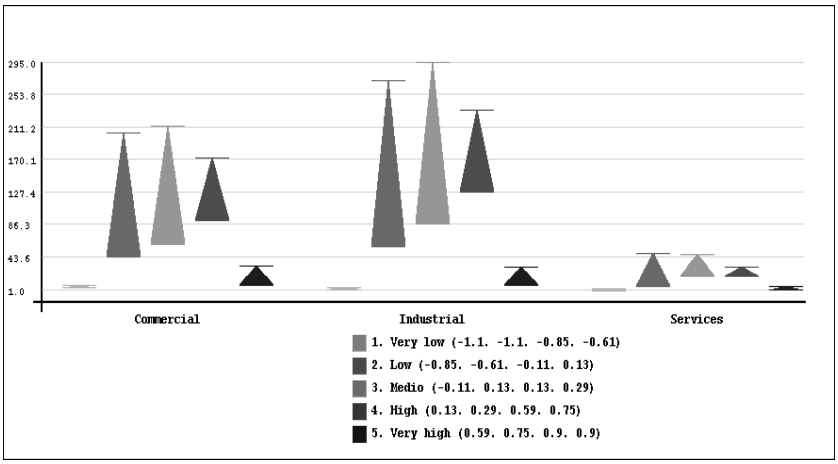


Figura 5.13: Número de empresas según fondo de maniobra y sector

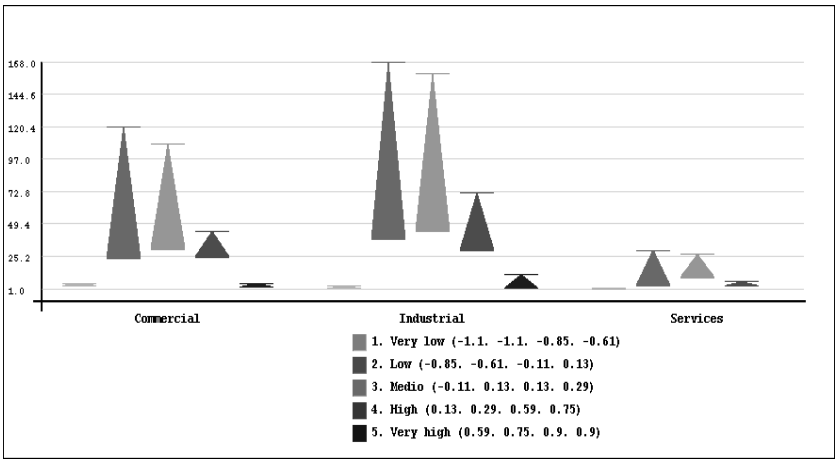


Figura 5.14: Número de empresas fallidas según fondo de maniobra y sector

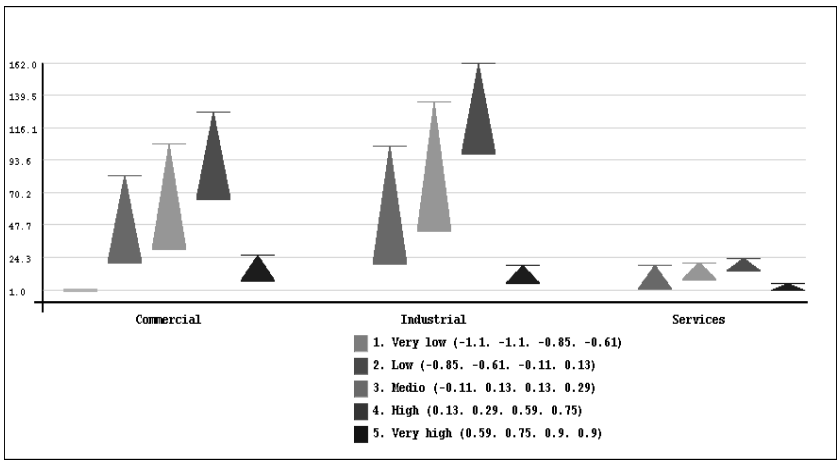


Figura 5.15: Número de empresas no fallidas según fondo de maniobra y sector

Refinemos el análisis viendo de forma independiente el fondo de maniobra para las empresas según hayan quebrado (Figura 5.14) o no (Figura 5.15). Considerando únicamente las empresas que presentan valores intermedios en el fondo de maniobra, es decir, valores *bajos*, *medios* o *altos*, se observa que: entre las que se encuentran en una situación de quiebra es menor el número de empresas cuanto mayor es el valor del indicador; entre las que tienen una buena situación financiera es mayor el número de empresas cuanto mejor es el indicador. Esta situación es lógica si se tiene en cuenta que, en principio, el valor del fondo de maniobra es indicativo del nivel de equilibrio financiero de la empresa.

5.3. DataCubo sobre datos del censo

El último datacubo difuso lo hemos construido utilizando los datos de un censo obtenidos del repositorio de datos para procesos de extracción de conocimien-

to de la Universidad de California[6]. Está pensada para predecir los ingresos de una persona partiendo del resto de los datos (p.e. edad, estudios, sexo, raza, etc.). Nosotros la utilizaremos para construir un datacubo añadiendo agrupaciones sobre algunos valores. En el caso de los ingresos, existen dos únicos valores: superior a 50.000 $ o inferiores a esta cantidad. Hubiera sido interesante contar con los valores y establecer categorías sobre los valores concretos.

5.3.1. Estructura del DataCubo

En la Figura 5.9 se recoge la estructura del datacubo construido. Para realizarlo, hemos definido 9 dimensiones. No hemos considerado los ingresos, dado que sólo contenían dos valores y nuestro objetivo no es predecir cuándo se dará cada valor.

En todos los casos se ha utilizado el mínimo y el máximo como t-norma y t-conorma para calcular las *relaciones de parentesco extendido*. Veamos detalladamente cada una de las dimensiones.

5.3.1.1. Dimensión *Persona*

En esta dimensión se modelan los principales datos de cada una de las personas censadas en cuanto a su edad, sexo y raza. El nivel base de la misma será cada combinación de estas tres variables. Sobre este, definimos tres niveles para agrupar los valores base según estas tres características:

- *Edad*: tendremos la edad de cada persona. Los posibles valores pueden pertenecer al intervalo [17,90], que son los considerados adultos dentro del rango de valores del censo. Estos los agruparemos como hemos hecho con el datacubo sobre datos médicos considerando *jóvenes*, *adultos* y *mayores*

[6]URL: http://kdd.ics.uci.edu/

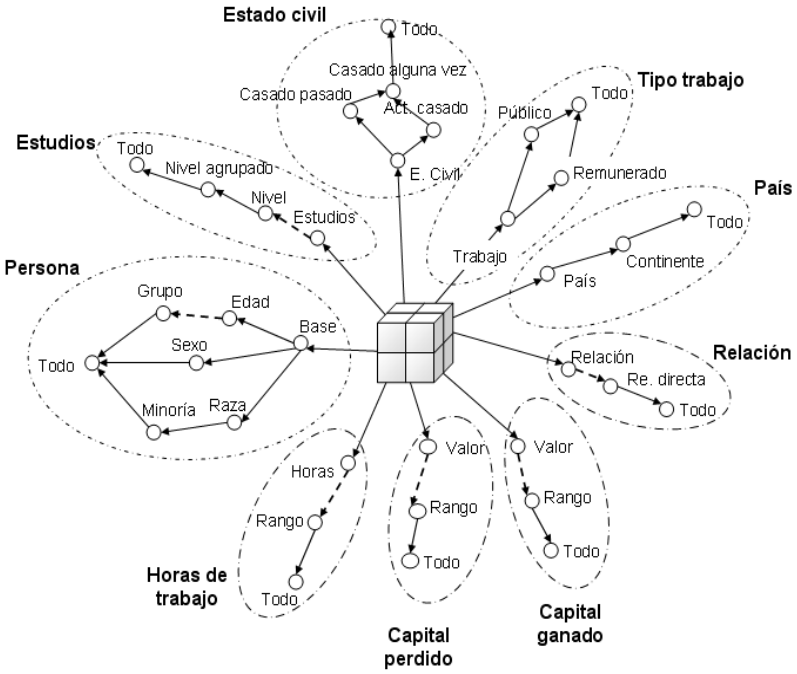


Figura 5.16: DataCubo sobre los datos del censo

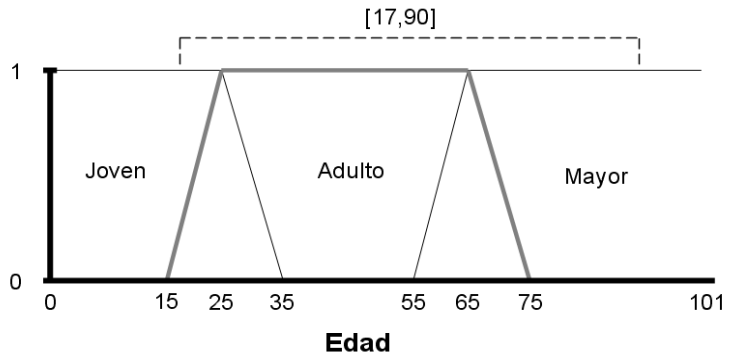


Figura 5.17: Etiquetas sobre las edades

según la misma agrupación, aunque restringido al intervalo definido (Figura 5.17).

- *Sexo*: sólo tendremos dos valores (mujer y hombre).
- *Raza*: las razas consideradas se han agrupado según se consideren *minoría* o no. Esta agrupación se ha definido de forma difusa. La raza negra supone un 10 % de la población según el censo. Otras razas (asiática e india) presentan porcentajes que no superan el 3 %. Por esto, considerarlas a todas al mismo nivel no parece lógico. La asignación que hemos hecho de la razas a minoría viene recogida en la tabla 5.5.

La estructura de la dimensión sería

$$\textit{Persona} = (\{\textit{Base, Edad, Grupo, Sexo, Raza, Minoría, Todo}\}, \leq_{Persona}, \textit{Base, Todo})$$

donde $\leq_{Persona}$ define las relación jerárquica entre los niveles.

Raza	$\mu_{\textbf{Minoría}}$
Amer-Indian-Eskimo	1
Asian-Pac-Islander	0,7
Black	0,5
Other	1
White	0

Cuadro 5.5: Pertenencia de las razas a valor *Sí* en el nivel *Minoría*

Nivel	**Estudios**
Básicos	1/Preschool, 1/1st-4th, 1/5th-6th, 1/7th-8th, 0'8/9th
Medios	0'2/9th, 1/10th, 1/11th, 1/12th, 1/HS-grad, 0'2/Assoc-voc, 0'2/Some-college
Altos	0'8/Some-college, 0'8/Assoc-voc, 1/Bachelors, 1/Assoc-acdm, 0'2/Prof-school
Avanzados	0'8/Prof-school, 1/Doctorate, 1/Master

Cuadro 5.6: Agrupación de los estudios según su nivel

5.3.1.2. Dimensión *Estudios*

Otro de las variables que consideramos es los máximos estudios que han cursado. Estos estudios (según el sistema estadounidense) los hemos agrupado según un nivel: *básicos*, *medios*, *altos* y *avanzados*. Este concepto de nivel lo solemos entender no forma precisa sino que lo solemos manejar con bordes imprecisos. Por eso, la relación con los estudios lo modelaremos mediante una relación difusa como la recogemos en la tabla 5.6.

Estos niveles los hemos agrupado a su vez en dos categorías: *básicos/medios* y *altos/avanzados*. Con esto, la estructura resultantes es

Estudios =($\{$*Estudios, Nivel, Nivel agrupado, Todo*$\}$,$\leq_{Estudios}$,*Estudios,Todo)*,

5.3.1.3. Dimensión *Estado civil*

Esta dimensión considera el estado civil de las personas. El nivel base lo que establece es su estado actual dentro de las categorías: divorciado (*Divorced*), casado (con dos categorías: *Married-AF-spouse* y *Married-civ-spouse*), casado con el cónyuge ausente (*Married-spouse-absent*), nunca casado (*Never-married*), separado (*Separated*) y viudo (*Widowed*).

Sobre este nivel se han establecido tres para agrupar el estado civil según ha evolucionado. A saber:

- *Casado en el pasado*: agrupa bajo la etiqueta *Sí* a los estado que implican que actualmente no está casado pero lo estuvo en el pasado.
- *Casado alguna vez*: en este caso se trata de agrupar a los que han estado o están casados y a los que nunca lo han estado.
- *Actualmente casado*: la etiqueta *Sí* agrupa a los que están actualmente casados.

La estructura de la dimensión será

$$\textit{Estado Civil} = (\{\textit{Estado Civil, Casado en el pasado, Casado alguna vez, Actualmente casado, Todo}\}, \leq_{EC}, \textit{Estado Civil, Todo}),$$

5.3.1.4. Dimensión *Tipo de trabajo*

Otra de las variables que contextualizan los hechos es el tipo de trabajo que realicen las personas censadas. Se consideran los tipos funcionario del gobierno central (*Federal-gov*), funcionario del gobierno local (*Local-gov*), funcionario del gobierno estatal (*State-gov*), privado (*Private*), autónomo (con valores *Self-emp-inc* y *Self-emp-not-inc*), no remunerado (*Without-pay*) y nunca ha trabajado (*Never-worked*).

Estos valores se han agrupado según son trabajos remunerados (nivel *Remunerado*) y si es funcionario o no (nivel *Público*). La dimensión queda

Tipo de trabajo =($\{$*Trabajo, Público, Remunerado, Todo*$\}$, $\leq_{TT}$, *Trabajo, Todo)*,

5.3.1.5. Dimensión *País*

El país de procedencia también es considerado dentro del datacubo. Existen 42 posibles nacionalidades, que hemos agrupado según el continente al que pertenecen los países correspondientes.

Estructura:

País =($\{$*País, Continente, Todo*$\}$, $\leq_{País}$, *País, Todo)*,

5.3.1.6. Dimensión *Relación*

Otra variable considerada es la relación existente. Se considera como valores marido (*Husband*), esposa (*Wife*), no casado (*Unmarried*), hijo propio (*Own-child*), otra relación (*Other-relative*) y no en familia (*Not-in-family*). Estos valores se han agrupado según lo directa que sea la relación. Para ello los hemos divido en si lo son (etiqueta *Sí*) o no (etiqueta *No*). Como no en todos los casos es clara la asignación, o al menos no todos los valores cumplen la condición en el mismo grado, y ni siquiera mediante una relación difusa es fácil de hacer. Por ello, hemos utilizado las etiquetas mostradas en la Figura 5.18 para construir la relación (Tabla 5.7).

Para calcular la *relación de parentesco extendida* utilizaremos el operador A_β^{QM} con $\beta = 1{,}0$ para la t-norma, $\beta = 0$ para la t-conorma, y el orden propuesto en [DVV98] en ambos casos. La estructura de la dimensión sería

Relación =($\{$*Relación, Relación directa, Todo*$\}$, $\leq_{Relación}$, *Relación, Todo)*,

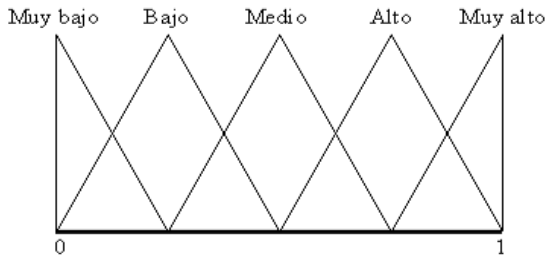


Figura 5.18: Etiquetas lingüísticas para el grado de la relación

Nivel	$\widetilde{\mu}_{\textbf{Relación directa}}$
Husband	Muy alta
Wife	Muy alta
Unmarried	Muy bajo
Own-child	Medio
Other-relative	Bajo
Not-in-family	Muy bajo

Cuadro 5.7: Relación directa

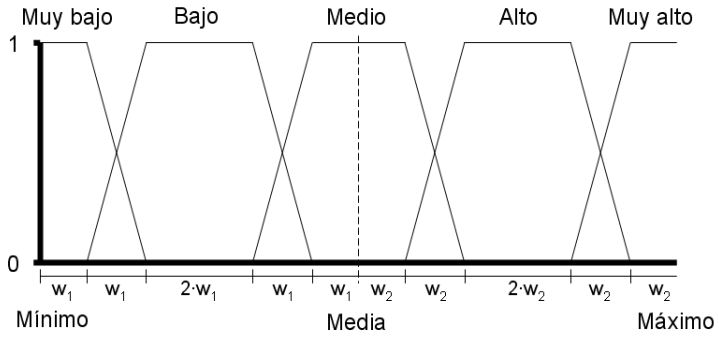


Figura 5.19: Etiquetas y grados de pertenencia para el nivel *Rango* en las dimensiones *Capital ganado* y *Capital perdido*

5.3.1.7. Dimensión *Capital ganado*

Otro variable de la base de datos representa el capital ganado. Esta variable es continua y presenta valores en el intervalo [0, 99999], y cuyo valor medio está entorno a 1079. Con esto datos, hemos agrupado los valores siguiendo una filosofía similar la la utilizada con los datos contables, pero utilizando en este caso las relaciones mostradas en la Figura 5.19). Para definirlas, hemos dividido los intervalos [mínimo, media] y [media, máximo] en seis sub-intervalos, de tamaño w_1 y w_2 respectivamente.

La estructura resultante para la dimensión será

Capital ganado =($\{$*Capital ganado, Rango, Todo*$\}$*,* $\leq_{CG}$*, Capital ganado, Todo),*

5.3.1.8. Dimensión *Capital perdido*

En este caso da el valor del capital perdido. Los valores se mueven en el intervalo [0, 4356], con media $87{,}5$. En esta dimensión hemos realizado una agru-

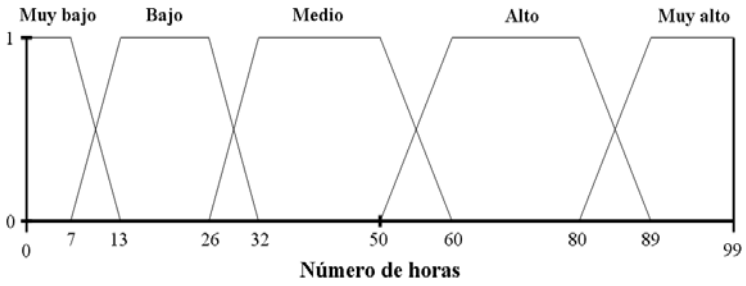


Figura 5.20: Etiquetas y grados de pertenencia para el nivel *Rango* en la dimensión *Horas de trabajo*

pación idéntica a la dimensión *Capital ganado*. Con esto, la estructura resultante sería

Capital perdido =($\{$*Capital perdido, Rango, Todo*$\}$, $\leq_{CP}$, *Capital perdido, Todo)*,

5.3.1.9. Dimensión *Horas de trabajo*

Esta dimensión se define sobre el número de horas semanales de trabajo. Los valores de esta variable están en el rango [0,99], con moda 40. Las hemos agrupado según se considere un número *muy bajo*, *bajo*, *medio alto* o *muy alto* de horas trabajadas según se muestra en la Figura 5.20.

Considerando este nivel, la estructura de la dimensión es

Horas de trabajo =($\{$*Horas, Rango, Todo*$\}$, $\leq_{HT}$, *Horas, Todo)*,

5.3.1.10. Medidas

Las medidas que se consideran en el DataCubo son dos. Por un lado tenemos el número de registros en el conjunto de datos inicial que comparten los valores de

los niveles básicos de las dimensiones definidas. Este hecho lo hemos denominado *Número*. En las base de datos original también existe una variable que establece el *peso* (variable *Weight*) de cada registro. Esta variable nos da para cada registro el número de personas a las que correspondería esa combinación de valores. Este valor también lo consideraremos. Para ello tendremos un hecho llamado *Peso* que tendrá la representatividad de la combinación de valores de los niveles base.

5.3.1.11. DataCubo

La estructura completa del DataCubo será

C_{Censo} = (\{*Persona, Estudios, Estado civil, Tipo de trabajo, País, Relación, Capital ganado, Capital perdido, Horas de trabajo*\}, \{*Peso, Número*\}$\bigcup \emptyset$,Ω,*A*),

donde A es la relación que asocia a cada hecho las coordenadas que lo caracterizan en el espacio que definen las dimensiones. A continuación presentaremos algunas consultas de ejemplo sobre esta estructura.

5.3.2. Consultas

Como ejemplo de consultas vamos a estudiar si influye o no ser funcionario para las horas que se trabaja. Para ello vamos a consultar el número de personas que hay en cada rango de horas de trabajo según si el trabajo es público o no. El resultado se muestra en la Figura 5.21.

A la vista del gráfico, lo primero que se observa es que existen más personas que tienen un trabajo privado que como funcionario, cosa que es normal. Por otra parte lo que se ve es que la mayoría de tratarse de un trabajador de sector privado existe una mayor relación entre los que trabajan más horas de la media que en el caso del sector público.

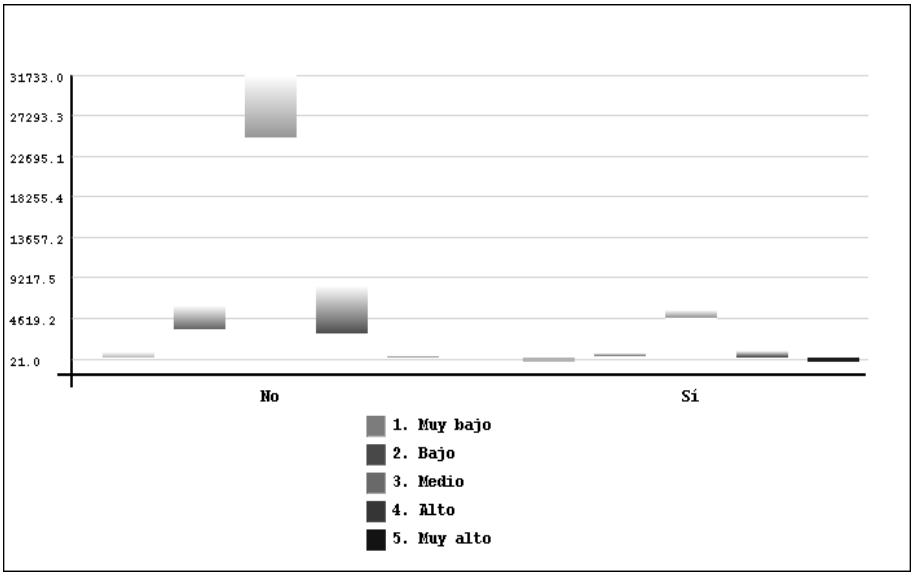


Figura 5.21: Número de personas que hay en cada rango de horas de trabajo según si el trabajo es en el sector publico o no

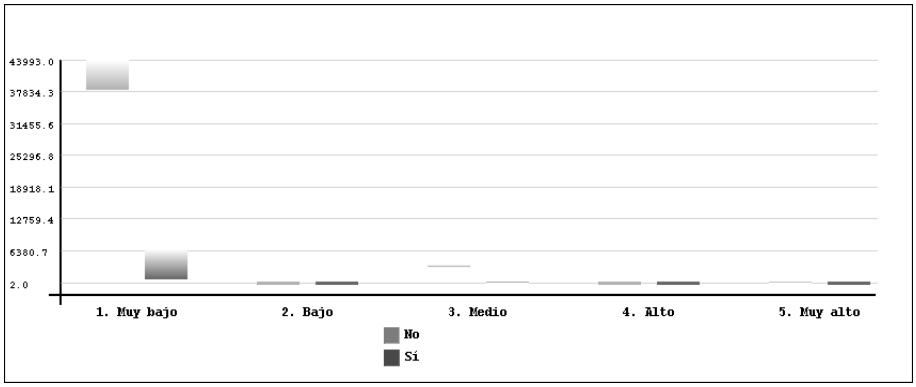


Figura 5.22: Número de personas que hay en cada rango de capital ganado según se consideren minoría o no

La otra pregunta que queremos responder es si ser minoría o no influye para el capital que se ha ganado. El resultado de la consulta se puede ver en la Figura 5.22.

Capítulo 6

Conclusiones y trabajos futuros

Una conclusión es el punto en el que se cansó de pensar

MÁXIMA DE MATZ
Ley de Murphy

En esta memoria hemos abordado el problema de tratar la imprecisión e incertidumbre en el modelo multidimensional de cara al usuario y a procesos de extracción de conocimientos. Resultado del mismo ha sido la formalización de un modelo multidimensional difuso con dos objetivos principales: permitir modelar información definida de forma imperfecta y ocultar la complejidad subyacente de cara al usuario final del sistema. Sobre este modelo también nos habíamos propuesto desarrollar una técnica de extracción de conocimiento basada en reglas de asociación. En este capítulo vamos a resumir los principales resultados obtenidos, así como los nuevos retos que han surgido.

- Se ha desarrollado un modelo multidimensional para manejar la imprecisión y la incertidumbre por medio de la lógica difusa. El modelo propuesto puede

trabajar con imprecisión e incertidumbre en diferentes aspectos:

- En los hechos se permite la representación y manejo de la imprecisión e incertidumbre, permitiendo trabajar con valores difusos simples o conjuntos difusos. De esta manera se puede integrar hechos provenientes de diversas fuentes, estableciendo la incertidumbre de la fuente o la imprecisión de los datos reflejados.
- En las dimensiones se permite la definición de múltiples jerarquías en las que la relaciones entre elementos se definen de forma imprecisa. La utilización de varios jerarquías en cada dimensión permite enriquecer los análisis permitiendo múltiples criterios de clasificación de los valores. Si añadimos la posibilidad de definir relaciones jerárquicas de forma imprecisa, da la posibilidad de incluir información definida de forma vaga o que presenten imprecisión. En algunos dominios complejos existen conceptos no claramente definidos que no pueden modelarse con relaciones clásicas. En estos casos las relaciones difusas nos permite modelarlas de forma adecuada.

  Estas relaciones también permiten modelar algunos conceptos de forma más cercana a como las entienden los usuarios finales (p.e. cualidad de ser joven, o ser mes cálido). De esta forma, los resultados a las consultas también vendrán caracterizadas por conceptos más entendibles para el usuario a interpretar los resultados.

  El modelo calcula las relaciones entre elementos de niveles no adyacentes de forma automática por lo que se facilita el diseño de los datacubos al no tener que establecer un valor para la relación de cada valor con el resto del dominio de la dimensión.
- El modelo facilita la integración de información proporcionada por expertos tanto en la incorporación de elementos no bien definidos co-

mo en las relaciones entre los mimos. En el primer caso, el modelo permite incluir conceptos definidos mediante conjuntos difusos.

En cuanto a las relaciones, el modelo permite la definición de las relaciones entre los elementos mediante la utilización de etiquetas lingüísticas (p.e. la relación entre 25 años y Joven es *alta*, el mes de Mayo es cálido con grado *medio*). Para ello se ha propuesto el operador A_{β}^{OM} para calcular las relaciones entre niveles no adyacentes. Este operador es una extensión del operador OWA propuesto por Yager ([Yag88]) para trabajar sobre números difusos. Este operador se ha definido de forma que su comportamiento como función lógica se controle mediante un único valor (β) según el cual se calculan los pesos necesarios (independientemente de la ariedad del operador requerido).

- Para que el modelo resulte operativo, hemos definido las operaciones habituales de los modelos multidimensionales para tratar con la imprecisión tanto en las jerarquías como lo hechos: roll-up, drill-down, dice, slice y pivot. Estas operaciones se han definido tanto para el caso de utilizar relaciones jerárquicas difusas y lingüísticas. Las operaciones propuestas son cerradas de forma que el resultado de aplicar cualquiera de ellas es un datacubo válido para aplicar nuevas operaciones. De esta manera se permite al usuario refinar los análisis sin tener que partir desde el datacubo original. La operación de reducción del nivel de detalle (roll-up) se ha definido de forma que no se pierda información en su aplicación, es decir, se pueda deshacer aplicando drill-down. Las propiedades de asociatividad de estas operaciones también se han estudiado. Para poder realizar el cambio de nivel de detalle, hemos adaptado la propuesta de Runsdensteiner y Bic ([RB89]) de operadores de agregación para poder ser utilizados en nuestro modelo. De esta forma, la propuesta es completamente funcional.

- La mayor parte de la complejidad resultante del uso de la lógica difusa se ha ocultado de cara al usuario. Se ha definido un modelo en dos capas de forma que las *vistas de usuario* den un resultado más intuitivo para el usuario. En este sentido, hemos adaptado una técnica de resumen difuso (*resumen lingüístico*) para mostrar al usuario expresiones lingüísticas en lugar de conjuntos difusos, más complicados de interpretar. Además, hemos definido dos método de representación gráfica de números difusos para construir gráficas y mostrar los resultados de las consultas.

De los modelos previamente existentes, el más desarrollado de cara a el manejo de la imprecisión es el modelo propuesto por Laurent ([Lau02, Lau03b]). Respecto a este mejora el modelado de las dimensiones (unificando todas las relaciones en una única relación jerárquica y permitiendo calcular la relación entre valores de niveles no adyacentes sin tener que proporcionar un valor para cada par al modelar), dando facilidades para incorporar la información proporcionada por expertos (utilizando etiquetas lingüísticas para las relaciones jerárquicas) y, principalmente, al ocultar la complejidad añadida mediante la utilización de *vistas de usuario*.

- Sobre este modelo hemos desarrollado una técnica de extracción de conocimiento basada en reglas de asociación (COGARE):

  - Trabaja sobre la estructura multidimensional difusa propuesta, de forma que puede utilizar conceptos difusos para la construcción de reglas. Así pues, es posible obtener reglas que incluyan información dada por expertos mediante conceptos y/o relaciones mal definidas.
  - La técnica extrae relaciones entre elementos que pueden estar definidos a diferentes niveles en sus dimensiones. Utiliza un técnica *bottom-up*

para intentar extraer la máxima información posible sin redundancias de forma eficiente. Si a un nivel dado no encuentra relaciones significativas entre los elementos, generaliza los valores para buscarla a niveles jerárquicos más altos.

- Se ha desarrollado pensando en reducir la complejidad del conjunto de reglas resultante de cara al usuario. Para ello, se han identificado dos factores que influyen en la complejidad resultante (el número de reglas y la abstracción de los elementos que definen las reglas) y medidas para controlarlas. Para reducirla, el algoritmo utiliza las jerarquías definidas en las dimensiones para aumentar la abstracción de los elementos y reducir el número de reglas, permitiendo sacrificar calidad para conseguirlo. La técnica se ha probado utilizando datos reales. En los experimentos se ha obtenido un reducción del número de reglas entorno al 43 % y de la complejidad alrededor del 39 %. Si no se permite la pérdida de calidad, en la mayoría casos incluso se ha conseguido mejorarla.

El único mecanismo anterior de extracción de reglas sobre un modelo multidimensional difuso era el propuesto por Kaya y Alhajj ([AK03, KA05]). Respecto a este, el mecanismo propuesto lo mejora en primer lugar debido a que trabaja sobre un modelo más potente a la hora de representar dominios complejos y/o con imprecisión. Además, nuestro método está orientado a la reducción de la complejidad del resultado, obteniendo un conjunto de reglas más inteligible de cara a el usuario. Ambos trabajan a múltiples niveles utilizando diferentes soportes para cada uno. El modelo de Kaya y Alhajj precisa que el usuario proporcione un umbral de soporte para cada nivel. Nuestro modelo propone un mecanismo de cálculo de los umbrales, de forma que el usuario sólo ha de proporcionar un único valor.

Como se ha comentado, el trabajo realizado ha abordado los objetivos perseguimos, dando respuesta a cada uno de ellos. Durante el desarrollo del mismo han aparecido nuevos retos y líneas de investigación para continuar con el trabajo realizado:

- Referente al modelo multidimensional existen dos grandes vertientes a seguir:
  - El modelo puede ayudar a la integración de información provenientes de múltiples fuentes. Sin embargo, esta tarea no es trivial. En este punto habría que profundizar para estudiar metodologías de integración de esquemas con incompatibilidades para obtener datacubos difusos.
  - En muchos dominios existen datos en forma de documentos: textos (p.e. historiales médicos), imágenes (p.e. radiografías), etc. Sería interesante poder considerar estos tipos de datos en las dimensiones para enriquecer los esquemas multidimensionales resultantes. En estos casos, habría que estudiar la modelización de estos datos, las relaciones con otros elementos dentro de las jerarquías y dar mecanismos para la realización de consultas en las que se establezcan condiciones sobre los mismos.
- Respecto a la técnica COGARE de extracción de reglas de asociación sobre la estructura:
  - De los experimentos se observa que el tiempo requerido para la generalización es elevado. Pensamos que se podría mejorar utilizando heurísticas distintas para la selección del elemento candidato para la generalización en cada paso o esquemas de generalización alternativos.
  - La técnica extrae relaciones entre elementos situados en diferentes dimensiones (relaciones inter-dimensionales). Una posible extensión

sería que considerara también elementos pertenecientes a la misma dimensión (relaciones intra-dimensionales).

- Sería interesante desarrollar otras técnicas de extracción de conocimiento sobre el modelo difuso para que el usuario pudiera tener diferentes visiones da la información implícita.
- En cualquier caso, las técnicas deberán adaptarse para trabajar con datos de otros tipos si el modelo multidimensional es extiendo para trabajar con ellos.

Apéndice A

F-Cube Factory

Es más fácil luchar por unos principios que vivir de acuerdo con ellos

LEY ADLER
Ley de Murphy

A.1. Introducción

Este apéndice está dedicado a presentar el sistema *F-Cube Factory* el cual implementa el modelo multidimensional propuesto así como el método de extracción de reglas COGARE. Este sistema se ha construido extendiendo la funcionalidad de un sistema OLAP para DataCubos difusos ([DÁMF02]). En la siguiente sección presentaremos la arquitectura del sistema así como sus principales características. Posteriormente veremos cómo trabajar con el sistema.

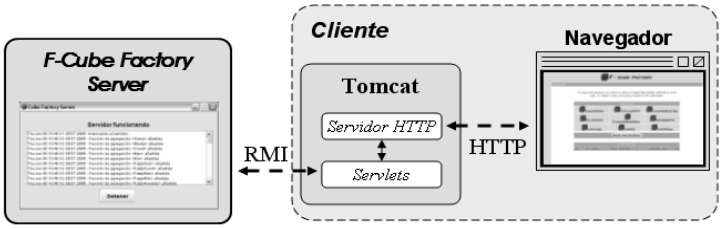


Figura A.1: Arquitectura del sistema

A.2. Arquitectura

El sistema *F-Cube Factory* se ha desarrollado siguiendo una arquitectura cliente/servidor (Figura A.1). Para su desarrollo se ha utilizado el lenguaje de programación Java debido a sus características:

- Es un lenguaje que produce aplicaciones independientes de la máquina y sistema operativo sobre el que se va a ejecutar. Actualmente existen implementaciones para los principales sistemas operativos sobre múltiples plataformas.
- Está muy integrado con el trabajo en redes de forma que tiene mecanismos para el desarrollo de aplicaciones cliente/servidor definidos en el estándar del lenguaje (p.e. RMI).
- Existe una interfaz de comunicación estándar con los sistemas de bases de datos (JDBC).
- Tanto las herramientas de desarrollo como las necesarias para su posterior implementación se pueden obtener de forma totalmente gratuita.

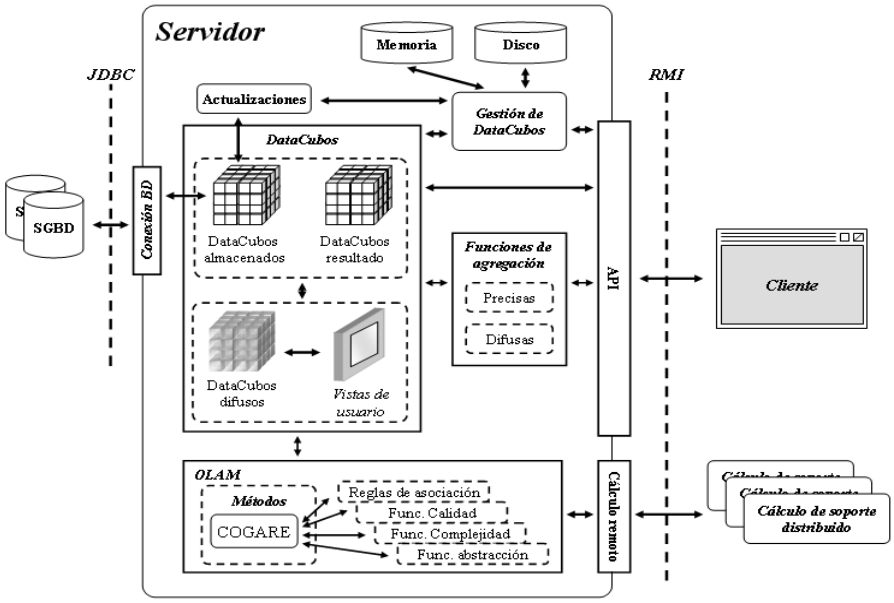


Figura A.2: Arquitectura del servidor

En las siguientes secciones comentaremos brevemente las principales características del servidor y del cliente.

A.2.1. Servidor

La arquitectura del servidor se refleja en la Figura A.2. A continuación presentaremos brevemente los principales módulos del servidor.

A.2.1.1. DataCubos

Una de los principales módulos del servidor es el encargado de los DataCubos. Actualmente el sistema soporta tres tipos distintos de DataCubos:

- *DataCubos almacenados*: estos DataCubos se definen utilizando datos almacenados en un sistema gestor de bases de datos (SGBD) con modelo de datos relacional. El esquema multidimensional obtenido se almacena a su vez en el mismo sistema. Como puede verse se trata de una estructura ROLAP para trabajar con DataCubos. La comunicación con los SGBD se realiza mediante JDBC, utilizando SQL estándar. El sistema debería poder trabajar con cualquier sistema relacional que cumpla dicho estándar y cuyo controlador JDBC satisfazca la especificación 2.0 del mismo. Actualmente se ha probado satisfactoriamente con Oracle 9i y 10g, así como PostgreSQL 7.4.

- *DataCubos resultados*: cualquier consulta realizada sobre los DataCubos anteriores dará como resultado un DataCubo que es gestionado mediante una estructura multidimensional pura (MOLAP). Estos DataCubos se han implementado completamente en Java, utilizándose la posibilidad de salvar objetos a disco para asegurar su persistencia.

  Conceptualmente estos DataCubos y los anteriores utilizan el mismo modelo multidimensional preciso, diferenciándose sólo la implementación del mismo.

- *DataCubos difusos*: utilizando un enfoque MOLAP se ha implementado el modelo multidimensional difuso propuesto, con la posibilidad de utilizar etiquetas lingüísticas en las relaciones de parentesco. Este tipo de DataCubos se construyen partiendo de un DataCubo preciso existente, sobre el cual posteriormente se definirán las estructuras difusas.

  Para mejorar la eficiencia en las consultas sobre estos DataCubos, al crearlos se precalculan las relaciones de parentesco extendido entre los elementos de un nivel y cada uno de los valores del nivel base. De esta manera, sólo se calcula una vez y en las consultas se mejora sensiblemente la eficiencia.

```
...
FUNCTION=Operator A (0.9)
TYPE=AND
CLASS=es.ugr.decsai.CubeFactory.dimension.aggregation.OperatorATypeAND
PARAM=BETA
VALUE=0.9
...
```

Cuadro A.1: Ejemplo de definición de una función de agregación para las relaciones de parentesco extendido en la configuración del servidor

Este módulo se ha desarrollado dando la posibilidad de añadir nuevas funciones para el cálculo de la relaciones de parentesco extendidas, tanto difusas como lingüísticas y vistas de usuario. Para ello, en el caso de las relaciones jerárquicas, se han definido dos interfaces (*AggregationANDType* y *AggregationORType* para MAM y MOM respectivamente) de tal forma que cualquier clase que las implemente podrá utilizarse con sólo definirla en la configuración del sistema (Tabla A.1), permitiendo establecer parámetros para su funcionamiento. Actualmente se han implementado algunas t-normas y t-conormas al igual que el operador A_{β}^{OM} desarrollado (Sección 2.1.6.3).

En el caso de las vistas de usuario, la interfaz es *UserViewMethod*, permitiendo añadir nuevos métodos de la misma forma que en el caso de las relaciones jerárquicas. En la implementación actual se han incluido la *media ponderada* y el *resumen lingüístico* como ejemplos.

Relacionado con este módulo están los que se encargan de la gestión de los DataCubos (*Gestión de DataCubos* y *Actualizaciones*) y conexión con la base de datos (*Conexión BD*). El primero de ellos se encarga del almacenamiento de los datos. De esta forma, cuando se requiere un DataCubo, se localiza en disco y se

carga en memoria (salvando otros a disco si no hay suficiente memoria).

El que se encarga de las actualizaciones implementa una política *off-line*, de tal forma que el usuario define una fecha y hora para la actualización y un periodo para volver a actualizar. Esta funcionalidad sólo está disponible para los DataCubos almacenados.

La conexión con la base de datos se abstrae mediante la utilización de un módulo (*Conexión BD*) que, mediante JDBC, se conecta con el sistema gestor de bases de datos relacional.

A.2.1.2. Funciones de agregación

En este módulo se gestionan las funciones de agregación necesarias para el cambio de nivel de detalle en los DataCubos.

El servidor está desarrollado pensando en facilitar la definición de nuevas funciones de agregación de forma simple. Para ello sólo es necesario implementar un interfaz Java llamada *OLAPAggregate*. De esta forma se han definido las funciones de agregación que acompañan al servidor. Para los modelos precisos (los DataCubos almacenados y resultado) se han implementado las habituales: suma, media, conteo, máximo y mínimo. Cabe destacar que en el caso de los DataCubos almacenados no se se utilizan las funciones aportadas por el SGBD sino que los datos se traen al sistema y es este el encargado de agregarlos. De esta manera, se puede utilizar cualquier funcionón definida en el servidor y que no exista en el SGBD.

En el caso difuso se han implementado las adaptaciones de las mismas funciones utilizando el enfoque de Rundensteiner y Bic que presentamos en la Sección 2.1.6.4.

```
♯ Métodos de OLAPMining
♯
♯ Estructura para cada método:
♯
♯ METHOD=<Nombre del método>
♯ CLASS=<Nombre de la clase>
♯ PARAMNAME=<Nombre parámetro 1>
♯ PARAMVALUE=<Valor del parámetro 1>
♯ ...
♯ PARAMNAME=<Nombre parámetro n>
♯ PARAMVALUE=<Valor del parámetro n>

[OLAPMINING]

METHOD=COGARE
CLASS=es.ugr.decsai.CubeFactory.DataMining.COGARE

[OLAPMINING]
```

Cuadro A.2: Definición de métodos de minería de datos

A.2.1.3. OLAM

El módulo OLAM se encarga de implementar la funcionalidad de métodos de minería de datos sobre los DataCubos. Actualmente se ha implementado el método COGARE presentado en esta memoria. Para ello, también se han implementado las clases necesarias para manejar *reglas de asociación*, conjuntos de elementos (*ItemSets*) y la diferentes funciones necesarias (funciones de calidad de reglas, complejidad de conjuntos de reglas y funciones de abstracción). Añadir nuevas funciones de este tipo es tan simple como en los casos anteriores (extendiendo la interfaces *QualityFunction*, *ComplexityFunction* y *Function* respectivamente).

Añadir nuevos métodos de extracción de conocimiento es igual de simple (interfaz *DataMiningProcess*). Posteriormente hay que añadir la definición a la configuración del servidor (Tabla A.2).

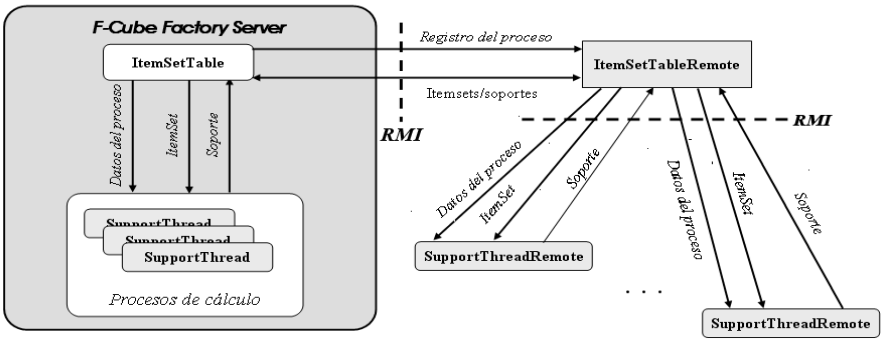


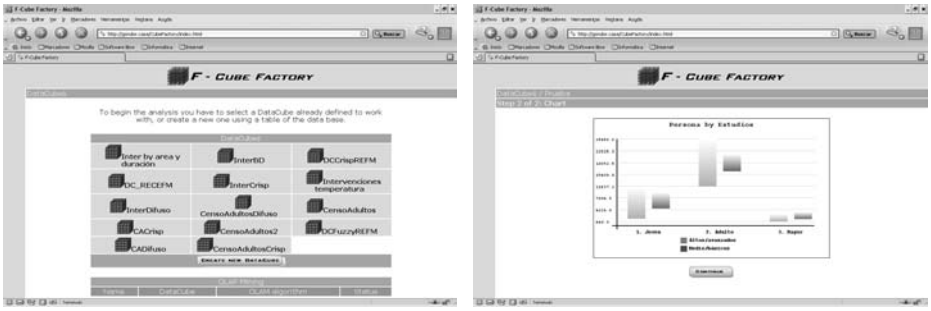
Figura A.3: Esquema del cálculo distribuido y paralelo

Dado que el proceso de cálculo de conjuntos de elementos frecuentes es bastante costoso en tiempo, se ha implementado el cálculo del soporte de los conjuntos de forma paralela y distribuida:

- *Paralela*: utilizando la posibilidad que brinda Java de utilizar hebras, se permite la utilización de múltiples procesos locales para el cálculo de los soportes. Para ello, se utiliza un esquema productor/consumidor.

- *Distribuido*: este esquema productor/consumidor local se ha extendido para que permita también la ejecución de procesos en equipos remotos. Para ello, se ha definido un servidor de cálculo distribuido utilizando RMI que es el que establece la comunicación entre los procesos y el servidor OLAP (Figura A.3). El cálculo paralelo también se ha integrado de manera que pueden coexitir ambos enfoques.

  En futuros desarrollos se pretende extender este esquema de cálculo paralelo/distribuido a otros procesos de minería de datos.



a) Pantalla principal b) Ejemplo de gráfica con valores difusos

Figura A.4: Pantallas del cliente WEB

A.3. Cliente

El cliente del sistema ha sido implementado para que sea ligero, es decir, no necesite grandes recursos para ejecutarse. Por esto, la opción que hemos adoptado es la de implementarlo en dos partes:

- El cliente se implementa como una aplicación web. Dado que la comunicación con el servidor se hace mediante RMI, hemos optado por implementarlo utilizando Servlet (Figura A.1).
- De esta forma, la interfaz de cara al usuario será mediante un navegador sin ningún otro requisito. El aspecto del cliente de cara al usuario se muestra en la Figura A.4.

Así pues, casi la totalidad de la funcionalidad del cliente está situada en el lado del servidor de páginas web que implante la aplicación. Este componente es el que implementa las gráficas de representación de números difusos que hemos presentado (Sección 3.6).

En la siguiente sección recogemos un pequeño manual de usuario sobre su utilización.

A.4. Manual de usuario

En esta sección vamos a presentar brevemente cómo realizar algunas de las tareas más habituales con el cliente. Comenzaremos presentando la creación y edición de DataCubos y la realización de consultas. Posteriormente veremos la aplicación de técnicas de minería de datos sobre los mismos.

A.4.1. Creación y edición de DataCubos

Comenzaremos viendo la creación de un DataCubo base y posteriormente cómo se puede ir refinando la estructura.

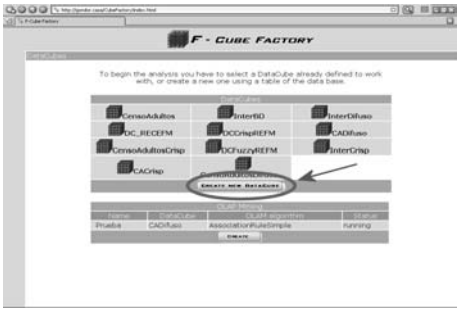
A.4.1.1. Creación de un DataCubo

Para la crear un DataCubo partimos de los datos almacenados en la base de datos relacional configurada. Comenzaremos seleccionando la opción correspondiente en la pantalla principal (Figura A.5.a).

El primer paso consisten en establecer el nombre del DataCubo y seleccionar la tabla o vista de la base de datos sobre la que se va a construir (Figura A.5.b).

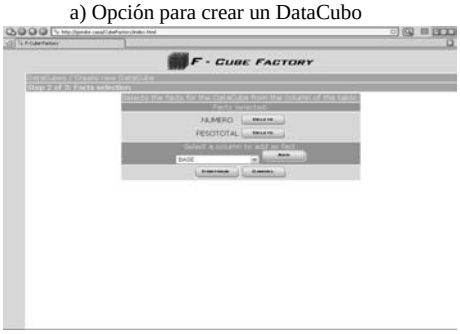
De las columnas definidas en la tabla o vista, el siguiente paso es seleccionar las que contienen los hechos para el nuevo DataCubo (Figura A.5.c). El último paso será seleccionar las dimensiones que caracterizarán a los hechos. Para cada una establecemos el nombre que tendrá, el nivel base que definirá el máximo nivel de detalle y la columna de la base de datos de la cual se poblará de datos (Figura A.5.d).

Si no hay ningún error (Figura A.5.e) nos mostrará un mensaje indicando que la estructura del DataCubo se ha creado con éxito (Figura A.5.f). Tras este paso, todas las estructuras en la base de datos estarán listas para cargar el DataCubo de datos. Sin embargo, la estructura resultante es muy simple, veremos como añadir más niveles.



a) Opción para crear un DataCubo

b) Nombre del DataCubo y tabla origen



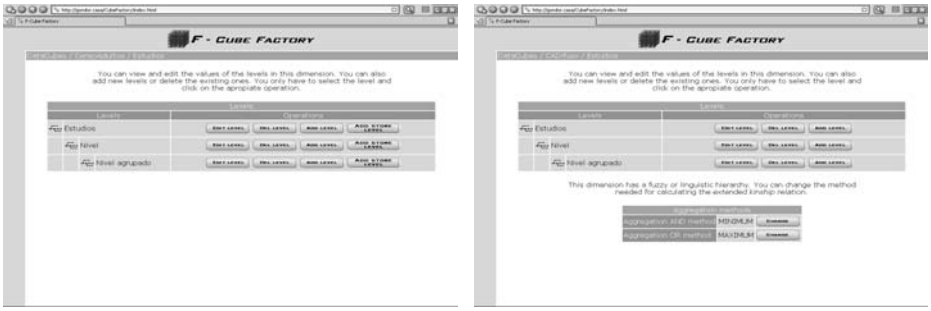
c) Selección de los hechos

d) Selección de dimensiones y niveles base

e) DataCubo creado con éxito

f) Datos del DataCubo

Figura A.5: Pantallas para la creación de un DataCubo



a) DataCubo almacenado b) DataCubo difuso

Figura A.6: Pantallas de edición de una dimensión

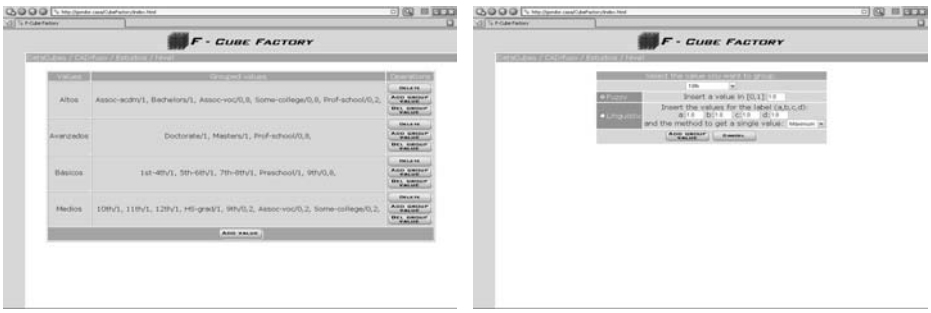
A.4.1.2. Creación de un DataCubo difuso

Para la creación de un DataCubo difuso partiremos de uno preciso ya definido. De esta forma obtendremos una estructura base que posteriormente iremos refinando incluyendo las relaciones difusas y/o lingüísticas que se deseen.

A.4.1.3. Edición de dimensiones y niveles

Dependiendo de si se trata de un DataCubo almacenado, resultado o difuso tendremos opciones distintas (Figura A.6). En el caso almacenado se tendrá la opción adicional de definir un nuevo nivel utilizando para ello una columna de la tabla origen del DataCubo. En el difuso la diferencia radica en que aparecerá la posibilidad de definir qué funciones tipo MAM y MOM se utilizan para calcular las relaciones de parentesco extendida.

El resto de las opciones son comunes y sirven para eliminar un nivel (y toda la jerarquía que cuelga de éste), la creación de un nuevo nivel sin origen en la base de datos o la edición del cualquiera de los definidos. Todos los niveles se pueden eliminar salvo el base.



a) Pantalla de edición de niveles

b) Añadir un nuevo valor agrupado en el caso difuso

Figura A.7: Pantallas de edición de niveles

A.4.1.4. Edición de niveles

Al editar un nivel tendremos la posibilidad de crear nuevos valores, eliminar alguno de los existentes o modificar las relaciones con respecto al nivel hijo (Figura A.7).

A la hora de añadir valores agrupados en el caso DataCubo difuso se puede utilizar tanto relaciones difusas como lingüísticas. En el primer caso se solicita al usuario un valor en el intervalo [0,1] para representar la relación de parentesco (Figura A.7.b). En el caso lingüístico se pide que se introduzcan los valores que definen el número difuso subyacente a la etiqueta a utilizar.

A.4.1.5. Definición de *Vistas de Usuario*

En la pantalla que muestra los datos de un DataCubo difuso existe la posibilidad de definir las vistas de usuario que se puedan utilizar a la hora de mostrar los hechos definidos (Figura A.8).

Figura A.8: Pantalla de definición de *vistas de usuario*

A.4.2. Consultas sobre DataCubos

Partiendo de la pantalla principal, el primer paso es seleccionar el DataCubo sobre el que vamos a aplicar la consulta (Figura A.4.a). El siguiente punto es seleccionar la operación de realizar consulta (Figura A.9.a). Tras establecer el nombre del DataCubo resultado (Figura A.9.b), seleccionamos los niveles a los cuales queremos realizar la consulta (Figura A.9.c).

Ya tenemos las dimensiones y niveles para definir el detalle de los hechos. Ahora tenemos que seleccionar los hechos y las funciones de agregación que utilizar para calcular los nuevos hechos (Figura A.9.d).

También podemos establecer condiciones sobre el espacio de partida a la hora de seleccionar los hechos (Figura A.9.e). Se pueden establecer condiciones en cualquier dimensión y cualquier nivel, se haya elegido para el nuevo DataCubo o no.

Este es el último paso, los siguiente que obtendremos será una pantalla con el resumen de la consulta realizada (Figura A.9.f). Tras realizar estos pasos, volviendo a la pantalla principal encontraremos un nuevo DataCubo con el nombre que le hemos dado a la consulta. Podremos ver el resultado o trabajar con el como si se tratara de cualquier otro DataCubo definido.

a) Información sobre DataCubo

b) Nombre del DataCubo resultado de la consulta

c) Selección de los niveles

d) Selección de hechos y funciones de agregación

e) Establecer condiciones sobre los valores

f) Resumen de la consulta realizada

Figura A.9: Pantallas para realizar una consulta

Figura A.10: Pantalla de selección de *vistas de usuario*

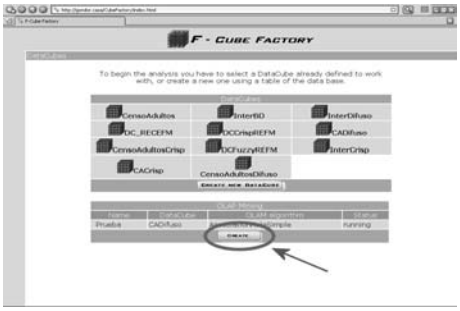
a) DataCubos almacenados y resultado b) DataCubos difusos

Figura A.11: Pantallas de opciones de las gráficas

A.4.3. Mostrar los hechos

Existen dos maneras de ver los hechos definidos: mediante una tabla (botón *Show records* o una gráfica (botón *Chart*). En el primer caso, si es un DataCubo difuso nos mostrará una pantalla donde nos permite escoger una vista de usuario que se haya definido para cada hecho o verlos sin procesar (Figura A.10).

En el caso de las gráficas, nos permitirá escoger entre diferentes representaciones y órdenes para mostrar los hechos (Figura A.11). Hay que destacar que los gráficas sólo pueden construirse si el DataCubo tiene 2 o menos dimensiones.



a) Opción parar crear un nuevo proceso desde la pantalla principal

b) Seleccionar el DataCubo y el método que aplicar

c) Establecer los parámetros del proceso

d) Proceso creado y ejecutándose

Figura A.12: Pantallas para lanzar un proceso de minería de datos

A.4.4. Procesos de minería de datos

En estos puntos vamos a mostrar brevemente el proceso que se ha de seguir para crear un proceso de minería de datos, ejecutarlo y, posteriormente, consultar su estado y los resultados obtenidos, en el caso de que haya terminado.

A.4.4.1. Crear un proceso

Para ejecutar un proceso de minería de datos sobre alguno de los DataCubos creados, se comienza desde la pantalla principal seleccionando la opción de crear

un proceso en la parte dedicada a los mismo (Figura A.12.a).

El primer paso a realizar es, aparte de establecer el identificativo para el proceso, seleccionar el DataCubo sobre el que aplicar el método (Figura A.12.b). La opción restante en esta pantalla será seleccionar el proceso de minería de datos a utilizar. Para ello, se muestran los procesos registrados en el servidor así como una breve descripción sobre su funcionamiento.

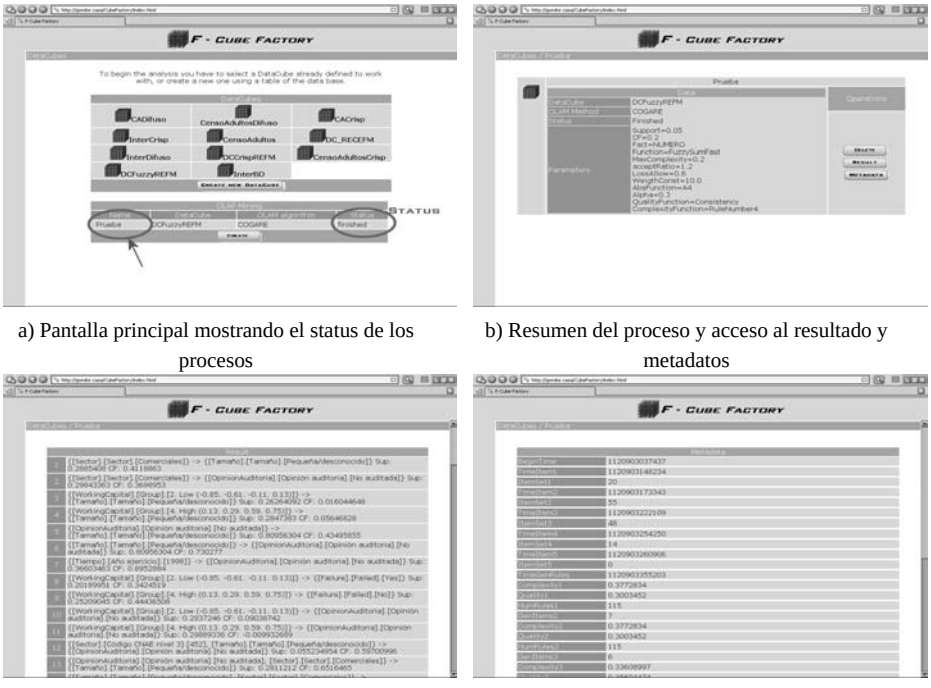
Cada proceso tiene unos parámetros diferentes para su aplicación. Así pues, el siguiente paso será establecer el valor de los mismos (Figura A.12.c). En este pantalla se muestran los nombres de los parámetros, los valores establecidos por defecto, si los hay, y una pequeña descripción de su utilidad. El cliente intenta identificar según el nombre de los parámetros si se trata de alguna selección de funciones u otros componentes definidos en el servidor. De esta manera para estos parámetros mostrará una lista desplegable para seleccionar uno de los componentes definidos.

Al pulsar en continuar, si no ha se produce ningún error se mostrará una pantalla informativa de la creación del proceso (Figura A.12.d) y el mismo comenzará a ejecutarse.

A.4.4.2. Consultar un proceso

En la pantalla principal se muestran los procesos de minería de datos existentes (Figura A.13.a), dando alguna información como su nombre, el DataCubo sobre el que se aplica, el proceso de OLAM utilizado, y su status (ejecutándose, finalizado o si se ha producido un error).

Pulsando sobre el nombre del proceso, el cliente nos muestra más información sobre el proceso (Figura A.13.b). En esta pantalla nos muestras los parámetros utilizados y, si se ha producido un error, el mensaje devuelto por el método para informar del mismo. Si no se ha producido ningún error y el proceso ha terminado, nos da la opción de consultar el resultado del proceso (Figura A.13.c) o los



a) Pantalla principal mostrando el status de los procesos

b) Resumen del proceso y acceso al resultado y metadatos

c) Resultados del proceso

d) Metadatos del proceso

Figura A.13: Pantallas para consultar un proceso de minería de datos

metadatos sobre el proceso (Figura A.13.d). También se da la opción para eliminar el proceso.

Si aún no ha terminado, sólo se muestra información sobre los parámetros.

Bibliografía

[AAD+96] Agarwal S., Agrawal R., Deshpande P., Gupta A., Naughton J. F., Ramakrishnan R. y Sarawagi S. (1996) On the computation of multidimensional aggregates. En *VLDB '96: Proceedings of the 22th International Conference on Very Large Data Bases*, páginas 506–521. Morgan Kaufmann Publishers Inc., San Francisco, CA, EE.UU.

[ABP04] Atzmueller M., Baumeister J. y Puppe F. (2004) Quality measures for semi-automatic learning of simple diagnostic rule bases. En *15th International Conference on Applications of Declarative Programming and Knowledge Management*.

[AC04] An A. y Cercone N. (Junio 2004) An empirical study on rule quality measures. *Lecture Notes in Computer Science* 1711: 482–491.

[AGS95] Agrawal R., Gupta A. y Sarawagi S. (Septiembre 1995) Modeling multidimensional databases. Informe técnico, IBM, IBM Almaden Research Center.

[AIS93] Agrawal R., Imielinksi T. y Swami A. (1993) Mining association rule between sets of items in large databases. En *Proceedings of ACM SIGMOD*, páginas 207–216.

[AK03] Alhajj R. y Kaya M. (2003) Integrating fuzziness into OLAP for multidimensional fuzzy association rules mining. En *ICDM '03: Proceedings of the Third IEEE International Conference on Data Mining*, página 469. IEEE Computer Society, Washington, DC, EE.UU.

[ALTK97] Albrecht J., Lehner W., Teschke M. y Kirsche T. (1997) Building a real data warehouse for market research. En *DEXA '97: Proceedings of the 8th International Workshop on Database and Expert Systems Applications*, página 651. IEEE Computer Society, Washington, DC, EE.UU.

[Ant65] Anthony R. (1965) *Planning and Control Systems: A Framework for Analisys*. Harvard Business School Division of Reseach Press.

[AS94] Agrawal R. y Sritkant R. (1994) Fast algorithms for mining association rules in large databases. En *Proceedings of 20th International Conference on Very large data Bases*, páginas 478–499.

[BBK+00] Berchtold S., Böhm C., Keim D. A., Kriegel H.-P. y Xu X. (2000) Optimal multidimensional query processing using tree striping. En *DaWaK 2000: Proceedings of the Second International Conference on Data Warehousing and Knowledge Discovery*, páginas 244–257. Springer-Verlag, Londres, UK.

[BD85] Bortolan G. y Degani R. (1985) A review of some methods for ranking fuzzy subsets. *Fuzzy Sets and Systems* 15: 1–19.

[BDH99] Barbara Dinter Carsten Sapia M. B. y Höfling G. (1999) OLAP markey and research: Initiating the cooperation. *Journal of Computer and Information Management* 2(3).

[Bli89] Blizard W. D. (1989) Multiset theroy. *Notre Dame Journal of Formal Logic* 30: 36–66.

[BMUT97] Brin S., Motwani R., Ullman J. D. y Tsur S. (1997) Dynamic itemset counting and implication rules for market basket data. En *Proceedings of the ACM SIGMOD International Conference on Management of Data*.

[Bru96] Bruha I. (1996) *Machine Learning and Statistics*, capítulo Quality of decision rules: Definitions and classification shcemes for multiple rules. John Wiley & Sons Inc.

[BSH98] Buzydlowski J. W., Song I.-Y. y Hassell L. (1998) A framework for object-oriented on-line analytic processing. En *DOLAP '98: Proceedings of the 1st ACM international workshop on Data warehousing and OLAP*, páginas 10–15. ACM Press, New York, NY, EE.UU.

[BSSV03] Blanco I., Sánchez D., Serrano J. M. y Vila M. A. (Enero 2003) A new proposal of aggregation functions: The linguistic summary. *Lecture Notes in Computer Science* 2715: 127–134.

[BT90] Brazdil P. y Torgo L. (1990) Knowledge adquisition via knowledge integration. En *Current Trends in Knowledge Acquistion*.

[CD97] Chaudhuri S. y Dayal U. (1997) An overview of data warehousing and OLAP technology. *SIGMOD Rec.* 26(1): 65–74.

[Chu04] Chua A. (2004) Knowledge management system architecture: a bridge between km consultants and technologists. *Information Management* 24: 87–98.

[CHY96] Chen M.-S., Han J. y Yu P. S. (1996) Data mining: An overview from a database perspective. *IEEE Trans. Knowl. Data Eng.* 8(6): 866–883.

[CM98] Chakrabarti K. y Mehrotra S. (1998) Dynamic granular locking approach to phantom protection in R-Trees. En *ICDE '98: Proceedings of the Fourteenth International Conference on Data Engineering*, páginas 446–454. IEEE Computer Society, Washington, DC, EE.UU.

[CM99] Chakrabarti K. y Mehrotra S. (1999) The hybrid tree: An index structure for high dimensional feature spaces. En *ICDE '99: Proceedings of the 15th International Conference on Data Engineering*, página 440. IEEE Computer Society, Washington, DC, EE.UU.

[Cod93] Codd E. (1993) Providing OLAP (On-line Analytical Processing) to user-analysts: An IT mandate. Informe técnico, E.F. Codd and Associates.

[Col96] Colliat G. (1996) OLAP, relational, and multidimensional database systems. *SIGMOD Rec.* 25(3): 64–69.

[CT97] Cabibbo L. y Torlone R. (1997) Querying multidimensional databases. En *Proceeding of the 6th Int. Workshop on databases programming languages (DBPL6)*. Estes Pork (EE.UU.).

[CT98] Cabibbo L. y Torlone R. (1998) A logical approach to multidimensional databases. En *EDBT '98: Proceedings of the 6th International Conference on Extending Database Technology*, páginas 183–197. Springer-Verlag, Londres, UK.

[DÁMF02] Donaire-Ávila A. C. y Molina-Fernández C. (2002) Cube Factory: Desarrollo de un nuevo sistema OLAP. Proyecto fin de carrera, E.T.S. Ingeniería Informática.

[db2] DB2 OLAP Server, http://www-306.ibm.com/software/data/db2/db2olap/.

[dbm] DBMiner http://www.dbminer.com.

[DF97] Dean P. y Famili A. (1997) Comparative performance of rule quality measures in an induction system. *Applied Intelligence* 7(2): 113–124.

[DMBSV03] Delgado M., Martín-Bautista M. J., Sánchez D. y Vila M. A. (Enero 2003) On a characterization of fuzzy bags. *Lecture Notes in Computer Science* 2715: 119–126.

[DMSV03] Delgado M., Marin N., Sanchez D. y Vila M. (2003) Fuzzy association rules: General model and applications. *IEEE transactions on Fuzzy Systems* 11(2): 214–225.

[DSV99] Delgado M., Sánchez D. y Vila M. (1999) Fuzzy cardinality based evaluation of quantified sentences. *International Journal of Aproximate Reasoning* .

[DT99] Datta A. y Thomas H. (1999) The cube data model: a conceptual model and algebra for on-line analytical processing in data warehouses. *Decision Support Systems* 27: 289–301.

[DVV93a] Delgado M., Verdegay J. y Vila M. (1993) Linguistic decision making models. *International Journal on Intelligent Systems* 7: 479–492.

[DVV93b] Delgado M., Verdegay J. y Vila M. (1993) On aggregation operations of linguistic labels. *International Journal of Intelligent Systems* 8: 351–370.

[DVV98] Delgado M., Verdegay J. y Vila M. (1998) A procedure for ranking fuzzy numbers using fuzzy relations. *Fuzzy Sets and Systems* 26(1): 49–62.

[DVW98] Delgado M., Vila M. y Wozman W. (1998) A fuzziness measure for fuzzy numbers: Applications. *Fuzzy Sets and Systems* 94: 205–216.

[Dyr96] Dyreson C. (1996) Information retrieval from an incomplete data cube. En *Proceeding of the 22nd Int. Conf. on VLDB*, páginas 532–543. Morgan Kaufman Publishers, Estambul (Turquía).

[EGY04] English J., George R. y Yazici A. (Septiembre 2004) A fuzzy spatio-temporal query model for facilating decision support in operational planning. En *Current Ussues in Data and Knowledge Engineering*, páginas 225–232. Varsovia.

[EKK00] Ester M., Kohlhammer J. y Kriegel H.-P. (2000) The DC-Tree: A fully dynamic index structure for data warehouses. En *ICDE '00: Proceedings of the 16th International Conference on Data Engineering*, página 379. IEEE Computer Society, Washington, DC, EE.UU.

[EKS97] Ester M., Kriegel H.-P. y Sander J. (1997) Spatial data mining: A database approach. En *SSD '97: Proceedings of the 5th International Symposium on Advances in Spatial Databases*, páginas 47–66. Springer-Verlag, Londres, UK.

[FH93] F. Herrera J. L. V. (1993) Linguistic assessments in group decision. En *Proc 1th European Congress on Fuzzy and Intelligent Technologies*, páginas 941–948.

[FH96] F. Herrera E. Herrera-Viedma J. L. V. (1996) Direct approach processes in group decision making using linguistic OWA operators. *Fuzzy Sets and Systems* 79: 165–176.

[FPSSU96] Fayyad U. M., Piatetsky-Shapiro G., Smyth P. y Uthurusamy R. (1996) *Advances in Knowledge Discovery and Data Mining*. AAAI/MIT Press.

[GCB+97] Gray J., Chaudhuri S., Bosworth A., Layman A., Reichart D. y Venkatrao M. (1997) Data Cube: A relational aggregation operator generalizing group-by, cross-tab, and sub-totals. *Data Mining and Konwledge Discovery* 1: 29–53.

[GR02] Grabmeier J. y Rudolph A. (2002) Techniques of cluster algorithms in data mining. *Data Mining and Knowledge Discovery* 6: 303–360.

[GSM71] Gorry G. y Scott Morton M. (1971) A framework for management information systems. *Sloan Management Review* 13: 50–70.

[Gut84] Guttman A. (1984) R-trees: a dynamic index structure for spatial searching. En *SIGMOD '84: Proceedings of the 1984 ACM SIGMOD international conference on Management of data*, páginas 47–57. ACM Press, New York, NY, EE.UU.

[Han97] Han J. (1997) OLAP mining: Integration of OLAP with data mining. En *IFIP Conf. on Data Semantics*, páginas 1–11.

[HCC93] Han J., Cai Y. y Cercone N. (1993) Data-driven discovery of quantitative rules in relational databases. *IEEE Transactions on Knowledge and Data Engineering* 5(1): 29–40.

[HCC+97] Han J., Chiang J. Y., Chee S., Chen J., Chen Q., Cheng S., Gong W., Kamber M., Koperski K., Liu G., Lu Y., Stefanovic N., Winstone L., Xia B. B., Zaiane O. R., Zhang S. y Zhu H. (1997) DBMiner: a system for data mining in relational databases and data warehouses. En *CASCON '97: Proceedings of the 1997 conference of the Centre for Advanced Studies on Collaborative research*, página 8. IBM Press.

[HCC98] Han J., Chee S. H. S. y Chiang J. Y. (1998) Issues for on-line analytical mining of data warehouses. En *Proc. 1998 SIGMOD Workshop on Research Issues on Data Mining*.

[HF95] Han J. y Fu Y. (1995) Discovery of multiple-level association rules from large databases. En *Proc. of 1995 Int. Conf. on Very Large Data Bases (VLDB'95)*, páginas 420–431.

[HGN00] Hipp J., Güntzer U. y Nakhaeizadeh G. (2000) Algorithms for association rule mining — a general survey and comparison. volumen 2, páginas 58–64. ACM Press, New York, NY, EE.UU.

[HLC03] Hong T.-P., Lin K.-Y. y Chien B.-C. (Enero 2003) Mining fuzzy multiple-level association rules from quantitative data. *Applied Intelligence* 18(1): 79–90.

[HPY00] Han J., Pei J. y Yin Y. (2000) Mining frequent patterns without candidate generation. En *Proceedings of the 2000 ACM SIGMOD international Conference on Managment of Data*.

[HS93] Houtsma M. y Swami A. (1993) Set-oriented mining of association rules in relational databases. Informe técnico, IBM Almaden Research Center.

[Hyp] Hyperion BI platform, http://www.hyperion.com.

[IH94] Inmon W. H. y Hackathorn R. D. (1994) *Using the Data Warehouse*. Wiley-QED Publication.

[Inm96] Inmon W. (1996) *Building the Data Warehouse*. Wiley Comnputer, New York, 2 edición.

[JKPT04] Jensen C., Kligys A., Pedersen T. y Timko I. (2004) Multimendional data modeling for location-based services. *The VLDB journal* 13: 1–21.

[JOL] JOLAP http://jcp.org/en/jsr/detail?id=069.

[KA05] Kaya M. y Alhajj R. (2005) Fuzzy OLAP association rules mining-based modular reinforcement learning approach for multiagent systems. *IEEE Transactions On Systems, Man, And Cybernetics* 35: 326–338.

[KF94] Kamel I. y Faloutsos C. (1994) Hilbert R-Tree: An improve R-tree using fractals. En *Proceedings of the 20th VLDB Conference*, páginas 500–509. Santiago, Chile.

[KHC97] Kamber M., Han J. y Chiang J. (1997) Metarule-guided mining of multi-dimensional association rules using data cubes. En *Knowledge Discovery and Data Mining*, páginas 207–210.

[Kim96] Kimball R. (1996) *The Data Warehouse Toolkit*. John Wiley & Sons, New York.

[KL03] Kaser O. y Lemire D. (2003) Attribute value reordering for efficient hybrid OLAP. En *DOLAP '03: Proceedings of the 6th ACM international workshop on Data warehousing and OLAP*, páginas 1–8. ACM Press, New York, NY, EE.UU.

[KLKL98] Kim D. W., Lee E. J., Kim M. H. y Lee Y. J. (1998) An efficient processing of range-MIN/MAX queries over data cube. *Inf. Sci.* 112(1-4): 223–237.

[Knu81] Knuth D. (1981) *The Art of Computer Programming*. Addison-Wesley, Reading.

[KS98] Katayama N. y Satoh S. (1998) SR-Tree: An index structure for nearest-neighbor searching of high-dimensional point data. *Systems and Computers in Japan* 29(6): 703–717.

[KSK96] K. S. Kim S. M. (1996) Application of fuzzy multisets to fuzzy database systems. En *Fuzzy Systems Symposion: Soft Computing in Intelligent Systems and Information Processing*, páginas 115–120.

[KY95] Klir G. J. y Yuan B. (1995) *Fuzzy Sets and Fuzzy Logic. Theory and Applications*. Prentice Hall PTR, Upper Saddle River, New Jersey.

[Lak76] Lake J. (1976) Sets, fuzzy sets, multisets and functions. *Journal of the Londres Math Society* 12(2): 232–326.

[Lau02] Laurent A. (2002) *Extraction de connaissances pertinentes à partir de baes de données multidimensionnelles*. Tesis doctoral, Laboratoire d'Informatique de Paris 6.

[Lau03a] Laurent A. (2003) A new approach for the generation of fuzzy summaries based on fuzzy multidimensional databases. *Intelligent Data Analysis* 7(2): 155–177.

[Lau03b] Laurent A. (2003) Querying fuzzy multidimensional databases: unary operators and their properties. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.* 11(Supplement): 31–45.

[LBMD+00] Laurent A., Bouchon-Meunier B., Doucet A., Gançarski S. y C. M. (2000) Fuzzy data mining from multidimensional databases. En *Euro-International Symposium of Computational Intelligence*, volumen 54, páginas 278–283.

[LBMD01] Laurent A., Bouchon-Meunier B. y Doucet A. (2001) Towards fuzzy-OLAP mining. En *Proc. Work. PKDD Database Support for KDD*, páginas 51–52.

[LC00] Lui C.-L. y Chung F.-L. (2000) Discovery of generalized association rules with multiple minimum supports. En *PKDD '00: Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery*, páginas 510–515. Springer-Verlag, Londres, UK.

[LFH00] Lu H., Feng L. y Han J. (2000) Beyond intratransaction association analysis: mining multidimensional intertransaction association rules. *ACM Trans. Inf. Syst.* 18(4): 423–454.

[LJF94] Lin K. I., Jagadish H. V. y Faloutsos C. (1994) The TV-Tree: an index structure for high-dimensional data. *The VLDB Journal* 3(4): 517–542.

[LW96] Li C. y Wang X. (1996) A data model for supporting on-line analytical processing. En *Proceeding of the 5th Int. Conf. in Information and Knoledge Management (CIKM)*.

[Man97] Mannila H. (1997) Methods and problems in data mining. En *ICDT '97: Proceedings of the 6th International Conference on Database Theory*, páginas 41–55. Springer-Verlag, Londres, UK.

[Mic90] Michalski R. (1990) Pattern recognition as rule-guided inductive inference. *IEEE Transactions on Patter Analysis and Machine Learning* .

[MK00] Muyeba M. K. y Keane J. A. (2000) Interestingness in attribute-oriented induction (AOI): Multiple-level rule generation. En *PKDD '00: Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery*, páginas 542–549. Springer-Verlag, Londres, UK.

[ML99] Martinez-Lopez L. (1999) *Un Nuevo Modelo de Representación de Información Lingüística Basado En 2-Tuplas Para la Agregación de Preferencias Lingüísticas*. Tesis doctoral, Universidad de Granada, Granada, España.

[NH94] Ng R. T. y Han J. (1994) Efficient and effective clustering methods for spatial data mining. En *VLDB '94: Proceedings of the 20th International Conference on Very Large Data Bases*, páginas 144–155. Morgan Kaufmann Publishers Inc., San Francisco, CA, EE.UU.

[NNT03] Niemi T., Nummenmaa J. y Thanisch P. (2003) Normalising OLAP cubes for controlling sparsity. *Data Knowl. Eng.* 46(3): 317–343.

[NTW00] Nguyen T. B., Tjoa A. M. y Wagner R. (2000) An object oriented multidimensional data model for OLAP. En *Proceedings of*

the 1st. International Conference on Web-Age Information Management (WAIN), number 1846 in LNCS, páginas 69–82. Springer.

[OLAa] OLAP Council http://www.olapcouncil.org.

[OLAb] OLAP4ALL http://www.olap4all.com.

[Ora] Oracle Express http://www.oracle.com.

[PCY95] Park J. S., Chen M. S. y Yu P. S. (Mayo 1995) An effective hash based algoritm for mining association rules. En *Proceedings of ACM SIGMOD*, páginas 175–186.

[Ped04] Pedrycz W. (2004) Associations and rules in data mining: a link analysis. *International Journal of Intelligent Systems* 19: 653–670.

[PJ98] Pedersen T. B. y Jensen C. S. (1998) Research issues in clinical data warehousing. En *SSDBM '98: Proceedings of the 10th International Conference on Scientific and Statistical Database Management*, páginas 43–52. IEEE Computer Society, Washington, DC, EE.UU.

[PJD01] Pedersen T., Jensen C. y Dyreson C. (2001) A foundation for capturing and querying complex multidimensional data. *Information Systems* 26: 383–483.

[Pow] Power OLAP http://www.paristech.com.

[PS91] Piatetskey-Shapiro G. (1991) *Knowledge Discovery in Databases*, capítulo Discovery, analysis, and presentation of strong rules, páginas 229–238. AAAI/MIT Press.

[PSF91] Piatetsky-Shapiro G. y Frawley W. J. (1991) *Knowledge Discovery in Databases*. AAAI/MIT Press.

[RB89] Rundensteiner E. y Bic L. (1989) Aggregates in posibilistic databases. En *Proceeding of the 15th Conf. in Very Large Databases (VLDB'89)*, páginas 287–295. Amsterdam (Holanda).

[SA95] Srikant R. y Agrawal R. (1995) Mining generalized association rules. En *VLDB '95: Proceedings of the 21th International Conference on Very Large Data Bases*, páginas 407–419. Morgan Kaufmann Publishers Inc., San Francisco, CA, EE.UU.

[Sal98] Salka C. (1998) Ending the ROLAP/MOLAP debate: usage based aggregation and flexible HOLAP. En *14th Int. Conf. on Data Engineering*, página 180.

[SAS] SAS OLAP Server, http://www.sas.com/.

[Sho] ShowCase Essbase http://www.spss.com/essbase/.

[SM99] Schaefer P. y Mitchell H. (1999) A generalized OWA operator. *International Journal of Intelligent Systems* 14: 123–143.

[SON95] Savasere A., Omiecinski E. y Navathe S. (1995) An efficient algorithm for mining association rules in large databases. En *Proceedings of the 21st Conference on Very Large Databases*.

[sql] Miscrosoft SQL Server, http://www.microsoft.com/sql/default.mspx.

[SS94] Sarawagi S. y Stonebraker M. (1994) Efficient organization of large multidimensional arrays. En *Proceedings of the Tenth International Conference on Data Engineering*, páginas 328–336. IEEE Computer Society, Washington, DC, EE.UU.

[SS98] Shen L. y Shen H. (Enero 1998) Mining flexible multiple-level association rules in all concept hierarchies (extended abstract). En

Database and Expert Systems Applications: 9th International Conference, DEXA'98, volumen 1460, páginas 786–796.

[Tho97] Thomsen E. (1997) *OLAP Solutions: building Multidimensional Information Systems*. John Wiley & Sons.

[TK00] Tan P.-N. y Kumar V. (2000) Interestingness measures for association patterns: a perspective. En *KDD-2000 Workshop on Postprocessing in Machine Learning and Data Mining*.

[TLHF03] Tung A. K. H., Lu H., Han J. y Feng L. (2003) Efficient mining of intertransaction association rules. *IEEE Transactions on Knowledge and Data Engineering* 15(1): 43–56.

[Tor97] Torra V. (1997) The wieghted OWA operator. *International Journal of Intelligent Systems* 12: 153–166.

[Tor02] Torra V. (2002) Learning wieghted for quasi-weighted menas. *IEEE Transaction on Fuzzy Systems* 10(5): 653–666.

[TP98] Trujillo J. y Palomar M. (1998) An object oriented approach to multidimensional database conceptual modeling (OOMD). En *DOLAP '98: Proceedings of the 1st ACM international workshop on Data warehousing and OLAP*, páginas 16–21. ACM Press, New York, NY, EE.UU.

[TS94] Theodoridis Y. y Sellis T. K. (1994) Optimization issues in R-tree construction (extended abstract). En *IGIS '94: Proceedings of the International Workshop on Advanced Information Systems*, páginas 270–273. Springer-Verlag, Londres, UK.

[VS99] Vassiliadis P. y Sellis T. (1999) A survey of logical models for OLAP databases. *SIGMOD Rec.* 28(4): 64–69.

[WB97] Wu M. y Buchmann A. (Marzo 1997) Research issues in data warehousing. En *BTW'97*, páginas 61–68. Springer, Ulm.

[XML] XMLA for Analysis http://www.xmla.org/.

[Yag86] Yager R. R. (1986) On the theory of bags. *International Journal of General Systems* 13: 23–37.

[Yag88] Yager R. (1988) On ordered weighted averaging aggregtion operators in multicriteria decison making. *IEEE Transactions on Systems, Man and Cybernetics* 18: 183–190.

[Yag93a] Yager R. R. (1993) MOM and MAM operators for aggregation. *Information Science* 69: 259–273.

[Yag93b] Yager R. R. (1993) Toward a unified approach to aggregation based upon MOM and MAM operators. En *Proceeding of World Conference on Neural Networks*, volumen 2, páginas 619–622. Portland (EE. UU.).

[Yag94] Yager R. R. (1994) Aggregation operators and fuzzy systems modeling. *Fuzzy Sets and Systems* 67: 129–145.

[Yag98] Yager R. (1998) Including importances in OWA aggregations using fuzzy systems modeling. *IEEE Transaction on Fuzzy Systems* 6(2): 286–294.

[Yen00] Yen S.-J. (Enero 2000) Mining generalized multiple-level association rules. En *Principles of Data Mining and Knowledge Discovery: 4th European Conference, PKDD 2000*, volumen 1910, páginas 679–684.

[Zad65] Zadeh L. (1965) Fuzzy sets. *Information and Control* 8: 338–353.

[Zad75a] Zadeh L. (1975) The concept of linguistic variable and its application to approximate reasoning, I. *Information Sciences* 8: 199–249.

[Zad75b] Zadeh L. (1975) The concept of linguistic variable and its application to approximate reasoning, II. *Information Sciences* 8: 301–357.

[Zad88] Zadeh L. (1988) Fuzzy logic. *Computer* 21(4): 83–93.

[ZDN97] Zhao Y., Deshpande P. M. y Naughton J. F. (1997) An array-based algorithm for simultaneous multidimensional aggregates. En *SIGMOD '97: Proceedings of the 1997 ACM SIGMOD international conference on Management of data*, páginas 159–170. ACM Press, New York, NY, EE.UU.

[Zhu98] Zhu H. (Diciembre 1998) *On-Line Analytical Mining of Association Rules*. Tesis doctoral, Simon Fraser University.

[ZPOL97] Zaki M. J., Parthasarathy S., Ogihara M. y Li W. (1997) New algorithms for fast discovery of assocaition rules. En *Proceedings of the 3rd International Conference on KDD and Data Mining*.