Tesis Doctoral

Un Modelo de Reglas de Asociación Temporales Basado en LifeSpans

Juan María ALE

Presentada a la Facultad de Ciencias Exactas de la Universidad Nacional de La Plata como parte de los requisitos para la obtención del título de
Doctor en Ciencias

Directores de Tesis
Dr. Gustavo Héctor ROSSI Dr. Alberto Oscar MENDELZON

La Plata, Octubre de 2003

Facultad de Ciencias Exactas
Universidad Nacional de La Plata – Argentina

Agradecimientos

Al Dr. Gustavo H. Rossi por su apoyo y orientación constante, sin cuya ayuda esta tesis no hubiera sido posible.

Al Dr. Alberto O. Mendelzon por su apoyo y guía, sus sugerencias y su paciente y minuciosa revisión de los trabajos que son la base de esta tesis.

A mis colegas Dr. Alejandro Vaisman, por sus comentarios y observaciones sobre la tesis, y Lic. Mauricio Minuto Espil por sus comentarios sobre mi investigación.

A mis tesistas que colaboraron con la implementación de los algoritmos y la experimentación, en particular, Cecilia Ruz, Adriana Colareda y Belén Blasquez..

A mi esposa María Rosa, mis hijos y nietos, por su amor, apoyo y comprensión, en las largas horas de escritura.

A mi madre, por su amor y aliento.

Resumen – Abstract – Keywords

Resumen

El objetivo de esta tesis es la definición de un modelo de regla de asociación con marco temporal y el desarrollo de técnicas y algoritmos para la extracción y análisis de información en grandes volúmenes de datos. El problema del descubrimiento de conocimiento en grandes bases de datos, en particular el representado por reglas de asociación, ha sido extensivamente estudiado en sus diversas formas. El punto de vista temporal en las reglas de asociación ha sido introducido desde dos vertientes: el enfoque de calendarios y el presentado en esta tesis, basado en la definición de lifespan o período de vida de los objetos analizados. Se describirá el modelo propuesto, se analizará su relación con otras propuestas y se presentarán algoritmos para la implementación de dicho modelo. En la parte experimental se presentarán resultados obtenidos, relacionados tanto con el desempeño computacional de los algoritmos propuestos, como los de utilidad práctica en el proceso de análisis final de resultados. A estos efectos se emplearán bases de datos sintéticas de amplia utilización y bases con datos reales en las que se pueden observar las ventajas del uso del modelo aquí presentado. Con el fin de extender las posibilidades de análisis se asocia a los objetos estudiados la noción de comportamiento temporal. Dicho comportamiento, considerado como una serie temporal, es pasible de ser estudiado eficientemente a través de su representación por medio de wavelets.

Abstract

This thesis is aimed in defining a temporal association rule model and developing algorithms and techniques for the efficient extraction and analysis of temporal rules, in large databases. The problem of knowledge discovery, in particular that expressed by association rules, has been extensively studied in diverse ways. The temporal perspective in association rules has been introduced from two alternates points of view: the calendar approach and the presented in this thesis, based in the concept of lifespan or period of life of the objects involved. Here we describe the proposed model, analyze the relationship among this model and other temporal models and present algorithms in order to implement it. On the experimental side, we present results related to performance as well as usefulness in the final analysis. To this end we use synthetic databases as well as data from the real world. In the latter is where it is possible appreciate the advantages of the proposed model. According with this aim, we developed the notion of temporal behavior, associated to every one of the sets of objects analyzed. We associate with the temporal behavior a temporal series, which can be efficiently studied using the wavelet transform representation.

Keywords: Data Mining, Association Rules, Temporal Data Mining, Temporal Rules, Temporal Association Rules, Wavelet Analysis, Similarity Queries.

Contenidos

Lista de Figuras

Lista de Tablas

Capítulo 1

Introducción

Innumerables organizaciones poseen bases de datos que contienen cantidades significativas de datos almacenados. El volumen de estos datos continúa creciendo en forma exponencial debido a que las características de la tecnología actual facilita la recolección y acumulación de datos. Esas características son el desarrollo masivo de sistemas de altos volúmenes de transacciones; el concepto de que la información corporativa constituye un "Activo Corporativo"[Uth96]; el éxito de la tecnología de bases de datos; la drástica reducción de los costos del hardware, en particular los medios de almacenamiento; la multiplicación de los medios de captura de datos, tales como los lectores de código de barras, tarjetas magnéticas, terminales de punto de venta, cajeros automáticos, terminales de autogestión, etc..
Un resultado son las grandes fuentes de datos tales como las Bases de Datos Científicas, los Data Warehouses y la Internet.

Se sabe que los datos son valiosos porque fueron recolectados originalmente para soportar actividades particulares de una organización, por lo que podrían existir relaciones valiosas aún no descubiertas en los datos. Es decir, existe la sospecha que puede haber "vetas" de información útil, oculta en las masas de datos sin analizar o poco analizados. El desafío es cómo reconocer esas relaciones.

Nuestras habilidades para recolectar datos son muy superiores a las facilidades con que contamos para analizarlos. En su trabajo de análisis de datos, un analista humano debe hacer frente a enormes cantidades de

información digital, por lo que se requieren nuevas técnicas y herramientas para poder superar esa sobrecarga de información.

Esas nuevas técnicas computacionales y herramientas constituyen la razón de ser de un área emergente conocida como **Data Mining o Descubrimiento del Conocimiento en Bases de Datos**[SSU96]. Si bien estos dos nombres se usan en forma intercambiable no son exactamente lo mismo.

El ***descubrimiento del conocimiento en bases de datos*** (Knowledge Discovery in Databases - KDD) se define como la extracción no trivial, desde los datos, de información implícita, previamente desconocida y potencialmente útil [FPS96].

El descubrimiento en bases de datos busca descubrir asociaciones notables, no reconocidas, entre ítems de datos en una base de datos existente [Man96]. El potencial del descubrimiento proviene del reconocimiento que contextos alternativos pueden revelar valiosa información adicional [FPS96a].

Data mining se refiere a la aplicación de algoritmos para la extracción de patrones utilizando los datos disponibles. Así, Data Mining es el componente del proceso KDD que tiene que ver fundamentalmente con los medios por los cuales se extraen y se enumeran los patrones, a partir de los datos.

Una definición para Data Mining es la siguiente: *es la exploración y análisis, por medios automáticos o semiautomáticos, de grandes cantidades de datos con el fin de descubrir reglas y patrones significativos.*

El minado de las bases de datos es un área de interés relativamente nueva. Se han desarrollado varios métodos, pero los resultados no han sido completamente satisfactorios. Esto se debe, en parte, a la imprecisión natural de las grandes masas de datos y a la complejidad inherente de las grandes colecciones de datos multiatributos.

El descubrimiento del conocimiento incluye la *evaluación* y posiblemente la *interpretación* de los patrones, para poder decidir qué constituye conocimiento y que no. También incluye la elección de esquemas de codificación, preprocesamiento, muestreo y proyecciones de los datos previo al paso de Data Mining.

El origen de Data Mining se puede encontrar, básicamente, en la intersección de áreas del conocimiento tales como la Estadística, la Inteligencia Artificial(Machine Learning) y las Bases de Datos [HKMT95]. Un motivo para el creciente interés en Data Mining es el hecho que muchas organizaciones, tanto públicas como privadas, con interés comercial o científico, han acumulado enormes cantidades de datos y, en general, se estima que del análisis de esos datos pueden surgir ventajas competitivas o novedosas soluciones a antiguos problemas.

El resto del capítulo está organizado de la siguiente manera. La sección 1.1 presenta los orígenes y conceptos básicos de Data Mining y KDD, incluyendo una enumeración de las tareas más conocidas. En la sección 1.2 se describe el problema y el enfoque para el descubrimiento de *reglas de asociación* de manera informal ya que en el capítulo 2 se tratará formalmente y en más detalle. La sección 1.3 plantea la motivación y la estructura de esta tesis.

1.1 Conceptos básicos. El Proceso de Descubrimiento del Conocimiento en Bases de Datos.

Comenzaremos esta sección analizando brevemente los orígenes de Data Mining, ya que ésta combina métodos y técnicas provenientes de áreas como machine learning, la estadística y las bases de datos.

Los métodos de machine learning son un componente central en Data Mining. La inducción de reglas y el aprendizaje de árboles de decisión son ejemplos típicos. Pero data mining *no es* machine learning[Man96]. Entre las diferencias encontramos las siguientes: machine learning se concentra básicamente en el paso inductivo o de aprendizaje del proceso de KDD; la segunda diferencia se refiere a los roles relativos de *conceptos* y *datos:* machine learning está más orientado a descubrir los mecanismos que producen los datos, mientras que en Data Mining el foco está puesto en los datos; otra diferencia tiene que ver con la complejidad del conocimiento obtenido, donde el interés de machine learning es el aprendizaje de cosas difíciles para los seres humanos; una última diferencia a mencionar es la cantidad de datos que se manejan: en machine learning los datos normalmente caben en memoria, mientras que en Data Mining, las bases de datos usualmente tienen un tamaño superior en varios órdenes de magnitud, aunque no solo en cantidad de objetos sino también en la cantidad de atributos.

En estadística, el término *data mining* ha sido usado en el pasado, en un sentido peyorativo, para designar un tipo de análisis de datos en el que no estaban claramente definidas las hipótesis. Sin embargo, las técnicas

conocidas como *análisis exploratorio de datos*(Exploratory Data Analysis – EDA) tienen mucho en común con las objetivos y métodos de Data Mining. Por otra parte, resulta evidente que existen técnicas estadísticas que no han sido todavía debidamente explotadas en Data Mining. Veamos ahora algunos aspectos que diferencian a Data Mining de la estadística: Data Mining maneja aspectos de complejidad computacional que son significativos, del mismo modo que los volúmenes de datos a considerar, y finalmente el uso de métodos de machine learning.

Un sistema de gestión de bases de datos(sgbd) permite el almacenamiento y consulta flexibles de grandes conjuntos de datos estructurados. Sin embargo, las consultas en Data Mining no son las usuales para este tipo de sistema. Por el contrario, una consulta en Data Mining es del tipo *hallar algo interesante en esta base de datos* y, evidentemente, no es posible expresarla en un lenguaje tipo SQL. El interés, dentro del área de bases de datos, por los problemas relacionados con el soporte a la toma de decisiones, ha generado una serie de soluciones ad-hoc para Data Mining que enfocaron, fundamentalmente, aspectos de escalabilidad. Data Mining o KDD, como una nueva aplicación de bases de datos, ha impulsado la investigación en técnicas de optimización para consultas complejas y grandes volúmenes de datos que involucran medios de almacenamiento terciario, lenguajes de consulta de muy alto nivel e interfases amigables que permitan a usuarios no expertos realizar consultas complejas ad-hoc [SSU96].

El objetivo del proceso de KDD es obtener conocimiento útil desde grandes colecciones de datos. El proceso de KDD es iterativo e interactivo, por lo que muchos pasos requieren decisiones del usuario. El siguiente es un bosquejo de los pasos básicos[Man96, FPS96]:

I. Comprender el dominio de aplicación: el usuario debe tener cierto tipo de conocimiento previo acerca del área de aplicación, para que el proceso tenga algún valor.

II. Preparar los datos: incluye subetapas tales como selección de las fuentes de datos, integración de fuentes heterogéneas, eliminación de inconsistencias, eliminación de ruido, definición de estrategias para el tratamiento de datos faltantes, etc..

III. Descubrir reglas o patrones(data mining): en este paso se descubren patrones interesantes de aparición frecuente en los datos. Es posible usar diversas técnicas y métodos. Se incluye aquí seleccionar la tarea específica de Data Mining, es decir, decidir si la meta del proceso de KDD es clasificación, agrupamiento, sumarización, etc.. Asimismo se deben seleccionar los algoritmos de Data Mining que se usarán para la búsqueda de patrones en los datos y los modelos y parámetros correspondientes.

IV. Postprocesamiento de los patrones descubiertos: incluye pasos como selección de patrones, visualización, etc., que deben permitir al usuario entender lo que se ha descubierto, visualizar los datos y los patrones, contrastar la información obtenida con la información previamente disponible.

V. Implementar los resultados: incorporación de este conocimiento en el sistema, o simplemente documentarlo e informarlo a las partes interesadas. Esto también incluye la verificación y resolución de conflictos potenciales con algún conocimiento previo.

Como se dijo, el proceso es iterativo. Así, los resultados del paso de Data Mining pueden evidenciar la necesidad de cambios a los datos en el paso

de preparación; el postprocesamiento de patrones puede inducir al usuario a buscar algún otro tipo de patrón, etc..

El paso III del proceso de KDD puede tener como finalidad alguna de las siguientes metas primarias de alto nivel de Data Mining: *predicción y descripción* [FPS96]. En el caso de la predicción, se usan algunas variables o campos en la base de datos para predecir valores, desconocidos o futuros, de otras variables. En la descripción, el objetivo es hallar patrones de comportamiento en los datos.

Las metas de la predicción y la descripción se alcanzan usando las siguientes tareas primarias de Data Mining [CHY96]:

i. Mining de *Reglas de Asociación*: es el descubrimiento de asociaciones entre objetos. Una regla de asociación es de la forma $A_1 \wedge \ldots \wedge A_j \rightarrow B$ y su significado es que el objeto B tiende a aparecer junto con los objetos $A_1, \ldots, A_k$ en la base de datos. Estas reglas pueden obtenerse a nivel básico, es decir, los objetos que aparecen explícitamente en la bd, y también en múltiples niveles, correspondientes a jerarquías o taxonomías definidas sobre esos objetos. El ejemplo más conocido es el de análisis de canasta en supermercados o *market basket analysis*. Otro ejemplo es el descubrimiento de regularidades en bases de datos de telecomunicaciones.

ii. *Clasificación*: es una función que mapea o clasifica un ítem de dato en una de varias clases predefinidas, conforme a un modelo de clasificación. Ese modelo se construye utilizando una base de datos de muestra denominada *training set*. Cada ítem de dato en el training set pertenece a

una determinada clase ya conocida. El objetivo de la clasificación es desarrollar un modelo para cada clase, en la forma de *reglas de clasificación.* Ejemplos de aplicación incluyen la clasificación de tendencias en mercados financieros, diagnóstico y tratamiento médicos, predicción de respuestas de clientes y la identificación automatizada de objetos de interés en grandes bases de datos de imágenes.

iii. *Clustering o Segmentación*: es la identificación de clases o clusters para un conjunto de objetos no clasificados, basado en sus atributos. Los objetos se agrupan de manera que se maximicen las similitudes intraclase y se minimicen las similitudes interclase, basados en algún criterio. Después de definidos los clusters, se rotulan los objetos con su clase correspondiente y se sumarizan las características comunes de los objetos en cada cluster, conformando así la descripción de la clase. Un ejemplo de aplicación es el descubrimiento de subpoblaciones homogéneas de consumidores.

iv. *Caracterización*: es la sumarización o abstracción de un conjunto de datos relevantes en una relación denominada *relación generalizada*, la que es utilizada luego para la extracción de *reglas características*. Dichas reglas representan las características de la base de datos denominada *target class*. Un ejemplo es la sumarización de un conjunto de síntomas de una determinada enfermedad por medio de un conjunto de reglas características.

v. *Discriminación*: es el descubrimiento de características o propiedades que distinguen la clase examinada, llamada clase *target*, de otras clases, llamadas clases *contrastantes*. Se descubre un conjunto de *reglas discriminantes* que sumarizan las características que distinguen la clase target de las clases contrastantes. Un ejemplo se presenta cuando se desea distinguir una enfermedad de otras: se define una regla discriminante que sumariza los síntomas que diferencian la enfermedad de las otras.

vi. Mining de *evolución*: es la detección y evaluación de regularidades en datos de evolución de ciertos objetos cuyo comportamiento cambia en el tiempo. Una aplicación es la determinación de los patrones de crecimiento de los precios de las acciones de un determinado sector.

vii. *Desviación*: es el descubrimiento y evaluación de los patrones de desviación de objetos en una base de datos con características temporales. El comportamiento esperado de los objetos es dado por el usuario o se computa en base a algún supuesto. Un ejemplo es el descubrimiento y evaluación de un conjunto de acciones cuyo comportamiento se desvía de la tendencia de la mayoría de las acciones durante un período de tiempo.

viii. *Regresión*: es una función un ítem de dato a una variable de predicción con valores reales. Ejemplos son la predicción de la demanda para un nuevo producto como función de los gastos de publicidad y la predicción de la cantidad de biomasa presente en una región dadas mediciones satelitales.

1.2 Reglas de Asociación.

El descubrimiento de reglas de asociación, introducido en [AIS93], ha sido identificado como un problema importante y uno de los de mayor éxito en las aplicaciones de Data Mining. Un ejemplo de este tipo de regla es que el 95% de los clientes que compran pan y leche también compran manteca. La motivación original para la búsqueda de estas reglas proviene del llamado *Market Basket Analysis*, el que usa la información acerca de qué compran los clientes, para poder identificarlos y conocer mejor sus hábitos de compra: la meta es hallar regularidades en el comportamiento de los clientes en términos de *combinaciones de productos que se compran en conjunto*. La utilidad de las reglas está, en este caso, en soportar decisiones relativas a precios de los productos, promociones, disposición de góndolas, etc..

Otros ejemplos que señalan la utilidad del descubrimiento de reglas de asociación son los siguientes:

* productos comprados con tarjeta de crédito, (alquiler de auto y habitación de hotel) dan una idea del próximo producto que comprará posiblemente el cliente.
* servicios opcionales comprados por clientes de telecomunicaciones permiten determinar cómo ofrecer esos servicios en conjunto para maximizar los ingresos.
* combinaciones poco usuales de reclamos de seguros pueden indicar fraude.
* Historias clínicas de pacientes pueden indicar complicaciones basadas en ciertas combinaciones de tratamientos.
*. Información sobre edad, hábito de fumar, niveles altos de colesterol

indican alto riesgo de enfermedades cardíacas.

En el caso del supermercado, los datos básicos son las transacciones de caja donde, para cada compra, se registra la fecha y los ítems o artículos comprados en esa transacción. Para cada ítem figura también la cantidad comprada por el cliente; pero en este trabajo haremos abstracción de ese atributo y consideraremos sólo el modelo binario: interesa si el cliente compró o no un determinado ítem. Existen, sin embargo, numerosas variantes de esta tarea, entre las que se cuentan extensiones para reglas cuantitativas [SA96a], reglas sobre taxonomías de ítems [SA95], reglas multiniveles [HF96], patrones secuenciales [AS95, MTV95, MT96, MTV97], etc.

Uno de los principales desafíos en esta tarea es que el número de ítems o artículos en un supermercado suele ser superior a 10.000. De esta manera, la cantidad de elecciones de posibles canastas resultaría de $2^{10.000}$, lo que es equivalente a aproximadamente $10^{3.000}$ potenciales canastas diferentes. Si bien en la práctica cada canasta tendrá, por ejemplo, no más de 30 ítems, el total de canastas resulta 10^{100}. Este es el problema de la explosión combinatoria de casos, con el que deberá enfrentarse cualquier algoritmo que pretenda descubrir reglas de asociación en este tipo de base de datos.

El conjunto de ítems se modeliza como $\mathbf{R} = \{A_1, \ldots, A_n\}$, $\mathbf{d}$ es una colección de subconjuntos de $\mathbf{R}$ denominada el *conjunto de transacciones*. Cada transacción $\mathbf{t}$ es un conjunto de ítems tal que $\mathbf{t} \subseteq \mathbf{R}$.

Dado $\boldsymbol{X} \subseteq \mathbf{R}$ un conjunto de ítems, si $|\boldsymbol{X}| = k$ se dice que $\boldsymbol{X}$ es un k-itemset o simplemente un itemset. Una transacción $\mathbf{t}$ soporta $\boldsymbol{X}$ si $\boldsymbol{X} \subseteq \mathbf{t}$.
Por convención, una regla de asociación es una implicación de la forma $\boldsymbol{X} \Rightarrow \boldsymbol{Y}$ y su significado es que la presencia del conjunto $\boldsymbol{X}$ implica la

presencia de otro conjunto Y, donde $X \subset \mathbf{R}$, $Y \subset \mathbf{R}$ y $X \cap Y = \varnothing$. La regla $X \Rightarrow Y$ se satisface en **d** con *confianza c* si c% de las transacciones en **d** que contienen X también contienen Y. La regla $X \Rightarrow Y$ tiene *soporte s* en **d** si s% de las transacciones en **d** contienen $X \cup Y$.

Dados un umbral de confianza θ y un umbral de soporte σ, el problema del descubrimiento de reglas de asociación consiste en hallar *todas* las reglas de asociación tal que su confianza y soporte no sean menores que los umbrales establecidos.

El descubrimiento de todas las reglas de asociación en un conjunto de transacciones **d** se puede realizar en dos fases [AIS93]:

Fase 1. Hallar todos los conjuntos de ítems(*itemsets*) $X \subseteq \mathbf{R}$ que sean frecuentes, es decir, que su frecuencia supere el soporte mínimo σ establecido.

Fase 2. Usar los itemsets frecuentes X para hallar las reglas: verificar para todo $Y \subset X$, con $Y \neq \varnothing$, si la regla $X \setminus Y \Rightarrow Y$ se satisface con confianza suficiente, es decir, supera la confianza mínima θ establecida.

Un *itemset frecuente* X es un conjunto de ítems tal que su soporte es igual o superior al mínimo fijado por el usuario. De la elección del nivel de soporte depende el número de itemsets frecuentes a ser hallados y la utilidad de dichos itemsets. El proceso de minado de los itemsets frecuentes tiene como finalidad hallar todos los itemsets con soporte mayor o igual al umbral o soporte mínimo σ.

Todas las reglas de asociación se obtienen a partir de los itemsets frecuentes X de tamaño $k \geq 2$, considerando todos los subconjuntos de X

que pueden aparecer como antecedente, o consecuente, de la regla, excepto X y $\varnothing$, y que tengan suficiente confianza.

1.3 Motivación y Estructura de la Tesis

El modelo de reglas de asociación presentado en la sección anterior considera, implícitamente, que los ítems involucrados existen durante el período de tiempo íntegro que corresponde a la recolección de datos. Es decir, cualquier ítem puede aparecer en cualquiera de las transacciones de la base de datos **d**. Lamentablemente este supuesto diverge de la realidad en varios aspectos: en distintos momentos aparecen, prácticamente en forma continua, nuevos ítems y, del mismo modo, otros ítems desaparecen, por diversos motivos. Por ejemplo, un producto que aparece en el mercado en un determinado momento y, luego de un tiempo, desaparece, tal como las "subnotebooks" entre los productos de computación. Otro ejemplo son síntomas nuevos que exhibe un paciente, la incorporación de un nuevo tratamiento, la desaparición de ciertos síntomas como consecuencia de un tratamiento, la suspensión de tratamientos, etc. En casos como este último, podemos advertir que el modelo tradicional resulta totalmente inadecuado, ya que la mayoría de los ítems, en este caso las diversas variables, rara vez se presentan todos juntos. Es decir, en casi todos los casos los ítems duran un determinado tiempo.

Otro caso son los ítems que aparecen juntos en determinadas ocasiones. Por ejemplo, en los primeros meses de Otoño se venden juntas vacunas antigripales y comprimidos de vitamina C Por lo que consideramos de gran significación la introducción de la variable tiempo en el modelo de reglas de asociación.

En esta tesis definimos un modelo de regla de asociación temporal [AR00], como una extensión al modelo estándar, mediante la inclusión de la dimensión tiempo, explícitamente representada por los timestamps de las transacciones de la base de datos. De acuerdo con esta visión, podemos hallar reglas de asociación restringidas al denominado período de vida de los ítems componentes de cada regla. Este concepto nos permite descubrir reglas interesantes y útiles que con el modelo estándar no podrían haber sido descubiertas. En el modelo no temporal, básicamente el problema surge de la definición de ítems o itemsets frecuentes y la forma de cálculo de su frecuencia: el soporte no debe considerar el tiempo previo o posterior a la manifestación de los ítems en la base de datos. Este modelo incluye nociones tales como *obsolescencia* que permite eliminar de la consideración ítems antiguos sobre los que ya se ha perdido interés y *soporte temporal*, el tiempo de vida de un ítem y su relación con el soporte temporal mínimo, umbral que los ítems deben superar para ser tomados en cuenta.

Este modelo difiere de los otros modelos de regla de asociación temporal en que no requiere el uso de calendarios ni ningún otro conocimiento previo.

Un ejemplo de una regla de asociación temporal, de acuerdo con este modelo básico, es el siguiente

Jugonaranjanatural⇒lecheconvitaminaC [2%, 87%], [12/6/00, 31/8/03]

donde la regla expresa que el par de productos "Jugo de naranja natural" y "Leche con vitamina C" aparecen en un 2% de las transacciones de la base de datos y en el 87% de las transacciones en que aparece "Jugo de naranja natural" también aparece "Leche con vitamina C". Observemos que esta

regla tiene asociado el intervalo [12/6/2000, 31/8/2003], el lifespan de ambos ítems en conjunto, como período de validez.

A continuación extendemos el modelo recién descripto adoptando el concepto amplio de lifespan, donde consideramos que éste puede estar integrado por un conjunto de subintervalos en los que el ítem o itemset resulta frecuente [AR02]. Esta extensión distingue nuestro modelo temporal de otros en el sentido que se cumple la propiedad de a priori, usada para la generación de los itemsets frecuentes. Otra extensión significativa resulta de adscribir a cada itemset su historia de apariciones en la base de datos. Dicha historia, básicamente un histograma, se representa por una serie temporal. Del análisis de esta serie podemos extraer información valiosa referida al comportamiento del itemset en el transcurso del tiempo, dentro de su lifespan. Para poder tratar más fácilmente las historias usamos la transformación DWT(Discrete Wavelet Transform) o transformada wavelet discreta que presenta, para nuestra finalidad una serie de ventajas sobre otras transformaciones más conocidas.

Para el descubrimiento de este último tipo de reglas, denominadas Reglas de Asociación Temporales Generalizadas, hemos diseñado un algoritmo eficiente que toma ventaja de las características temporales de los itemsets. Para este algoritmo contamos con dos implementaciones. La primera de ellas construida como una extensión del sistema WEKA de la Universidad de Waikato, New Zeland, realizada en Java, para bases de transacciones de pequeño porte, y otra realizada en PL/SQL sobre Oracle 9, para bases de datos más grandes. La experimentación de estos algoritmos fue realizada sobre una base de transacciones sintética y, en particular para la segunda implementación, sobre una base de datos real conteniendo datos de posiciones arancelarias aduaneras.

1.3.1 Principales contribuciones

La siguiente es una lista de contribuciones al campo de data mining, realizadas en esta tesis:

- Definición de un modelo de regla de asociación basado en el concepto de lifespan de los itemsets. Este modelo no requiere de ningún conocimiento previo, tal como los necesarios para la definición de intervalos de búsqueda, ciclos o calendarios. Entre los nociones introducidas para la elaboración de este modelo se cuentan la de *obsolescencia*, según la cual ítems antiguos, es decir, aquellos con un fin de período de vida anterior a un parámetro t_0, carecen de interés y pueden ser eliminados de la consideración. Otra noción es la de *soporte temporal*, de acuerdo con la cual sólo resultan de interés los ítems o itemsets cuyo lifespan supera un mínimo establecido τ. Finalmente la noción de *subintervalo frecuente*, lo que permite considerar períodos dentro del lifespan de un itemset dentro del cual éste resulta frecuente y, por lo tanto, de interés para el usuario.

- Definición de la noción de historia de un itemset. Este concepto permite representar el comportamiento temporal de cada itemset, en el sentido de variaciones de frecuencia en el tiempo, dentro de su lifespan. Asimismo, permite representar la base de transacciones íntegra de una manera compacta y eficiente. Esto facilita realizar sobre esta representación diversos tipos de análisis, incluso descubrir con suma facilidad reglas basadas en

otros enfoques. Se ha propuesto transformar la representación de la historia, considerada como una serie temporal, por medio de un análisis wavelet, habiendo argumentado que dicha transformada resulta totalmente adecuada al tipo de dato de nuestro problema.

- Un algoritmo eficiente para el descubrimiento de las reglas temporales, que aprovecha las características temporales intrínsecas de los itemsets. Este algoritmo, llamado Temporal Dynamic Itemset Counting reduce la cantidad de lecturas de la base de datos de transacciones, en relación a Apriori, y ha sido posible incorporarle numerosas optimizaciones, basadas en las características temporales de los itemsets.

- Extensiva experimentación realizada con datos reales y con datos sintéticos de amplio uso [AS94], a los que se les ha incorporado el tiempo de transacción, para lo cual definimos un modelo de transacción temporal. Hemos evaluado tiempos de ejecución y escalabilidad para diversos valores de soporte mínimo, número de reglas descubiertas, número de itemsets descubiertos para distintos valores de soporte, etc., habiendo extraído una serie de conclusiones con respecto a los parámetros utilizados.

- Comparación del modelo basado en los lifespans de los itemsets con diversos trabajos y enfoques de descubrimiento de reglas de asociación temporales.

1.3.2 Estructura de la Tesis

El resto de la tesis está organizado de la siguiente forma: el Capítulo 2 presenta, en su primer parte, el contexto de nuestro trabajo, es decir, el problema de las reglas de asociación en su forma clásica y, a modo de introducción, los conceptos básicos de la teoría de wavelets u onditas. La segunda parte del Capítulo 2 la dedicamos a presentar una reseña del estado del arte en lo que a descubrimientos de reglas temporales se refiere. El Capítulo 3 describe nuestro modelo temporal básico, mientras que el Capítulo 4 trata en detalle el modelo de Reglas de Asociación Temporales Generalizadas. La experimentación realizada con las dos implementaciones de nuestros algoritmos se detalla en el Capítulo 5. El Capítulo 6 compara detalladamente nuestro modelo con otros trabajos presentados en el Capítulo 2. Por último, en el Capítulo 7 exponemos las conclusiones y direcciones de futuros trabajos.

Capítulo 2

Reglas de Asociación, Onditas y Reglas de Asociación Temporales

En este capítulo presentamos el contexto de nuestro trabajo, comenzando con las reglas de asociación en su forma clásica. Luego incluimos una breve introducción a la teoría de onditas o wavelets a efectos de fundamentar parte del trabajo del Capítulo 4. Finalmente, presentamos una reseña del estado del arte referido a reglas de asociación temporales, sus aplicaciones y algoritmos.

El resto del capítulo está organizado como sigue. La Sección 2.1 está dedicada a las reglas de asociación, la formulación del problema, fundamentos teóricos y algoritmos. La Sección 2.2 presenta muy brevemente la teoría de wavelets y su aplicación a series de tiempo. El estado del arte en reglas de asociación temporales es presentado en la Sección 2.3, mientras que la Sección 2.4 concluye el capítulo.

2.1 Reglas de Asociación

La tarea de minado de asociaciones introducida en [AIS93] ha sido presentada en la Sección 1.2 del Capítulo 1. Esta tarea consiste en hallar interrelaciones entre conjuntos de ítems en bases de datos muy grandes. Agrawal define este problema de la siguiente forma. Sean $\mathbf{R}=\{A_1, \ldots, A_m\}$ un conjunto de m literales designados como *ítems*, **d** una base de datos de transacciones, donde cada transacción consiste de un conjunto de ítems $\boldsymbol{X} \subseteq \mathbf{R}$ y tiene asociado un identificador único o ***tid***. Un conjunto de ítems se denomina un itemset. Sea $\boldsymbol{X}$ un itemset de tamaño k, luego $\boldsymbol{X}$ es llamado un k-itemset. Una transacción $\boldsymbol{t} \in \mathbf{d}$ se dice que contiene un

itemset X si $X \subseteq t$. El *soporte* de un itemset X es el porcentaje de transacciones en **d** que contienen X:

$$soporte(X) = |\{t \in \mathbf{d} \mid X \subseteq t\}| \ / \ |\{t \in \mathbf{d}\}|$$

Desde un punto de vista probabilístico, se dice que el soporte de un itemset es la probabilidad de encontrar dicho itemset al extraer una transacción t, al azar, desde la base de transacciones **d**.

Un itemset es *frecuente* si su soporte es igual o superior a un soporte mínimo σ especificado por el usuario. Una regla de asociación es una implicación condicional entre itemsets, de la forma $X \Rightarrow Y$, donde $X, Y \subseteq \mathbf{R}$ y $X \cap Y = \varnothing$. El soporte de una regla tal como la anterior se obtiene como $soporte(X \cup Y)$.

La *confianza* de una regla de asociación tal como $X \Rightarrow Y$, se obtiene como

$$confianza(X \Rightarrow Y) = soporte(X \Rightarrow Y) \ / \ soporte(X)$$
$$= soporte(X \cup Y) \ / \ soporte(X)$$

De nuevo probabilísticamente, la confianza de una regla es la probabilidad condicional que una transacción contenga Y dado que contiene X.

Una regla $X \Rightarrow Y$ se considera *interesante* si $soporte(X \Rightarrow Y) \geq \sigma$ y $confianza(X \Rightarrow Y) \geq \theta$, donde θ es la confianza mínima establecida por el usuario.

Dados los umbrales de soporte y confianza σ y θ, respectivamente, fijados por el usuario, el problema del descubrimiento de reglas de asociación se puede descomponer en dos subproblemas [AIS93]:

1. Hallar todos los itemsets frecuentes en **d**, es decir, todos aquellos itemsets con soporte mayor o igual a σ. Este paso

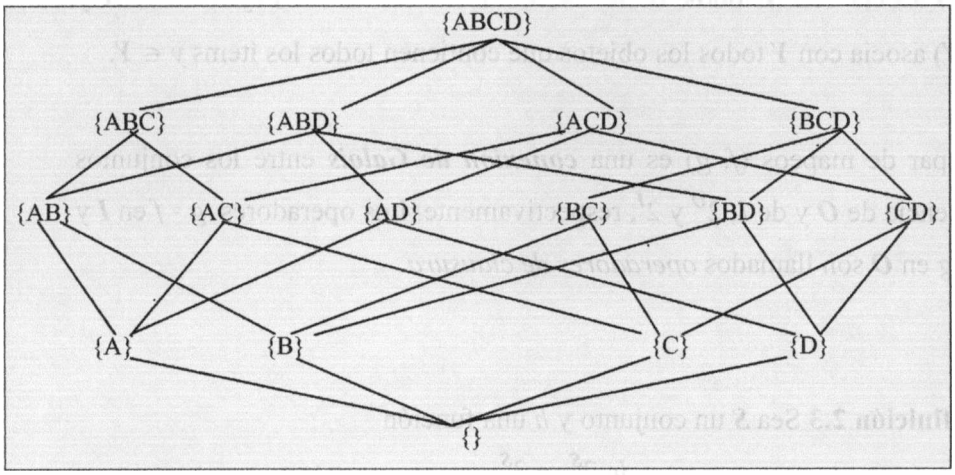
es computacionalmente intensivo, así como también en accesos a la base de datos **d**.

2. Para cada k-itemset **Z**, con k ≥ 2, generar todas las reglas de asociación $X \Rightarrow Z - X$, donde $X \subset Z$ y

$$\textit{confianza}\ (X \Rightarrow Z - X) \geq \theta.$$

El segundo paso es relativamente simple, lo que reduce el problema de descubrimiento de reglas de asociación al problema del descubrimiento de todos los itemsets frecuentes. Existen diversos algoritmos para resolver este subproblema que usan diferentes estrategias; pero todos coinciden en que usan un retículo o lattice de itemsets para descubrir los itemsets frecuentes. Es decir, todos usan las propiedades básicas que *todos los subconjuntos de un itemset frecuente son frecuentes* y que *todo superconjunto de un itemset no frecuente no es frecuente*, con el fin de podar el conjunto de itemsets, es decir, la lattice de itemsets. A modo de ejemplo, la Figura 2.1 muestra la lattice para **R** = {***A, B, C, D***}

Figura 2.1: Lattice de itemsets para **R** = {***A, B, C, D***}



2.1.1 Análisis Formal de Conceptos

El Análisis formal de Conceptos fue introducido por R. Wille en [Will82]. La tarea de mining de asociaciones está íntimamente relacionada con el trabajo de Wille. En esta subsección introduciremos los componentes básicos de esta teoría con el fin de poder utilizar dichos elementos en la formalización de principios y propiedades habitualmente presentados de manera intuitiva [Pas00].

Definición 2.1 Un ***contexto*** es una tripla $<O, I, V>$, donde O es un conjunto finito de objetos e I es un conjunto finito de ítems. $V \subseteq O \times I$ es una relación binaria entre objetos e ítems. Para $o \in O$ e $i \in I$ arbitrarios, si el objeto o está relacionado al ítem i, lo indicamos como $(o, i) \in V$.

Definición 2.2 Sea $<O, I, V>$ un contexto con $X \subseteq O$ e $Y \subseteq I$. Luego, para X e Y definimos los mapeos:

$f(X)$: $2^O \rightarrow 2^I, f(X) = \{y \in I \mid (\forall x \in X)\ (x, y) \in V\}$

$g(Y)$: $2^I \rightarrow 2^O, f(Y) = \{x \in O \mid (\forall y \in Y)\ (x, y) \in V\}$

$f(X)$ asocia con X todos los ítems comunes a todos los objetos $x \in X$ y $g(Y)$ asocia con Y todos los objetos que contienen todos los ítems $y \in Y$.

El par de mapeos (f, g) es una ***conexión de Galois*** entre los conjuntos potencia de O y de I, 2^O y 2^I, respectivamente. Los operadores $g \circ f$ en I y $f \circ g$ en O son llamados *operadores de clausura.*

Definición 2.3 Sea S un conjunto y h una función

$$h: 2^S \rightarrow 2^S$$

h es un ***operador de clausura*** sobre S si, para todo X, $Y \subseteq S$, h satisface las siguientes propiedades:

1. Extensión: $X \in h(X)$
2. Monotonía: si $X \subseteq Y$, luego $h(X) \subseteq h(Y)$
3. Idempotencia: $h(h(X)) = h(X)$

$X \subseteq S$ es ***cerrado*** si $h(X) = X$.

En nuestro caso, $h = g \circ f$ y $h' = f \circ g$ son operadores de clausura de Galois.

Dada una conexión de Galois (f, g) entre 2^O y 2^I, se cumplen las siguientes propiedades para todo Y, Y_1, $Y_2 \subseteq I$ y $X, X_1, X_2 \subseteq O$

1. $Y_1 \subseteq Y_2$ implica $g(Y_1) \supseteq g(Y_2)$ $1'$. $X_1 \subseteq X_2$ implica $f(X_1) \supseteq f(X_2)$
2. $X \subseteq g(Y)$ si y sólo si $Y \subseteq f(X)$.

Definición 2.4 Un **concepto** del contexto $<O, I, R>$ se define como un par (X, Y), donde $X \subseteq O$, $Y \subseteq I$, $f(X) = Y$ y $g(Y) = X$. Es decir, un concepto (X, Y) consiste de los conjuntos cerrados X e Y, ya que $X = g(Y) = g(f(X)) = g \circ f(X) = h(X)$ y del mismo modo $Y = h(Y)$. X se denomina la **extensión** e Y la **intensión** del concepto (X, Y).

Definición 2.5 Sea C el conjunto de itemsets cerrados derivados de un contexto, usando el operador de clausura de Galois h. El par $\mathbf{L}_C = < C, \leq >$ es una lattice completa denominada **Lattice de itemsets cerrados**. La estructura de lattice implica dos propiedades:

1. Existe un orden parcial sobre los elementos de la lattice tal que para todos los elementos $C_1, C_2 \in \boldsymbol{L}_C$, $C_1 \leq C_2$ si y sólo si $C_1 \subseteq C_2$.
2. Todos los subconjuntos de $\boldsymbol{L}_C$ tienen una máxima cota inferior, el **ínfimo**, y una mínima cota superior, el **supremo**.

2.1.2 Semántica de las Reglas de Asociación

En esta subsección redefinimos los conceptos asociados con las reglas de asociación utilizando la conexión de Galois (*f*, *g*) [Pas00, PBTL99, PBTL98].

En nuestra notación original del comienzo de esta sección, **R**, el conjunto de ítems, equivale a ***I***, el conjunto de los tid´s equivale a ***O*** y la base de datos de transacciones **d** se corresponde con el concepto de contexto. En adelante continuaremos usando la notación original con **d** como el conjunto de transacciones y, por lo tanto, reemplazará a ***O***; *f* y *g* no cambian.

Definición 2.6 El **soporte de un itemset** $\boldsymbol{Y} \subseteq \mathbf{R}$ en **d** es

$$\textbf{soporte}(\boldsymbol{Y}) = |g(\boldsymbol{Y})| \,/\, |\mathbf{d}|$$

Definición 2.7 El itemset ***Y*** se dice **frecuente** si su soporte en **d** es al menos σ. El conjunto **L** de itemsets frecuentes en **d** es

$$\mathbf{L} = \{\, \boldsymbol{Y} \subseteq \mathbf{R} \mid \text{soporte}(\mathbf{Y}) \geq \sigma \,\}$$

Definición 2.8 Sea **L** el conjunto de los itemsets frecuentes. Definimos **M**, el conjunto de los **itemsets frecuentes maximales** en **d** como

$$\mathbf{M} = \{\, \boldsymbol{Y} \in \mathbf{L} \mid \sim\exists \boldsymbol{Y}' \in \mathbf{L},\ \boldsymbol{Y} \subset \boldsymbol{Y}' \,\}$$

Un itemset frecuente es maximal si no es un subconjunto propio de ningún itemset frecuente.

A continuación demostraremos dos resultados que han sido tradicionalmente presentados como propiedades basadas en la intuición.

Propiedad 1: Todos los subconjuntos de un itemset frecuente son frecuentes.

Demostración: Sea Y, $Y' \subseteq \mathbf{R}$, $Y \in \mathbf{L}$ e $Y' \subset Y$. De acuerdo a la propiedad 1 de la conexión de Galois $Y' \subseteq Y$ implica $g(Y') \supseteq g(Y)$ implica soporte(Y') $\geq$ soporte(Y) $\geq \sigma$, de donde $Y' \in \mathbf{L}$.

Propiedad 2: Todo superconjunto de un itemset no frecuente es no frecuente.

Demostración: Sean Y, $Y' \subseteq \mathbf{R}$, $Y' \notin \mathbf{L}$ e $Y' \subset Y$. De acuerdo a la propiedad 1 de la conexión de Galois, $Y \supseteq Y'$ implica $g(Y) \subseteq g(Y')$ implica soporte(Y) $\leq$ soporte(Y') $\leq \sigma$, luego $Y \notin \mathbf{L}$.

En lo que sigue, usamos la conexión de Galois para redefinir *Regla de Asociación* y la noción de *regla válida.*

Definición 2.9 Una regla de asociación r es una implicación de la forma $r: Y \Rightarrow Z$ [**soporte**, **confianza**], donde $Y, Z \subseteq \mathbf{R}$, $Y \cap Z = \varnothing$ y

$$\textbf{soporte}\ (r) = |g(Y \cup Z)| \ / \ |\mathbf{d}|,$$

$$\textbf{confianza}\ (r) = \textbf{soporte}\ (Y \cup Z) \ / \ \textbf{soporte}\ (Y) = |g(Y \cup Z)| \ / \ |g(Y)|.$$

Definición 2.10 Una regla de asociación es una **regla válida** si su soporte y confianza son mayores o iguales a los umbrales σ y θ establecidos, respectivamente. El conjunto AR de reglas de asociación válidas resulta

$AR(\mathbf{d}, \sigma, \theta) = \{ r : Y \Rightarrow V - Y$ [**soporte, confianza**], $Y \subset V \mid V \subseteq \mathbf{L}$ y **confianza** $(r) \geq \theta \}$

En la siguiente subsección analizaremos los aspectos algorítmicos relacionados con el descubrimiento de las reglas de asociación.

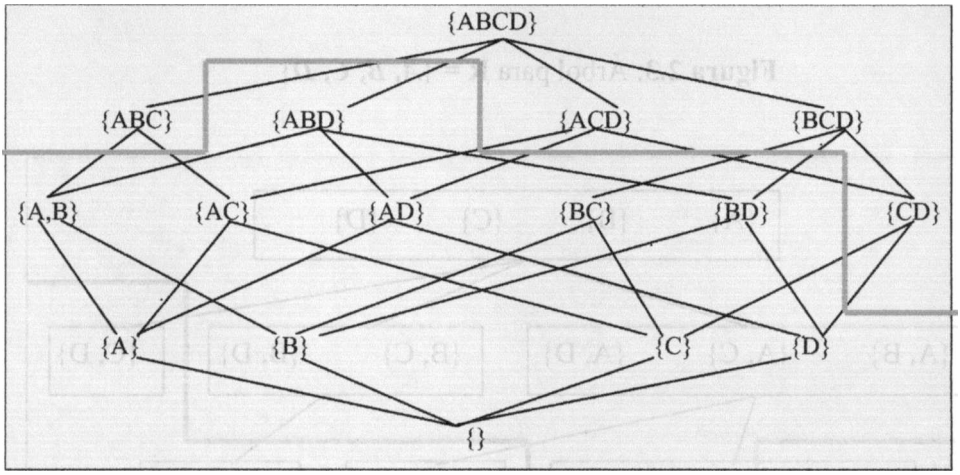
2.1.3 Algoritmos para el descubrimiento de Reglas de Asociación

Existe una gran variedad de algoritmos eficientes para el descubrimiento de reglas de asociación. Para su presentación ordenada utilizaremos el marco comparativo desarrollado por [HGN00]. Como se mencionó más arriba, la tarea del descubrimiento de reglas de asociación puede reducirse al problema de hallar todos los itemsets frecuentes con respecto a un umbral dado σ.

Sabemos que el espacio de búsqueda, aún con un número reducido de ítems en términos de aplicaciones reales, resulta inmenso. Como ejemplo, consideremos el caso **R** = { A, B, C, D}. El espacio de búsqueda forma una lattice, tal como la de la Figura 2.1. Los itemsets frecuentes están separados de los itemsets no frecuentes por una línea de trazo grueso, en color rojo, siendo los primeros los ubicados en la parte inferior y los segundos, en la parte superior. La existencia del *borde* que separa ambos grupos de itemsets es independiente de la base de datos **d**, de σ y está garantizado por la propiedad de clausura descendente del soporte de los itemsets, que designamos como Propiedad 1 en la subsección anterior. El conjunto de itemsets frecuentes maximales se denomina, en particular, *borde positivo*. El borde marcado separa las sublattices inducidas por los

itemsets frecuentes maximales. Por otra parte, denominamos *borde negativo* al conjunto de los itemsets V no frecuentes tal que algún $W \subset V$, con $|W| = |V| - 1$, sea frecuente.

Figura 2.2: Lattice para $\mathbf{R} = \{A, B, C, D\}$ con Borde



La mayoría de los algoritmos tiende a podar el espacio de búsqueda en base al borde: cuando se llega al borde, se omite analizar los itemsets que se encuentran por encima de él.

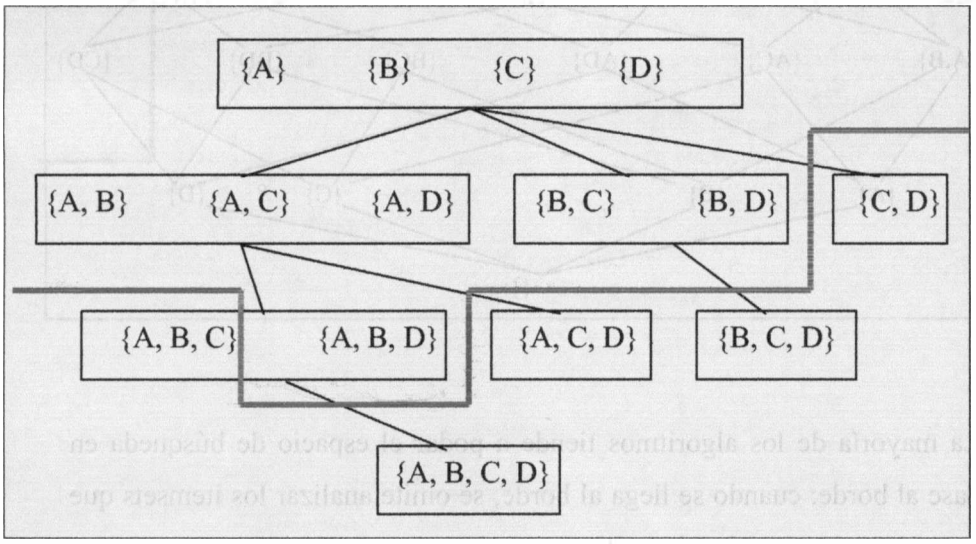
Para analizar como operan los algoritmos consideremos un itemset $X \subseteq \mathbf{R}$, donde los ítems componentes de X, $X.\text{ítem}_i$, se encuentran ordenados lexicográficamente. El n-prefijo de X, con $|X| \geq n$ se define por $P = \{X.\text{ítem}_i \mid 1 \leq i \leq n\}$, mientras que la clase de las extensiones de P resulta $E(P) = \{X \subseteq \mathbf{R} \mid |X| = |P| + 1 \text{ y } P \text{ es un prefijo de } X\}$.

Representamos las clases de las extensiones de P por medio de los nodos de un árbol. Dos nodos se conectan por un arco si todos los itemsets de una clase E pueden ser generados juntando dos itemsets de la clase predecesora E′. Continuando con nuestro ejemplo, la Figura 2.2 presenta el árbol para $\mathbf{R} = \{A, B, C, D\}$. Hemos invertido la lattice de la Figura 2.1

para conservar la representación estándar del árbol. La línea gruesa marca el borde que separa los itemsets frecuentes de los no frecuentes.

Podemos observar en el árbol que las clases E son clases de equivalencia, con respecto al prefijo.

Figura 2.3: Árbol para **R** = {***A, B, C, D***}



Una consecuencia de lo anterior es que si la clase predecesora E′ no contiene al menos dos itemsets frecuentes, luego E no contendrá **ningún** itemset frecuente. Por lo que si en nuestro recorrido del árbol hacia abajo encontramos una tal clase E′ entonces habremos encontrado el borde que separa los itemsets frecuentes de los no frecuentes. Como no es necesario ir más allá del borde, podamos la clase E y cualquier otra descendiente de E en el espacio de búsqueda.

Una distinción básica de los diversos algoritmos es la estrategia de búsqueda del borde. Entre esas estrategias se distinguen: *búsqueda*

primero en amplitud (BSF) y la *búsqueda primero en profundidad* (DSF).En el primer caso, BSF, se procede por niveles computando el soporte de todos los (k-1)-itemsets previo a tratar los k-itemsets. En el segundo, DSF, se desciende recursivamente aprovechando la estructura del árbol.

Cada itemset hallado en el recorrido de la lattice se considera potencialmente frecuente y se denomina itemset *candidato*. Para determinar el soporte de los itemsets candidatos encontramos dos alternativas: i) recorrer la base de datos **d**, en la que se cuentan las ocurrencias de cada uno de los candidatos y ii) para hallar el soporte de un itemset $\boldsymbol{Z} = \boldsymbol{X} \cup \boldsymbol{Y}$ se obtiene la lista de transacciones en las que se encuentran $\boldsymbol{X}$ e $\boldsymbol{Y}$ simultáneamente, es decir, la intersección de la lista de $\boldsymbol{X}$ con la lista de $\boldsymbol{Y}$; luego se considera la cantidad de elementos de la lista de transacciones que contienen $\boldsymbol{Z}$.

Caracterización de los Algoritmos

Los algoritmos presentados en esta sección son caracterizados de acuerdo con:

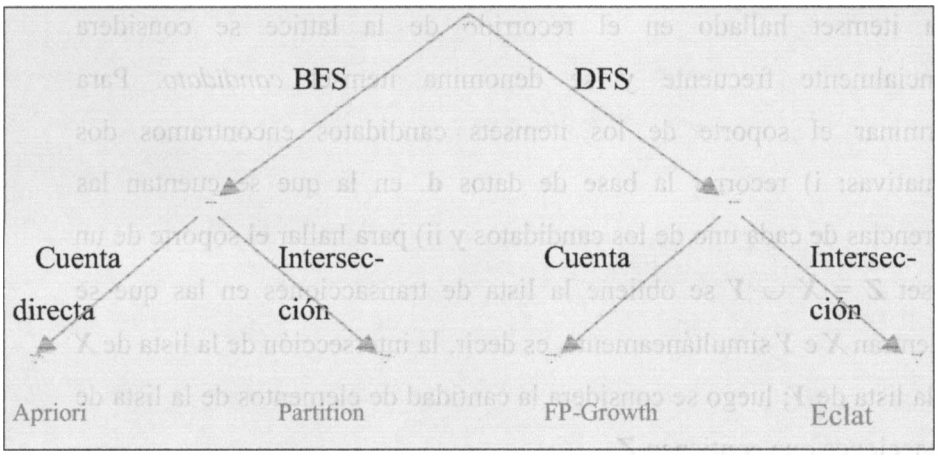
- La estrategia para recorrer el espacio de búsqueda y
- La estrategia para determinar los soportes de los itemsets.

La combinación de las diversas estrategias genera cuatro tipos de algoritmos, lo que se muestra en la Figura 2.4.

Como ejemplo de la primera clase, BFS y cuenta directa, detallaremos el algoritmo Apriori [AS94], mientras que para los otros casos describiremos brevemente un ejemplo de cada uno. Así, para búsqueda BFS e intersección de listas de identificadores de transacciones o tid's, nos

referiremos al algoritmo *Partition*, para búsqueda DFS y cuenta directa, el ejemplo es *FP-growth* y, finalmente, para búsqueda DFS e intersección de listas de tid's, describiremos el algoritmo *Eclat*.

Figura 2.4: Combinación de Estrategias y Algoritmos ejemplo



Algoritmo A priori

Se asume que los ítems en cada transacción se conservan clasificados en orden lexicográfico. La cantidad de ítems en un itemset se denomina *tamaño del itemset.* Los ítems dentro de un itemset están en orden lexicográfico. Se usa la notación c[1] c[2] ... C[k] para representar un k-itemset c que consiste de los ítems c[1], ..., c[k], donde c[1] < c[2] < ...< c[k]. Si c = X.Y e Y es un itemset, Y se denomina una m-extensión de X.

Notación:

k-itemset	un itemset con k ítems.
L_k	conjunto de k-itemsets frecuentes (con soporte mínimo).
C_k	conjunto de k-itemsets candidatos (potencialmente frecuentes). Cada miembro de C_k tiene 2 campos: i) itemset

ii) cuenta de soporte.

Figura 2.5: Algoritmo Apriori

```
1. L1 = { 1-itemsets frecuentes};
2. for (k = 2; Lk-1 ≠ ∅ ; k ++) do begin
3.     Ck = apriori-gen( Lk-1 ); //nuevos candidatos
4.     foreach transacción t ∈ d do begin
5.         Ct = subconjunto ( Ck , t ); //candidatos en t
6.         foreach candidato c ∈ Ct do
                                        //incrementar
7.                     c.count ++;  // su count
8.     end
9.     Lk = { c ∈ Ck / c.count ≥ minsup }
10. end
11. Respuesta = ∪k Lk ;
```

L_1 se obtiene en una primer pasada en la que se cuentan las ocurrencias de los ítems a efectos de determinar los 1-itemsets frecuentes. Cualquier pasada siguiente k consiste de dos fases: en la primera se obtienen los itemsets candidatos C_k de tamaño k, en base a los itemsets frecuentes L_{k-1} de tamaño k-1 obtenidos en la pasada k-1, por medio de la función *apriori-gen;* en la segunda fase se lee la base de datos de transacciones para calcular el soporte de los itemsets candidatos de C_k, para lo que se utiliza la función *subconjunto,* la que determina si cada miembro de C_k está contenido en la transacción t. El algoritmo hace tantas pasadas sobre la base de datos como la cardinalidad máxima de un itemset candidato.

Generación de candidatos *a priori*

La función *apriori-gen* toma como argumento L_{k-1}, el conjunto de todos los (k-1)-itemsets frecuentes, y devuelve un superconjunto de los k-itemsets frecuentes. Esta función ha sido organizada en un *Paso de Junta* y un *Paso de Poda* o prune.

1. Paso de Junta:

En SQL

Figura 2.6: paso de junta en Apriori-gen

```
insert into Ck
select p.item1, p.item2, ..., p.itemk-1, q.itemk-1
  from Lk-1 p, Lk-1 q
    where p.item1 = q.item1 and ... and
          p.itemk-2 = q.itemk-2 and p.itemk-1 <
  q.itemk-1;
```

En el siguiente paso(*pruning*) se eliminan todos los itemsets candidatos $c \in C_k$ tal que algún subconjunto de c con k-1 ítems no esté en L_{k-1}.

2. Paso de Poda:

Figura 2.7: Paso de poda en Apriori-gen

```
foreach itemset c ∈ Ck do
      foreach (k-1) subconjuntos s de c do
            if (s ∉ Lk-1) then
                  delete c from Ck;
```

Correctitud de *apriori.* Como parte de la prueba de correctitud se requiere demostrar que $L_k \subseteq C_k$. Ya que cualquier subconjunto de un itemset frecuente es también frecuente, si se extiende cada itemset en L_{k-1} con todos los posibles ítems y luego se eliminan todo los subconjuntos de k-1 ítems que no están en L_{k-1}, entonces nos queda un superconjunto de los itemsets en L_k.

La expresión SQL de *apriori-gen* es equivalente a extender L_{k-1} con cada ítem de R y luego eliminar los k-itemsets en los que el (k-1)-itemset obtenido al eliminar el k-ésimo ítem no está en L_{k-1}. La última condición del *where*, $p.item_{k-1} < q.item_{k-1}$, evita la generación de duplicados. Luego el C_k generado contiene a L_k. El paso siguiente(*pruning*) tampoco elimina ningún itemset que podría estar en L_k, ya que se eliminan de C_k los itemsets c tales que algún (k-1)-subconjunto no está en L_{k-1}.

DIC (Dynamic Itemset Counting) de [BMU97] es una variante de A priori que reduce notablemente la cantidad de pasadas a la base de datos de transacciones. Esto se logra particionando la base de datos **d** en M porciones y computando los candidatos de tamaño k ni bien se encuentra un par de frecuentes de tamaño k – 1, al terminar de analizar una porción, aunque no se hayan contado todas sus ocurrencias en **d**.

BSF e Intersección

El algoritmo *Partition* de [SON97] usa la propiedad de a priori y la intersección de conjuntos de tid's de **d.** *Partition* divide la base de datos **d** en particiones que son tratadas en forma independiente. Una pasada final sobre **d** completa, permite consolidar el soporte de todos los itemsets que eran frecuentes en las particiones.

DFS y cuenta directa

FP-growth fue introducido en [HPJ00]. Este algoritmo utiliza una novedosa estructura llamada FP-tree, o árbol de patrones frecuentes. FP-tree es una estructura de árbol de prefijos que permite almacenar información sobre los itemsets frecuentes, en forma comprimida. La generación del árbol se realiza contando ocurrencias en una búsqueda

DFS. Luego el problema de mining de la base de datos se transforma en el de mining el FP-tree.

El algoritmo FP-growth utiliza una estrategia "divide and conquer" para descomponer la tarea de mining en un conjunto de tareas más pequeñas. Cada una de esas tareas realizan el mining de determinados patrones, incluyendo en la última etapa la determinación del soporte de todos los itemsets frecuentes.

DSF e Intersección

El algoritmo Eclat propuesto en [ZPOL97] combina la búsqueda DFS con el enfoque de intersección de listas de tid's. Dada la intersección de dos listas de tid's, sólo interesa la lista resultante si su cardinal igual o supera el valor de soporte mínimo fijado.

Generación de reglas

Para generar las reglas, es necesario hallar todos los subconjuntos para todo itemset frecuente [AS94]. Luego, dado un itemset frecuente l debemos hallar, para cada subconjunto propio a de l, las reglas $a \Rightarrow (l - a)$ tal que $s(l) / s(a) \geq \theta$. La frecuencia de cualquier subconjunto a' de a es mayor que la frecuencia de a. En consecuencia, la confianza de $a' \Rightarrow (l - a')$ no puede superar la confianza de $a \Rightarrow (l - a)$. Es decir, si no es posible generar una regla utilizando todos los ítems de p con a como el antecedente, tampoco será posible si el antecedente es a'. Luego, para que se satisfaga una regla $a \Rightarrow (l - a)$, también se deben satisfacer las reglas de la forma $a' \Rightarrow (l - a')$, con a' cualquier subconjunto no vacío de a.
Esta propiedad es análoga a la del soporte de un itemset: si un itemset es frecuente también lo son todos sus subconjuntos. Aplicando estas ideas tenemos el siguiente algoritmo, el que parte de los itemsets frecuentes y

las correspondientes reglas con sólo un ítem en el consecuente. Se usan los consecuentes de esas reglas y la función *aprior-gen* para generar todos los consecuentes posibles con dos ítems en las reglas, luego tres ítems, etc..

```
1.   forall k-itemsets frecuentes l_k y k ≥ 2 do begin
2.      H_1 = { consecuentes de reglas desde l_k con sólo un ítem
                en el consecuente};
3.      call ap-genreglas (l_k, H_1);
4.   end

5.   procedure ap-genreglas(l_k: k-itemset frecuente, H_m:
     conjunto de consecuentes de m ítems)
6.   if (k > m + 1) then begin
7.      H_{m+1} = apriori-gen(H_m);
8.      forall h_{m+1} ∈ H_{m+1} do begin
9.         conf = s(l_k) / s(l_k − h_{m+1});
10.        if (conf > θ) then
11.           output la regla (l_k − h_{m+1}) ⇒ h_{m+1} con confianza
                 conf y soporte s(l_k);
12.        else
13.           delete h_{m+1} from H_{m+1};
14.     end
15.     call ap-genreglas(l_k, H_{m+1})
16.  end
```

Para hallar H_1 podemos utilizar el siguiente algoritmo:

```
Forall itemset frecuente l_k, k ≥ 2 do
      Call genreglas(l_k, l_k);

// genreglas genera todas las reglas válidas a ⇒ (l_k − a) para todo a
                                                          ⊂ l_k
procedure genreglas(l_k: k-itemset frecuente)
1.   A = {(k-1)-itemsets a | a ⊂ l_k }
2.   forall a ∈ A do begin
3.      if (s(l_k) / s(a)) ≥ θ
4.         output la regla a ⇒ (l_k − a) con confianza (s(l_k)/s(a)) y
     soporte s(l_k);
5.      end
6.   end
```

2.2 Introducción a Wavelets

Esta sección tiene por finalidad brindar soporte formal a la técnica usada para la representación de la historia de los itemsets, desarrollada en el Capítulo 4.

La principal motivación que alentó el desarrollo de wavelets e ideas relacionadas, fue la búsqueda de algoritmos rápidos para computar representaciones compactas de funciones y conjuntos de datos.

Wavelets, o pequeñas ondas, son funciones que se usan para representar datos u otras funciones [Gra95, LLZO02, MVW98, Dau92]. Esta idea tiene su antecedente en el análisis de Fourier, en el que se pueden superponer senos y cosenos para la representación de otras funciones.

Las wavelets son una familia de funciones de base ortogonal que exhiben una serie de ventajas para nuestra aplicación. Entre éstas se cuentan la posibilidad de eliminar ruido de muchos tipos de funciones como saltos, espigas, etc.; el tratamiento que hacen de funciones que están netamente contenidas en dominios finitos, tales como los lifespans de los itemsets; la posibilidad única de analizar de acuerdo a escala: los algoritmos de wavelets procesan los datos en diferentes escalas o resoluciones, de modo que si observamos una función con una ventana grande, notaremos las características gruesas, mientras que si lo hacemos con una ventana pequeña, observaremos las características de detalle [PW00].

Las funciones wavelets están localizadas en el espacio. Esta característica, junto con la de localización de frecuencia de las wavelets, hace que muchas funciones resulten "dispersas" cuando se transforman al dominio wavelet. Dicha dispersión atiende otro requerimiento de nuestra aplicación que es la posibilidad de compresión de datos.

El procedimiento de análisis wavelet consiste en adoptar una función prototipo llamada *wavelet madre*. El análisis de las dos dimensiones consideradas, tiempo y frecuencia, se realiza de la siguiente forma. Para el tiempo se utiliza una versión contraída, de alta frecuencia de la wavelet madre, mientras que la frecuencia se analiza con una visión dilatada, de baja frecuencia de la misma función.

La función original, por ejemplo una serie temporal, normalmente llamada señal, se puede representar en términos de una expansión wavelet, or medio de coeficientes de una combinación lineal de las funciones wavelet. Truncando los coeficientes que está por debajo de un umbral, se consigue representar los datos en forma rala o dispersa.

2.2.1 Qué es una wavelet

Una wavelet es una pequeña onda que crece y decrece en un período de tiempo limitado. Para formalizar el concepto, consideremos una función ψ sobre los reales $\mathcal{R}$ que satisface las dos propiedades siguientes:

1. La integral de ψ es cero

$$\int \psi(u)\, du = 0$$

2. La integral de ψ^2 es 1

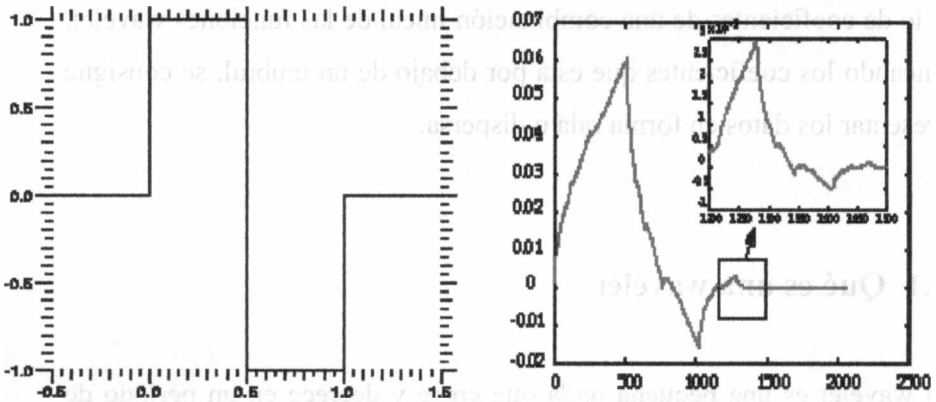
$$\int \psi^2(u)\, du = 1$$

En particular, para cualquier ε, con $0 < \varepsilon < 1$, debe existir un intervalo [-T, T] de longitud finita tal que la integral entre –T y T de ψ^2 sea $< 1 - \varepsilon$. Si ε es muy próximo a cero, luego ψ tiene un desvío insignificante de cero fuera de [-T, T].

Se considera que el intervalo [-T, T] es muy pequeño, comparado con (-∞,∞) y de (1) y (2) más arriba, surge que ψ tiene la forma de una pequeña onda o wavelet.

Ejemplos de wavelets son la llamada wavelet de Haar y la wavelet de Daubechies, cuyas gráficas podemos observar en la Figura 2.8 siguiente.

Figura 2.8: Ejemplos de wavelets



2.2.2 Transformada Wavelet Discreta

La dilatación y traslación de la función madre ψ (x) definen una base ortogonal o wavelet base

$$\psi_{s,h}(x) = 2^{s/2}\psi(2^s x - h) \qquad (1)$$

Las variables s y h son enteros que escalan y dilatan la función madre ψ para generar wavelets, tal como las familias de Haar y Daubechies. Los índices s y h indican la amplitud y la posición de la wavelet, respectivamente. Las funciones madre son reescalables por potencias de 2 y trasladadas por enteros.

ψ ($2^s x - k$) se obtiene desde una función ψ (x) por una dilatación binaria y una traslación diádica.

Una función $\psi \in L^2(\mathcal{R})$, el conjunto de las funciones de valores reales cuadrado integrables, se denomina una wavelet ortogonal si la familia $\{\psi_{s,k}\}$ de (1) es una base ortogonal de $L^2(\mathcal{R})$, es decir;

$$\langle \psi_{s,k}, \psi_{h,m} \rangle = \delta_{sk} \cdot \delta_{hm} \quad \text{con } s, k, h, m \in Z$$

y toda $f \in L^2(\mathcal{R})$ puede escribirse como

$$f(x) = \Sigma^{\infty}_{s,k=-\infty} c_{s,k} \cdot \psi_{s,h}(x)$$

La representación de f anterior se denomina una serie wavelet. Los coeficientes wavelets $c_{s,k}$ están dados por

$$C_{s,k} = \langle f, \psi_{s,k} \rangle$$

2.2.3 Análisis Multi-Resolución

Dada una wavelet madre $\psi(x)$, asumimos que la familia de wavelets $\psi_{s,h}(x)$ por ella generada, constituye una base ortonormal de la clase de señales de energía finita [LC00].

Para cada valor entero de s, las wavelets $\psi_{s,h}(x)$ generan un subespacio de señales que comparten la misma localización y que indicaremos por W_j.

Una función de escalamiento ϕ es esencialmente una función que puede ser escrita como una combinación de $\phi(2x-k)$, una versión de ϕ trasladada y escalada, que se expresa de la siguiente forma

$$\phi(x) = \sum_{k=-\infty}^{\infty} p_k \phi(2x - k). \qquad (2)$$

Esta expresión se conoce como la *ecuación de escalamiento*. La secuencia $\{p_k\}$ es llamada la secuencia de escalamiento de ϕ.

Considérese

$$\phi_{j,k}(x) := \phi(2^j x - k), \quad j, k \in Z. \qquad (3)$$

una versión de ϕ trasladada en k y escalada en $1/2^j$. Para una j fija se hace referencia al nivel j.

Sea Vj el espacio vectorial generado mediante traslaciones de ϕ en el nivel j.

Obsérvese que según (2) $V_0 \subset V_1$, y más aún se genera una secuencia de subespacios anidados V_j tales que

$$\cdots V_{-1} \subset V_0 \subset V_1 \subset V_2 \subset \cdots \qquad (4)$$

Proposición 2.2.1: Toda función suave en R se puede representar en términos de $\phi_{j,k}$ para algún j suficientemente grande, en otras palabras

$$\operatorname{clos}_{L^2}\left(\bigcup_{j \in Z} V_j\right) = L^2(R) \qquad (5)$$

Las propiedades (2), (3), (4) y (5) dan lugar a lo que se conoce como *Análisis Multi-Resolución.*

Dada una secuencia anidada de subespacios V_j, existen subespacios W_j que son el complemento ortogonal de V_j en V_{j+1} esto es

$$V_{j+1} = V_j \oplus W_j, \quad j \in Z$$

donde

$$W_j \perp W_{j'} \text{ si } j \neq j'$$

Puesto que los espacios están anidados según (4)

$$V_J = V_j \oplus \bigoplus_{k=0}^{J-j-1} W_{j+k} \text{ for } j < J$$

Proposición 2.2.2: Dada una función de escalamiento ϕ en V_j, existe otra función ψ en W_0 llamada *wavelet* tal que

$$\{\psi_{j,k} : k \in Z\} \quad \text{genera } W_j,$$

donde :

$$\psi_{j,k}(x) := \psi(2^j x - k), \quad j, k \in Z.$$

puede ser escrito en términos de $\phi(2^1x\text{-}k)$ que forma una base de V_1, entonces

$$\psi(x) = \sum_{k=-\infty}^{\infty} q_k \phi(2x - k)$$

conocida como *ecuación de escalamiento para la wavelet.* $\{q_k\}$ es llamada la secuencia de escalamiento para la wavelet.

2.2.4 Transformada Wavelet Rápida

Debido a que tanto V_0 como W_0 son subespacios de V_1, es decir

$$V_0 \subset V_1 \text{ y } W_0 \subset V_1$$

Es posible expresar ϕ y ψ en términos de las funciones base de V_1

$$\phi(x) = 2\Sigma\, h_k\phi(2x - k)$$

$$\psi(x) = 2\,\Sigma\, g_k\phi(2x - k)$$

Debido al análisis Multi-Resolución esas relaciones son también válidas entre V_{j+1}, V_j y W_j, para cualquier j.

Los coeficientes h_k y g_k definen unívocamente la función $\phi(x)$ y la wavelet $\psi(x)$.

Puesto que $V_{j+1} = V_j \oplus W_j, \quad j \in Z$, se puede expresar la función f(x) que se escribe en términos de las funciones base de V_{j+1}, en términos de las funciones base de V_j y W_j, es decir

$$f(x) = \Sigma\, a_{j+1,\,k}\, \phi_{j+1,\,k}(x) = \Sigma a_{j,l}\, \phi_{j,\,l}(x) + \Sigma d_{j,\,l}\, \psi_{j,\,l}(x)$$

con los coeficientes de transformación $a_{j,l}$ y $d_{j,\,l}$ definidos de la siguiente forma

$$a_{j,l} = \sqrt{2}\,\Sigma\, h_{k-2l}\, a_{j+1,\,k}$$

$$d_{j,\,l} = \sqrt{2}\,\Sigma\, g_{k-2l}\, a_{j+1,\,k}$$

esta operación se conoce como la Transformada Wavelet Rápida, llamada así porque su complejidad es proporcional a la longitud de la señal, es decir, $O(n)$.

La transición $a_j \rightarrow a_{j+1}, d_{j+1}$ corresponde a un cambio de base en V_j . La descomposición de una señal a_j en el nivel j en una señal de menor resolución a_{j+1} y una señal diferencia o detalle d_{j+1}, forma la base del algoritmo Piramidal

La transformada wavelet inversa puede lograrse por un método similar. Las figuras 2.9 y 2.10 ilustran el proceso directo de descomposición y el proceso inverso de reconstrucción.

Figura 2.9: el proceso de descomposición

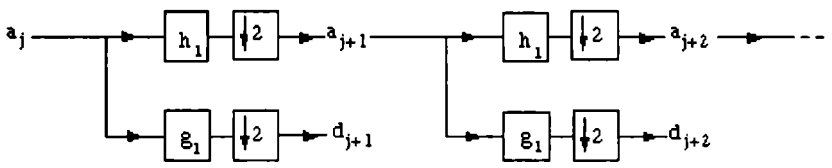
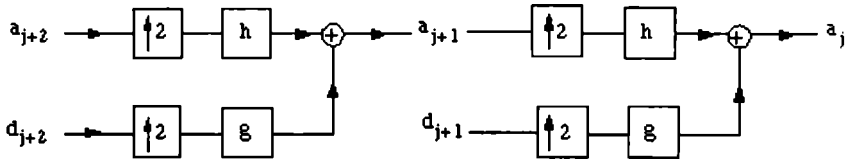


Figura 2.10: el proceso de reconstrucción



La implementación de las transformadas wavelet directa e inversa, descomposición y reconstrucción, respectivamente, se puede realizar por medio de los siguientes pasos:

1. *Seleccionar la wavelet base*: la transformada wavelet tiene un conjunto infinito de posibles funciones base. Las diversas familias se diferencian en el grado de compacidad con que las funciones base se localizan en el tiempo y en cuán suaves son. Las familias que se han utilizado en la representación de series temporales como las de los itemsets son Haar y Daubechies. En [PM02] han

experimentado con ambas, en particular Daubechies-12, mostrando que, en el caso de Haar, el error relativo en la reconstrucción fue de 1.657, mientras que con Daubechies el error se redujo a 0.477. Para ambas el ratio de compresión fue 0.078.

2. *Determinar el ancho de banda de la señal*: el ancho de banda es el número de espacios a transformar, o sea, desde el espacio más fino al más grueso.

3. *Computar los coeficientes del subespacio.*

Los coeficientes de detalle son usualmente normalizados, por lo que en muchas casos dichos coeficientes resultan ser muy pequeños en magnitud y son reemplazados por cero. Esto tiende a una fuerte compresión en la densidad de datos.

En resumen, el procesamiento de una señal s(t), de energía finita, en el contexto de un análisis multirresolución, puede definirse en los siguientes pasos:

1) Representación de la señal en el espacio de escala inicial V_0.
2) Cómputo de los coeficientes wavelets $c_{j,k}$ para los niveles j = -1, . . ., J y los coeficientes de escala $s_{J,k}$, para un nivel mínimo J < 0.
3) Clasificación y procesamiento de la información dada por los coeficientes.
4) Análisis del residuo o tendencia general.
5) Reconstrucción total o parcial de la señal a partir de los coeficientes wavelets.

2.3 Reglas de Asociación Temporales

Data mining temporal es considerado una importante extensión al campo de Data mining tradicional orientado al descubrimiento de patrones sobre estados de una base de datos[RS02]. Su importancia radica en que permite descubrir patrones que varían en el tiempo, es decir, habilita el mining del comportamiento de conjuntos de objetos como opuesto a descubrir reglas estáticas, válidas en un determinado estado.

Existe un gran número de paradigmas para la modelización y descubrimiento del conocimiento en data mining temporal. Entre ellos se cuentan los siguientes: descubrimiento de tendencias en series temporales, descubrimiento de similitudes entre series temporales, descubrimiento de patrones en secuencias, clasificación de datos temporales, clasificación de secuencias de eventos, clustering temporal y descubrimiento de reglas de asociación temporales.

Roddick y Spilipoulou [RS02, RHS01], con la finalidad de definir un marco que permita categorizar los numerosos trabajos dentro de esta subárea de data mining, establecen tres aspectos a considerar en cada caso. El primero de ellos se basa en los tipos de datos temporales que se analizan; el segundo se refiere al paradigma aplicado, mientras que el tercero se centra en la meta del proceso de descubrimiento de conocimiento.

Entre los tipos de datos temporales se consideran:

- Datos *estáticos*: no existe un contexto temporal. Eventualmente contienen referencias al tiempo de transacción.
- *Secuencias*: son listas ordenadas de eventos que, sin embargo, no necesariamente tienen asociado un timestamp.

- Datos *Timestamped*: se refiere a secuencias de datos, donde cada objeto tiene asociado un timestamp. Tal es el caso de las transacciones de canastas de mercado.
- *Temporal total*: en este caso, se trata de datos en una relación temporal, donde cada tupla puede tener una o más dimensiones temporales asociadas, como tiempo válido y/o tiempo de transacción.

En los últimos años se han desarrollado diferentes marcos conceptuales que encausaron diversas líneas de investigación en data mining temporal. Entre éstos se cuentan la lógica y el razonamiento temporal que orientaron el denominado pattern matching temporal y el mining de secuencias; el descubrimiento de conocimiento en bases de datos temporales, particularmente en bases de datos relacionales temporales, inicialmente orientado al análisis de series temporales; también, basado en el concepto de *monitoreo de la actividad*, se han desarrollado técnicas para el análisis de secuencias de eventos, orientadas a detectar la ocurrencia de patrones de comportamiento interesante.

Los tipos de conocimiento temporal se han dividido en dos amplias categorías: la extensión temporal de reglas, u otro tipo de representación, estáticas y la particular a los datos temporales. En la primer categoría encontramos las *reglas de asociación temporales*, la *clasificación* y *caracterización temporal*, entre otras. En la segunda categoría se distinguen el *análisis de tendencias*, el *análisis de secuencias* y el denominado *mining de segundo orden* sobre reglas temporales descubiertas.

El resto de esta sección está dedicada específicamente al descubrimiento de reglas de asociación temporales. Básicamente, constará de una revisión de los distintos trabajos publicados en el área, en orden aproximadamente

cronológico. Su finalidad es presentar el estado del arte en reglas de asociación temporal y servir de marco de referencia y comparación para esta tesis.

2.3.1 Reglas Temporales

Los primeros trabajos sobre data mining incluyendo aspectos temporales, están usualmente relacionados al análisis de secuencias de eventos [AS95, BWJL98, MTV95, SA96]. En estos trabajos, el objetivo es descubrir regularidades en la ocurrencia de ciertos eventos e interrelaciones temporales entre diferentes eventos. En particular, Mannila et al. en [MTV95] discuten el problema de reconocer episodios frecuentes en una secuencia de eventos, donde un episodio se define como una colección de eventos que ocurren durante intervalos de tiempo de un tamaño específico. Por otra parte, Agrawal y Srikant en [AS95] analizan el problema del descubrimiento de patrones secuenciales en una base de datos de transacciones, donde la solución consiste en crear una secuencia para todo cliente y buscar patrones frecuentes en cada secuencia, en base a una ventana de tiempo.

En Bettini et al. [BWJL98], los autores consideran patrones más complejos, donde se tratan distancias temporales con múltiples granularidades.

Chakrabarti et al. en [CSD98], en un enfoque totalmente diferente, usan el Principio de Descripción de Longitud Mínima junto con un esquema de codificación, para analizar la variación de la correlación entre ítems en el tiempo. La meta de este análisis es extraer patrones temporalmente sorprendentes. Utiliza para esto una medida de sorpresa que se modeliza

como la diferencia entre la probabilidad esperada y la observada, de la aparición concurrente de un par de ítems o grupo de ítems.

En lo que sigue, nos referiremos específicamente al problema del mining de reglas de asociación temporales.

2.3.2 Reglas de Asociación Temporales: El Enfoque de Calendarios

Si bien el trabajo sobre reglas de asociación cíclicas de Özden et al.[ÖRS98], no está basado en el concepto de calendario para el descubrimiento de patrones temporales, es incluido en esta sección por ser el antecedente inmediato de la siguiente publicación que es, a nuestro entender, el primer trabajo que vincula dicho concepto con el mining de reglas de asociación.

Reglas de Asociación Cíclicas (Özden, Ramaswamy y Silberschatz)

Uno de los primeros trabajos sobre reglas de asociación temporales es el de Özden et al. [ÖRS98], donde los autores estudian el problema de reglas de asociación que existen en ciertos intervalos de tiempo y exhiben variaciones cíclicas regulares en el tiempo.

Los autores observan que, aunque una regla de asociación puede cumplir con los mínimos establecidos de soporte y confianza dentro del espectro de tiempo total, el análisis de los datos en granularidades más finas puede revelar que la regla sólo se cumple en determinados intervalos de tiempo y no en otros. Así, las reglas pueden exhibir variaciones horarias, diarias,

etc., que tienen forma de ciclos. También observan que pueden existir reglas que no cumplan con los umbrales fijados y que sólo lo hagan en determinados intervalos. Si esos intervalos son periódicos, el descubrimiento de esas reglas y sus ciclos puede revelar información interesante.

Una regla de asociación se denomina *cíclica* si cumple con los parámetros de soporte y confianza a intervalos regulares. Se define el problema del *Descubrimiento de Reglas de Asociación Cíclicas* como el de generación de todos los ciclos de las reglas de asociación. Un ejemplo de este tipo de regla es el siguiente. Si *Café* $\Rightarrow$ *Mediaslunas* se cumple durante los intervalos 8 a 9 de la mañana y 5 a 6 de la tarde, todos los días, luego la regla *Café* $\Rightarrow$ *Mediaslunas* tiene los ciclos $c_1 = (24,8)$ y $c_2 = (24,17)$, donde la unidad de tiempo es una hora.

Los mismos autores, en el trabajo de Ramaswamy, Majan y Silberschatz [RMS98], reconocen que la periodicidad tiene un poder limitado para describir situaciones de la vida real ya que "en una base de día-a-día, los seres humanos tratan con patrones complicados que no pueden ser descriptos por simples periodicidades". Además, el modelo de reglas cíclicas es demasiado rígido, ya que no puede capturar una regla que exhibe un determinado patrón la mayoría de las veces, sino que debe hacerlo todas las veces. La generalización del trabajo sobre reglas cíclicas dio origen a las llamadas *reglas de asociación de calendarios*.

Reglas de Asociación de Calendarios (Ramaswamy, Majan y Silberschatz)

En este trabajo, Ramaswamy et al. [RMS98] introducen el problema del mining de patrones temporales definidos por el usuario en reglas de

asociación. La idea central es usar un *álgebra de calendarios* que permite definir y manipular grupos de intervalos de tiempo, para describir fenómenos de interés en las reglas. También introducen la noción de descubrimiento de patrones *fuzzy* en las reglas de asociación.

En este estudio, una regla se denomina *de calendario* si cumple con las restricciones de soporte y confianza durante toda unidad de tiempo contenida en un calendario, módulo un umbral de desfasaje. Este último modela el hecho que un regla puede cumplirse para la mayoría de las unidades de tiempo del calendario, pero no necesariamente para todas. En este caso, se dice que el calendario *pertenece* a la regla. No se requiere que la regla se cumpla para toda la base de transacciones, sino sólo para las transacciones cuyo timestamp pertenece a las unidades de tiempo especificadas por el calendario. El problema del *Descubrimiento de Reglas de Asociación de Calendarios* se define como la generación de todas las reglas de asociación, junto con sus calendarios.

El tratamiento de calendarios que se hace en este estudio, está basado en los trabajos de Allen [All85], Leban, McDonald y Forster [LMF86] y de Chandra, Segev y Stonebraker [CSS94]. Un *calendario* C se define como un conjunto de intervalos de tiempo $\{(s_1, e_1), (s_2, e_2), ..., (s_k, e_k)\}$. Se dice que C contiene la unidad de tiempo t si C contiene un intervalo (s_i, e_i) tal que $s_i \leq t \leq e_i$. Un calendario pertenece a una regla de asociación $X \Rightarrow Y$, si la regla tiene soporte y confianza suficientes para las unidades de tiempo contenidas en el calendario, con un desfasaje de *m*. Si el calendario contiene n unidades de tiempo, la regla debe cumplirse en, al menos, n –m unidades.

Reglas de Asociación Temporales en Bases de Datos Temporales (Chen, Petrounias y Heathfield)

En este estudio, Chen et al. [CPH98] investigan el descubrimiento de reglas temporales del tipo, en un ambiente médico, "algunos pacientes experimentan náuseas por alrededor de una hora, seguido por dolor de cabeza, *después de cada comida*". Para representar las reglas de asociación temporales, se introduce el uso de expresiones de tiempo calendario. Luego se plantea como objetivo del trabajo "identificar y representar ciertas formas de *reglas de asociación temporal*, desarrollar un modelo efectivo de mining y métodos para la búsqueda de esos patrones".

Los autores utilizan los conceptos de tiempo periódico, intervalos y repeticiones de Niezette y Stevenne [NS92] y Cukierman y Delgrande [CD96] para introducir las llamadas expresiones de calendario, tales como intervalos de calendarios y periódicas, en las reglas de asociación. Así, si **P** es una expresión de tiempo, una regla de asociación temporal es de la forma $\langle X \Rightarrow Y, \mathbf{P} \rangle$, donde $\Phi(\mathbf{P})$ es la interpretación de la expresión, es decir, la traducción a un conjunto de intervalos. La regla expresa que en los intervalos de $\Phi(\mathbf{P})$ la presencia de X en una transacción implica la presencia de Y en la misma transacción.

Además de soporte y confianza mínimos, los autores utilizan un umbral para el número de intervalos en que la regla se verifica, denominado *frecuencia.*

Para hallar los itemsets con suficiente soporte, utilizan una extensión del algoritmo A priori.

Reglas de Asociación Temporales Basadas en Calendarios (LI, Ning, Wang y Jajodia)

Este es, posiblemente, el trabajo más completo dentro de los que utilizan el enfoque de calendarios para definir los patrones temporales. En cierto sentido, puede decirse que incluye a los anteriormente descriptos en esta sección.
Los autores, Li et al.[LNWJ01] proponen un enfoque al descubrimiento de reglas temporales con menos requerimientos de conocimiento previo sobre los patrones temporales que [ORS98], [RMS98] y [CP98], previamente descriptos. En lugar de usar expresiones algebraicas de calendario o ciclos, utilizan los llamados *esquemas de calendario* para el descubrimiento de patrones temporales.

En [LNWJ01] un esquema de calendario está determinado por una jerarquía de conceptos de calendario y define un conjunto de *patrones de calendario*. Un ejemplo de un esquema es (año, mes, día), mientras que *todo 5° día de Enero de todo año* es un ejemplo de patrón de calendario. Vemos que un patrón de calendario se forma desde un esquema, fijando algunos componentes, como día y mes, y dejando libres otros, como año.

En este trabajo se definen dos tipos de regla temporal: precisas y fuzzy, donde estas últimas son similares a las de [RMS98]. Las primeras son las que se cumplen, es decir tienen soporte y confianza suficientes, para todo intervalo del esquema. Las reglas fuzzy son las que se cumplen, para un parámetro *m* dado, en m*100% de los intervalos.

Se define un esquema de calendario como un esquema relacional $\mathbf{R} = (f_n : D_n, f_{n-1} : D_{n-1}, \ldots, f_1 : D_1)$, donde cada atributo f_i es un nombre de una unidad de calendario, tal como año, mes, etc., mientras que los dominios D_i son subconjuntos finitos de números naturales. El esquema se

complementa con una restricción válida, la que permite excluir combinaciones que no corresponden a una interacción entre los atributos del calendario o intervalos de tiempo en los que no se está interesado. Dado un calendario **R**, un patrón de calendario sobre **R** es una tupla de la forma $<d_n, \ldots, d_1>$, donde $d_i \in D_i$ o $d_i =$ "*", el símbolo que representa a todos los valores de D_i. Por ejemplo, dado el esquema (año, mes día), el patrón de calendario <2002, *, 02> representa los intervalos de tiempo "el segundo día de todos los meses del año 2002".

Un patrón de calendario e *cubre* otro patrón e', en el mismo esquema calendario, si el conjunto de todos los intervalos de e' son un subconjunto de los intervalos de e. Un patrón con k asteriscos, con $k > 0$, se denomina un patrón k-estrella. Si $k = 0$, se denomina *intervalo de tiempo básico.*

Para descubrir las reglas de asociación temporales, los autores descomponen el problema en la forma clásica: i) hallar todos los itemsets frecuentes para todos los patrones de calendario estrella sobre el esquema de calendario dado y ii) generar las reglas usando los itemsets frecuentes y sus patrones de calendario. Como ocurre usualmente, el primer paso es el más complejo y el que mayor tiempo de ejecución insume.

El enfoque para el descubrimiento de los itemsets frecuentes en todos los patrones de calendario, se basa en Apriori. Li et al. desarrollan dos versiones de Apriori modificado, una para cada tipo de regla: precisas y fuzzy. En particular, se ha modificado la función Apriori-gen, utilizada para la generación de los itemsets candidatos.

2.3.3 Reglas de Asociación temporales: otro enfoque

Mining de Reglas Temporales Generales en un Base de Datos de Publicaciones (Lee, Chen y Lin)

En este trabajo, Lee et al. [LCL03, LLC01] definen una base de datos de publicaciones como un conjunto de transacciones, donde cada transacción es un conjunto de ítems cada uno de los cuales contiene un *período de exhibición* individual. Sobre este tipo de base de datos los autores observan que: i) una publicación antigua posee intrínsicamente mayor probabilidad de ser determinado como un itemset frecuente y ii) algunas reglas descubiertas pueden expirar en el interés de los usuarios.

En [LCL03], los autores definen las llamadas *reglas temporales generales de asociación*, como expresiones del tipo $(X \Rightarrow Y)^{t,\ n}$, donde t es el más tardío de los comienzos de los tiempos de exhibición de ambos itemsets X e Y, mientras que n es el tiempo final de la base de publicaciones. El intervalo (t, n) es el *período de exhibición máximo común* de X e Y. Una regla de asociación $X \Rightarrow Y$ se dice ser una regla de asociación temporal frecuente $(X \Rightarrow Y)^{t,\ n}$ si y sólo si su probabilidad es mayor que el soporte mínimo requerido, es decir $P(X^{t,n} \cup Y^{t,n}) > \sigma$, y la probabilidad condicional $P(Y^{t,n} \mid X^{t,n})$ es mayor que la mínima confianza establecida. Para calcular el soporte de cada itemset, se considera el número de transacciones que contienen el itemset $X^{t,n}$ dividido por el número de transacciones que contiene la base de datos de transacciones entre t y n, es decir, en el período de exhibición de X.

Dado un itemset $X^{t,n}$, éste se denomina un *itemset temporal maximal* en la base de datos parcial $bd^{t,n}$, si t es el número de la última de las particiones iniciales de todos los ítems que forman parte de X, en la base de datos **d**, y

n es el número de la última partición en $bd^{t,n}$. $X^{t+1,n}$ se dice que es un subitemset temporal de $X^{t,n}$.

Lo más significativo de este trabajo es el algoritmo denominado Progressive Partition Miner (PPM), de excelente performance, que permite descubrir este tipo de regla temporal. PPM considera que la base de datos se encuentra particionada de manera que, tanto t como n, resultan números de particiones, con n el número de la última partición. PPM no usa la propiedad de Apriori para la generación de los itemsets candidatos y logra, de acuerdo con la experimentación presentada, reducir el número de candidatos generados, con respecto al algoritmo Apriori.

Básicamente, PPM utiliza un umbral como filtro en cada partición para la generación de candidatos y procesar una partición por vez. Un conjunto progresivo de itemsets candidatos está compuesto por los dos tipos de candidatos siguientes: 1) los que resultaron trasladados desde el conjunto progresivo de candidatos previo, en el proceso de la partición anterior, y que permanecen como candidatos después de finalizado el proceso de la partición actual y 2) los candidatos que no estaban en el conjunto progresivo de candidatos de la partición anterior, sino que fueron tomados en cuenta después de procesar la parición actual. El primer tipo de candidatos se denomina tipo α mientras que los del segundo tipo se denominan tipo β. La información acumulada en las particiones previas es selectivamente utilizada para la generación de candidatos en las particiones posteriores.

A continuación describimos un bosquejo de los principales pasos de PPM:

1. Particionar la base de datos en basado en los períodos de exhibición.
2. Realizar una primer pasada a la base de datos.

3. Producir los itemsets temporales maximales candidatos de tamaño 2.
4. Usar los itemsets de tamaño 2 para generar los itemsets candidatos maximales y los subitemsets candidatos de tamaño k.
5. Realizar una segunda pasada a la base de datos.
6. Generar los k-itemsets frecuentes maximales y los k-subitemsets frecuentes.

2.4 Conclusión

En este Capítulo hemos revisado los fundamentos del Descubrimiento de Reglas de Asociación, formalizando conceptos que habitualmente se presentan de manera intuitiva. Como revisión de la bibliografía hemos expuesto una clasificación de los distintos algoritmos utilizados para hallar itemsets frecuentes.

Dado que en el Capítulo 4 utilizamos la noción de wavelets y técnicas asociadas, nos hemos referido a los fundamentos de la Transformación Wavelet Discreta.

El capítulo finaliza con un análisis detallado de diversos enfoques propuestos para el descubrimiento de reglas de asociación temporales. Se presentan los conceptos de diversos trabajos que se pueden enmarcar dentro de la categoría de calendarios. El último trabajo presentado se refiere al modelo usado para descubrir reglas de asociación temporales en bases de datos de publicaciones. Este enfoque ha sido ampliamente usado por los autores en diversas publicaciones. La finalidad de esta sección, además de describir el estado del arte, es proveer las bases para una comparación con el enfoque propuesto en esta tesis.

Capítulo 3

Descubrimiento de Reglas de Asociación Temporales: El Enfoque del Período de Vida de los Itemsets o Lifespan

En este capítulo introducimos los conceptos del modelo temporal básico. Su finalidad es resolver los problemas relacionados con la coexistencia en la base de datos de ítems de reciente aparición, con respecto al momento en que se realiza el estudio, con otros que son de aparición frecuente durante el período de registración de los datos, como asimismo ítems obsoletos cuyas reglas resultan de escaso o nulo interés. Más aún, en ciertos ambientes puede resultar prácticamente imposible encontrar ítems que aparezcan con cierta frecuencia durante el período de referencia del análisis. Este es el caso de las posiciones arancelarias aduaneras.

3.1 Introducción

El problema del descubrimiento de reglas de asociación proviene de la necesidad de descubrir patrones en datos de transacciones en un supermercado. Sin embargo, esos datos transaccionales son claramente temporales. Por ejemplo, cuando se recolectan los datos sobre los productos que se venden en un supermercado, el momento de la venta es registrado en la transacción. Este es el denominado *tiempo de transacción*, en el lenguaje de las bases de datos temporales, que además coincide con el *tiempo válido*([TCG+93]) y que corresponde al tiempo de la confirmación de la transacción de negocio en la caja. Según Kimball en

([Kim96]), l*a Dimensión Tiempo es la única dimensión que virtualmente se puede garantizar como siempre presente en todo data warehouse, porque virtualmente todo data warehouse es una serie de tiempo.*

En grandes volúmenes de datos, como los usados para propósitos de data mining, es posible hallar información relacionada con productos que no necesariamente existen durante todo el período de recolección de esos datos. De esta manera, podemos hallar productos que al momento de realizar el proceso de data mining ya han sido discontinuados. Por otra parte, puede haber también nuevos productos que fueron introducidos después de comenzar la recolección de datos. Así, podemos hallar un nuevo producto, tal como una unidad de DVD que fue introducido después del comienzo de la recolección de datos, como también un producto, tal como una unidad de disco flexible de 51/4", que ha sido discontinuado mucho antes de la finalización de la misma recolección. Algunos de esos nuevos productos podrían participar en asociaciones, pero pueden no ser incluidos en ninguna regla a causa de las restricciones de soporte establecidas por el modelo clásico de reglas de asociación [AIS93], definido de manera atemporal. Por ejemplo, si el número total de transacciones es 30.000.000 y fijamos el soporte mínimo en 0,5%, luego un producto particular debe aparecer en, al menos, 150.000 transacciones para ser considerado frecuente. Por otra parte, supongamos que esas transacciones fueron registradas durante los últimos 30 meses, a un promedio de 1.000.000 por mes. Bajo estas condiciones, tomemos un producto que ha sido vendido durante los 30 meses y cuya frecuencia de aparición llega justo al soporte mínimo establecido: aparece en promedio en 5.000 transacciones por mes. Consideremos ahora otro producto que fue incorporado en los últimos 6 meses y que aparece, en promedio, en 20.000 transacciones por mes. El número total de transacciones en que aparece es de 120.000; por esta razón este nuevo producto no es frecuente, aunque en los últimos 6 meses resulta cuatro veces más popular que el

primero. Sin embargo, si consideramos sólo las transacciones generadas desde que el producto apareció en el mercado, su soporte podría estar sobre el mínimo estipulado. En nuestro ejemplo, el soporte para el nuevo producto sería 2%, **relativo a su período de vida**, puesto que en 6 meses el total de transacciones sería alrededor de 6.000.000 y este producto aparece en 120.000 de ellas. En consecuencia, este nuevo producto y otros en similares condiciones podrían aparecer en reglas de asociación interesantes y potencialmente útiles.

Una forma de resolver el problema planteado es mediante la incorporación del tiempo en el modelo de descubrimiento de reglas de asociación. Llamaremos a estas reglas "*Reglas de Asociación Temporales*".

Un ejemplo de tal tipo de regla de asociación sería

Unidad de DVD ⇒ funda paraPalmtop, soporte= 5%, confianza= 85%, período de validez de la regla: del 15 de enero de 1999 a 25 de julio de 2003

Esta regla expresa que quienes compran una unidad de DVD *tienden a comprar* una funda para Palmtop. Esta regla se verifica en el período que media entre el 15 de enero de 1999 y el 25 de julio de 2003. Además expresa que en ese período aparece el par <unidad de DVD, funda para Palmtop> en un 5% de las transacciones y el primer componente aparece acompañado por el segundo en el 85% de las transacciones en que el primero aparece.

Este tipo de regla es pasible de ser aplicada en diversos dominios. Como un ejemplo, tomemos el caso de un grupo de pacientes que son sometidos a diversos estudios en el transcurso del tiempo. En cada momento se registran los resultados de dichos estudios, el comienzo de tratamientos, la

aparición de síntomas, etc.. Nuestro modelo de regla de asociación temporal permite capturar la vinculación o correlación entre las diversas variables, hecho que evidentemente ocurre en el tiempo, dinámicamente.

Un subproducto de la idea de incoporar el tiempo en el modelo, es la posibilidad de eliminar reglas obsoletas, de acuerdo al criterio del usuario. Además, es posible eliminar itemsets obsoletos como una función de su tiempo de vida, reduciendo de esta forma el monto de trabajo a realizar en la determinación de los itemsets frecuentes y, en consecuencia, en la determinación de las reglas.

Las reglas de asociación temporales introducidas en este capítulo, son una extensión del modelo atemporal. La idea básica es limitar la búsqueda para los conjuntos de ítems frecuentes o itemsets al período de vida de los ítems miembros del itemset. Así, cada regla tiene un marco temporal asociado correspondiente al período de vida de los ítems que participan en la regla. Si la extensión del tiempo de vida de una regla excede el mínimo estipulado por el usuario, analizaremos si la regla es frecuente en dicho período. A este efecto, se introduce el concepto de *soporte temporal*. Este concepto nos permite hallar reglas que, con el punto de vista tradicional de frecuencia, no sería posible descubrir.

A continuación analizaremos brevemente como el modelo planteado en este capítulo se relaciona con otros trabajos descriptos en el capítulo 2. Todos los enfoques presentados tienen la misma meta: el descubrimiento de reglas de asociación y sus períodos, o intervalos de tiempo, de validez. Nuestra propuesta fue formulada independientemente de las otras, pero comparte con ellas algunas similitudes.

Nuestro enfoque se basa en tomar en cuenta el período de vida o "*lifespan*" de los ítems, donde éste es el período transcurrido entre la primera y la

última vez que el ítem aparece en transacciones de la base de datos. Computamos el soporte de un itemset en el intervalo definido por su lifespan y definimos soporte temporal como la mínima amplitud del intervalo de tiempo que el usuario considera interesante. Es decir, ítems o itemsets cuyo período de vida es inferior al mínimo de tiempo establecido son desestimados, puesto que en la consideración del usuario no revisten ningún interés. Nuestro enfoque difiere fundamentalmente de los otros en que no es necesario definir intervalos o calendarios puesto que el lifespan es intrínseco a los datos.

El resto de este capítulo se organiza como sigue. En la sección 3.2 se introduce el modelo temporal básico; en la sección 3.3 discutimos el descubrimiento de reglas temporales, adaptando el método Apriori a modo de ejemplo, para hallar los itemsets frecuentes en su lifespan, y la generación de reglas. La sección 3.4 concluye el capítulo.

3.2 El Modelo Temporal Básico

El Modelo Temporal Básico de Reglas de Asociación Temporales se construye a partir de una serie de definiciones de sus componentes y el algoritmo que permite generar los itemsets temporalmente frecuentes. Comenzaremos definiendo la representación del tiempo a utilizar en el modelo.

Sea $T = \{ \ldots, t_0, t_1, t_2, \ldots \}$ el conjunto de tiempos, infinito numerable, sobre el cual se define un orden lineal $<_T$, donde $t_i <_T t_j$ significa que t_i ocurre antes o es anterior que t_j [TCG+93]. Asumimos que T es isomorfo a N(números naturales) y restringimos nuestra atención a intervalos cerrados $[t_i, t_j]$.

Definición 3.1: Sea $\mathbf{R} = \{A_1, \ldots, A_p\}$, donde los A_i son llamados *ítems*, **d** es una colección de subconjuntos de **R** llamada la *base de datos de transacciones*. Cada transacción **s** en **d** es un conjunto de ítems tal que $\mathbf{s} \subseteq \mathbf{R}$.

La definición de **R** incluye todo ítem de **d**, independientemente del momento en el cual este aparece. Asociado a s tenemos una marca de tiempo o *timestamp* t_s que representa el tiempo válido de la transacción **s**. Este timestamp se codifica habitualmente como Fecha y Hora; pero, en general, dependerá de la granularidad usada.

La Tabla 3.1 contiene un ejemplo de **d**, la base de datos de transacciones, donde $\mathbf{R} = \{A, B, C, D, E, F, G, H, I\}$.

Tabla 3.1

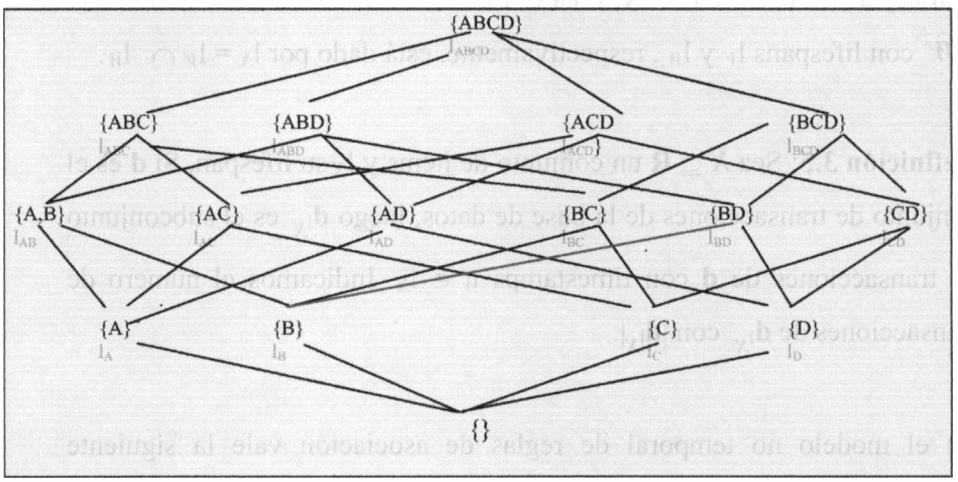
Base de Datos **d**		
T	Tid	Items
1	100	A C F H I
2	200	A B C G
3	300	B C D G I
4	400	AC I
5	500	C DEHI
6	600	ADFG

Todo ítem tiene un período de vida o *lifespan* en la base de datos, el que explícitamente representa la duración temporal de la información del ítem, es decir, el tiempo en el cual el ítem es relevante al usuario u observador. El lifespan de un ítem A_i está dado por un intervalo $[t_1, t_2]$, con $t_1 < t_2$. En **d** anterior vemos el item A que tiene lifespan [1, 6], el item B con lifespan [2, 3], etc..

Definición 3.2: Sea A_i un ítem de **R.** Con cada ítem A_i y base de datos **d**, asociamos un lifespan definido por un intervalo de tiempo $[A_i.t_1, A_i.t_2]$ o simplemente $[t_1, t_2]$ si se sobreentiende A_i. $l:A_i \rightarrow 2^T$ es una función que asigna un lifespan a cada ítem A_i en **R**. Nos referiremos a dicho lifespan como l_{Ai}. Luego, definimos l_d, el lifespan de **d**, como $l_d = \cup l_{Ai}, \forall i$.

En la figura 3.1 observamos la lattice para los itemsets que se pueden formar con **R** = {A, B, C, D} y sus lifespans asociados

Figura 3.1: Lattice de itemsets para **R** = {***A, B, C, D***} con sus lifespans



Una propiedad de un conjunto de items $X \subseteq$ **R** es el conjunto de transacciones en las que X está contenida y el cardinal de ese conjunto. Esto nos introduce a la noción de soporte que formalizamos con la siguiente definición.

Definición 3.3: Sea $X \subseteq$ **R** un conjunto de ítems, **s** *contiene* X, o X se verifica en **s**, si $X \subseteq$ **s**. El conjunto de transacciones en **d** que contiene X se

indica por $V(\boldsymbol{X}, \mathbf{d}) = \{\mathbf{s} \mid \mathbf{s} \in \mathbf{d} \wedge \boldsymbol{X} \subseteq \mathbf{s}\}$. $V(\boldsymbol{X}, \mathbf{d})$ se denomina el *conjunto de soporte*. Si la cardinalidad de $\boldsymbol{X}$ es k, $\boldsymbol{X}$ se denomina un k-itemset. Observando la Tabla 1 vemos que $V(C, \mathbf{d})$ es el conjunto de filas de **d** con identificadores 1 a 5.

Definición 3.4: El lifespan de un k-itemset $\boldsymbol{X}$, con $k > 1$, es $[t, t']$ donde

$t = \text{máx}\{t_1 \mid [t_1, t_2]$ es el lifespan de un ítem A_i en $\boldsymbol{X}\}$ y

$t' = \text{mín}\{t_2 \mid [t_1, t_2]$ es el lifespan de un ítem A_j en $\boldsymbol{X}\}$.

Puesto que las operaciones de conjuntos son válidas sobre lifespans, luego el lifespan l_X del k-itemset $\boldsymbol{X}$, donde $\boldsymbol{X}$ es la unión de los (k-1)-itemsets $\boldsymbol{V}$ y $\boldsymbol{W}$ con lifespans l_V y l_W, respectivamente, está dado por $l_X = l_V \cap l_W$.

Definición 3.5: Sea $\boldsymbol{X} \subseteq \mathbf{R}$ un conjunto de ítems y l_X su lifespan. Si **d** es el conjunto de transacciones de la base de datos, luego $\mathbf{d}_{l_X}$ es el subconjunto de transacciones de **d** con timestamps $t_i \in l_X$. Indicamos el número de transacciones de $\mathbf{d}_{l_X}$ con $|\mathbf{d}_{l_X}|$.

En el modelo no temporal de reglas de asociación vale la siguiente definición de soporte.

Definición 3.6: El *soporte* de $\boldsymbol{X}$ en **d**, indicado por $s(\boldsymbol{X}, \mathbf{d})$ es la fracción de las transacciones en **d** que contienen $\boldsymbol{X}$: $|V(\boldsymbol{X}, \mathbf{d})| / |\mathbf{d}|$. La frecuencia de un conjunto $\boldsymbol{X}$ es su soporte. Dado un umbral de soporte $\sigma \in [0, 1]$, $\boldsymbol{X}$ es frecuente si $s(\boldsymbol{X}, \mathbf{d}) \geq \sigma$. En este caso se dice que $\boldsymbol{X}$ tiene *soporte mínimo*.

En nuestro modelo temporal, deseamos ampliar la definición de soporte para incluir casos tales como los propuestos en el ejemplo de la introducción. En otras palabras, la incorporación del tiempo nos permitirá determinar si un itemset es frecuente por medio del cómputo del cociente

entre el número de transacciones que contienen el itemset y el número de transacciones en la base de datos, tal que su tiempo válido está incluido en el lifespan del itemset.

Evidentemente necesitamos filtrar ítems, y consecuentemente itemsets, con un período de vida muy corto. Por ejemplo, un ítem que se vendió en una única oportunidad sería considerado frecuente puesto que tendría un soporte de 100%, sin embargo no revestiría ningún interés por parte del usuario. Por este motivo definimos **soporte temporal** como la amplitud del lifespan de un itemset. Este parámetro permite establecer la significación de un ítem o itemset, en función de su tiempo de permanencia, por ejemplo en las ventas de productos, en el caso del análisis de canasta de mercado , o la duración de un síntoma para un paciente. Si un producto aparece en el mercado o un síntoma se manifiesta, durante un lapso muy corto, éste podría se descartado. Por esto es que necesitamos definir también un umbral para el soporte temporal: si l_d es el lifespan de la base de datos y $|l_d|$ es su duración, luego el umbral del soporte temporal τ es una fracción de $|l_d|$. Por ejemplo, si las transacciones corresponden a un período de n meses, τ, una fracción de los n meses, representa una cota inferior para el soporte temporal de un itemset.

Si la cantidad de transacciones de la base de datos es $|\mathbf{d}|$, luego $|\mathbf{d}| \cdot \tau / |l_d|$ nos daría una aproximación al mínimo número de transacciones a ser considerado como tamaño de la muestra. Luego $|\mathbf{d}| \cdot \tau / |l_d|$ debería ser un valor estadísticamente significativo, según el criterio del usuario.

Por otra parte, el usuario podría especificar un instante t_o, tal que cualquier ítem cuyo lifespan es $[t_1, t_2]$ y $t_2 < t_o$, es considerado obsoleto.

La siguiente es una nueva definición de soporte, de aplicación en nuestro modelo temporal:

Definición 3.7: El *soporte* de X en **d** sobre su lifespan l_X, que indicamos con $s(X, l_X, \mathbf{d})$, es la fracción de transacciones en **d** que contienen X durante el intervalo de tiempo correspondiente a l_X: $|V(X, \mathbf{d})| \ / \ |\mathbf{d}_{l_X}|$. La frecuencia de un conjunto X es su soporte.

Dado un umbral de soporte $\sigma \in [0, 1]$ y un umbral de soporte temporal τ, X es frecuente en su lifespan l_X si $s(X, l_X, \mathbf{d}) \geq \sigma$ y $| l_X | \geq \tau$. En este caso se dice que X tiene soporte mínimo en l_X.

El umbral de soporte o frecuencia σ es un parámetro dado por el usuario y depende de la aplicación. De la misma forma, el umbral de soporte temporal τ es dado por el usuario y también depende de la aplicación.

A efectos de clarificar las definiciones dadas, veamos ahora un ejemplo completo:

Ejemplo 3.1: sea **R** = { *A, B, C, D, E, F, G, H, I*} y **d** la misma de Tabla 1, con las siguientes seis transacciones:

$s_1 = \{A, C, F, H, I\}$, t: 1
$s_2 = \{A, B, C, G\}$, t: 2
$s_3 = \{B, C, D, G, I\}$, t: 3
$s_4 = \{A, C, I\}$, t: 4
$s_5 = \{C, D, E, H, I\}$, t: 5
$s_6 = \{A, D, F, G\}$, t: 6

Si establecemos como soporte mínimo $\sigma = 0.45$ y soporte temporal mínimo $\tau = 3$, podemos hallar los X frecuentes, lo que resulta en

$X_1 = \{A\}$, $l_{X_1} = [1,6]$,

puesto que encontramos A entre s_1 y s_6, donde éstas tienen las marcas de tiempo 1 y 6, respectivamente. El soporte de $\{A\}$ se computa como $s(\{A\}, l_{\{A\}}, \mathbf{d}) = |V(\{A\}, \mathbf{d})| / |\mathbf{d}_{[1,6]}| = 4 / 6 = 0.67$; el soporte temporal de A es $|l_A| = 6$. Luego $X_1 = \{A\}$ es frecuente porque $s(\{A\}, l_{\{A\}}, \mathbf{d}) = 0.67 > \sigma$ y $|l_A| = 6 > \tau$.

De la misma manera se obtienen los siguientes itemsets frecuentes; indicaremos para cada uno solamente sus lifespans:

$X_2 = \{C\}$, $l_{X_2} = [1,5]$;

$X_3 = \{D\}$, $l_{X_3} = [3, 6]$;

$X_4 = \{G\}$, $l_{X_4} = [2,6]$;

$X_5 = \{I\}$, $l_{X_5} = [1,5]$

$X_6 = \{A, C\}$, $l_{X_6} = [1, 5]$;

$X_7 = \{C,D\}$, $l_{X_7} = [3, 5]$;

$X_8 = \{C, I\}$, $l_{X_8} = [1, 5]$.

El conjunto vacío $\varnothing$ es trivialmente frecuente, luego no se lo considera en absoluto ya que no resulta de ningún interés.

Una regla de asociación temporal expresa que un conjunto de ítems tiende a aparecer junto con otro conjunto de ítems en las mismas transacciones en un marco temporal específico.

Definición 3.8: Una *Regla de Asociación Temporal* para **d** es una expresión de la forma $X \Rightarrow Y$ $[t_1, t_2]$, donde $X \subseteq \mathbf{R}$, $Y \subseteq \mathbf{R} \setminus X$, y $[t_1, t_2]$ es un marco temporal correspondiente al lifespan de $X \cup Y$ expresado en una granularidad determinada por el usuario.

Una regla de asociación temporal tiene tres factores asociados: *soporte*, *soporte temporal*, ambos ya definidos, y *confianza*, que definiremos a continuación.

Definición 3.9: La *confianza de una regla* $X \Rightarrow Y$ $[t_1, t_2]$, indicada por $conf(X \Rightarrow Y, [t_1,t_2], \mathbf{d})$ es la probabilidad condicional que una transacción de **d**, seleccionada aleatoriamente en el período de tiempo $[t_1, t_2]$, que contiene X también contiene Y:

$$conf(X \Rightarrow Y, [t_1, t_2], \mathbf{d}) = s(X \cup Y, l_{X \cup Y}, \mathbf{d}) \ / \ s(X, l_{X \cup Y}, \mathbf{d}),$$

donde $l_{X \cup Y} = \{[t_1, t_2]\}$.

Definición 3.10: La regla de asociación temporal $X \Rightarrow Y$ $[t_1, t_2]$ *se verifica* en **d** con *soporte s, soporte temporal* $|l_{X \cup Y}|$ y *confianza* c si s% de las transacciones de **d** contienen $X \cup Y$ y c% de las transacciones de **d** que contienen X también contienen Y, en el período de tiempo $[t_1, t_2]$.

Dado un conjunto de transacciones **d** y niveles mínimos de soporte, soporte temporal y confianza, el problema del descubrimiento de reglas de asociación temporales es generar todas las reglas de asociación que tienen al menos el soporte, soporte temporal y confianza dados.

Ejemplo 3.2: continuando con el ejemplo previo, supongamos que se ha establecibhdo el nivel de confianza mínima θ en 0.7. A partir del conjunto frecuente $\{A, C\}$ podemos considerar dos reglas posibles: $A \Rightarrow C$ [1,5] y $C \Rightarrow A$ [1,5]. La primera tiene confianza $conf(A \Rightarrow C, [1,5], \mathbf{d}) = (3/5) / (3/5) = 1.0$ que es superior al mínimo establecido $\theta = 0.7$. La segunda tiene confianza $conf(C \Rightarrow A, [1,5], \mathbf{d}) = (3/5) / (5/5) = 0.6$, de manera que resulta descartada.

En la siguiente sección abordaremos el problema de la generación de reglas temporales como una extensión del problema clásico de reglas atemporales.

3.3 Descubrimiento de Reglas Temporales

El descubrimiento de todas las reglas de asociación en un conjunto **d** de transacciones, se puede realizar en dos fases [AIS93]:

Fase 1. Hallar todo conjunto de ítems o itemset $X \subseteq \mathbf{R}$ tal que es frecuente, es decir, su frecuencia es gual o excede el soporte mínimo establecido σ.

Fase 2. Usar los itemsets frecuentes X para hallar las reglas: verificar, para todo $Y \subset X$ con $Y \neq \varnothing$, si la regla $X \setminus Y \Rightarrow Y$ se satisface con suficiente confianza, es decir, iguala o excede la mínima confianza establecida θ.

A continuación, introduciremos una serie de modificaciones a las dos fases enunciadas, a fines de soportar convenientemente el descubrimiento de reglas de asociación temporales.

Fase 1T. Hallar todo itemset $X \subseteq \mathbf{R}$ tal que X es *frecuente* en su lifespan l_X, es decir, $s(X, l_X, \mathbf{d}) \geq \sigma$ y $|l_X| \geq \tau$.

Fase 2T. Usar los itemsets frecuentes X para hallar las reglas: verificar, para todo $Y \subset X$ con $Y \neq \varnothing$, si la regla $X \setminus Y \Rightarrow Y$ $[t_1, t_2]$ se satisface con confianza suficiente, en otras palabras, iguala o excede la confianza mínima establecida θ en el intervalo $[t_1, t_2]$.

La Fase 1T, al igual que en el caso no temporal, insume la mayor parte de los costos computacionales y se describe en la siguiente sección.

3.3.1 Generación de los Itemsets Frecuentes

Cualquiera de los algoritmos propuestos en la literatura [AS94, BMU97, PCY95] para el descubrimiento de reglas de asociación, puede ser modificado convenientemente para su aplicación a reglas de asociación temporales.

Veamos, por ejemplo, el algoritmo *Apriori* de [AS94] en el que vamos a introducir pequeños cambios que nos permitirán descubrir reglas de asociación tomando en cuenta el tiempo. Como en la notación original, L_k representa el conjunto de los k-itemsets frecuentes. Cada miembro de este conjunto tendrá los siguientes atributos componentes:

i) itemset,
ii) limites inferior y superior del intervalo de tiempo de vida del ítem: t_1 y t_2,
iii) cuenta de soporte (Fr) del itemset en $[t_1, t_2]$ y
iv) número total de transacciones (FTr) halladas en el intervalo $[t_1, t_2]$.

C_k es el conjunto de k-itemsets candidatos; en otras palabras, los itemsets potencialmente frecuentes que tienen asociada la misma información que los miembros de L_k.

La figura 3.2 muestra un bosquejo del algoritmo Apriori modificado, que llamaremos Apriori Temporal:

Figura 3.2: Algoritmo Apriori Temporal

1. L_1 = { itemsets frecuentes de tamaño 1}; /*para cada itemset de tamaño 1 registramos el tiempo de su primera aparición en t_1 y el tiempo de su última aparición en t_2; en FTr contamos el número de transacciones registradas en el intervalo $[t_1, t_2]$ y eliminamos el ítem si éste resulta obsoleto */
2. **for** ($k = 2$; $L_{k-1} \neq \emptyset$; k ++) **do begin**
3. C_k = *apriori-gen*(L_{k-1}); /* nuevos candidatos con su lifespan asociado */
4. **foreach** transacción $s \in d$ **do begin**
5. C_s = *subset* (C_k , **s**); /*candidatos c en **s** y tal que el timestamp t de **s** está en el intervalo $[t_1, t_2]$ de c*/
6. **foreach** candidato $c \in C_s$ **do**
7. c.Fr ++;
8. **foreach** candidato $c \in C_k$ **do** /* tal que el timestamp t de **s** está en el intervalo $[t_1, t_2]$ de c */
9. actualizar c.FTr;
10. **end**
11. $L_k = \{c \in C_k \mid (c.Fr \geq \sigma.FTr) \wedge (c.t_2 - c.t_1 + 1 \geq \tau)\}$
12. **end**
13. Respuesta = $\cup_k L_k$;

L_1 se obtiene en la primer pasada, en la que se cuenta la ocurrencia de los ítems individuales a efectos de determinar los 1-itemsets frecuentes. Para cada itemset almacenamos su lifespan $[t_1, t_2]$. Además de contar la frecuencia absoluta para cada itemset, Fr, contamos también el número

total de transacciones entre t_1 y t_2, FTr. Luego, si Fr / FTr ≥ soporte minimo σ y si $t_2 - t_1 \geq$ soporte temporal mínimo τ, diremos que el itemset ***X*** tiene soporte mínimo en $[t_1, t_2]$.

Algunos ítems pueden haber sido eliminados de L_1 porque resultaron obsoletos, es decir, tienen un lifespan con intervalo $[t_1, t_2]$ y $t_2 < t_o$. Después de eliminar los ítems obsoletos, el siguiente lema nos asegura que no es necesario verificar, en adelante, la existencia de k-itemsets obsoletos con $k > 1$.

Lema 3.1: *Un k- itemset, con k > 1, es obsoleto si y sólo si contiene, al menos, un ítem obsoleto.*

Demostración: Se ***X*** un k-itemset, con $k > 1$.

i) Sea ***X*** un k-itemset obsoleto, es decir, su lifespan se corresponde con un intervalo $[t_{1X}, t_{2X}]$ y $t_{2X} < t_o$. Por definición 3.4, se cumple $t_{2X} = \text{mín}\{t_2 \mid [t_1, t_2]$ es el lifespan de un ítem A_j en $X\}$. Sea A el ítem cuyo t_2 es el mínimo de la expresión anterior. Luego $t_{2X} = t_2 < t_o$, de donde resulta que A es obsoleto, con lo que probamos la primer implicación.

ii) Para probar la segunda implicación, consideremos un ítem A contenido en ***X*** y A obsoleto. Supongamos, sin pérdida de generalidad, que A es el único ítem obsoleto de ***X***. Si $[t_{1A}, t_{2A}]$ es el intervalo del lifespan de A, entonces $t_{2A} < t_o$, puesto que A es obsoleto, y, además, t_{2A} es el mínimo de los t_2's, entre todos los componentes de ***X***. Nuevamente por definición 3.4, $t_{2X} = t_{2A} < t_o$ de donde *X* resulta obsoleto.

□

Continuando con la línea principal del algoritmo, toda pasada siguiente k consiste de dos fases: en la primera se obtienen los itemsets candidatos C_k de tamaño k, en base a los itemsets frecuentes de tamaño k - 1, contenidos en L_{k-1} , obtenidos en la pasada k-1, por medio de la función *apriori-gen*. El lifespan de un k-itemset con k > 1 se obtiene de la siguiente forma: si el k-itemset **u** es obtenido combinando los k-1-itemsets **v** y **w**, luego el lifespan de **u** es $[\mathbf{u}.t_1, \mathbf{u}.t_2]$, con $\mathbf{u}.t_1 = \max\{\mathbf{v}.t_1, \mathbf{w}.t_1\}$ y $\mathbf{u}.t_2 = \min\{\mathbf{v}.t_2, \mathbf{w}.t_2\}$. En la segunda fase se lee la base de datos de transacciones para computar el soporte de los itemsets candidatos de C_k. Para esto último se usa la función *subset; ésta* determina si cada miembro c de C_k está contenido en la transacción **s**. El timestamp t de **s** debe satisfacer $t \in l_c$. El algoritmo realiza tantas pasadas sobre la base de datos como la cardinalidad máxima de un itemset candidato.

Generación de candidatos *a priori*

La generación de los candidatos se realiza por medio de la función *apriori-gen*. Esta función toma como argumento L_{k-1}, el conjunto de todos los (k-1)-itemsets frecuentes y devuelve un súper conjunto de los k-itemsets frecuentes, cada uno con su lifespan asociado, representado genéricamente por el intervalo $[t_1, t_2]$. Esta función ha sido organizada en dos pasos: *Paso de Junta* y *Paso de Poda.*

1. Paso de Junta:

Esta operación se describe por medio de la siguiente expresión tipo SQL

insert into C_k

select $p.item_1, p.item_2, ..., p.item_{k-1}, q.item_{k-1}, t_1, t_2$ /* cada itemset resultante tendrá un intervalo asociado $[t_1, t_2]$ tal que $t_1 = \text{máx}\{p.t_1, q.t_1\}$ y $t_2 = \text{mín}\{p.t_2, q.t_2\}$ */

from L_{k-1} p, L_{k-1} q
where $p.item_1 = q.item_1$ *and* ... *and*
$p.item_{k-2} = q.item_{k-2}$ *and*
$p.item_{k-1} < q.item_{k-1}$;

En el próximo paso (*poda*), se eliminan todos los itemsets candidatos $c \in C_k$ tal que cualquier subconjunto de c con k-1 ítems no está en L_{k-1} o la intersección de su lifespan con el lifespan de c resulta $< \tau$. De la misma forma, los itemsets c tal que $|l_c| < \tau$ son también eliminados.

2. Paso de Poda:

foreach itemset $c \in C_k$ **do**
if $|l_c| < \tau$ **then**
eliminar c from C_k
else
foreach (k-1) subsets s in c **do**
if $s \notin L_{k-1}$ **then**
eliminar c from C_k;

Ejemplo 3.3: Dado **R** = {A, B, C, D, E, F, G, H, I} y una base de transacciones **d**, usaremos la versión temporal de *Apriori* para determinar todos los conjuntos frecuentes de ítems de **R** en **d**. Asumimos que el soporte mínimo se ha fijado en $\sigma = 0.4$ y el soporte temporal mínimo es $\tau = 3$. En la primer pasada se obtiene L_1; cada ítem original tiene definido un lifespan que ha sido observado en la lectura de la base de datos. A partir de ahora, el lifespan de los itemsets será calculado como una función del lifespan de los itemsets componentes en la función Apriori-gen (Paso de Junta).

Las figuras 3.3 y 3.4 muestran el ejemplo en detalle.

Figura 3.3: Ejemplo 3.3 (primer parte)

Base de Datos **d**		C1			L1		
T Tid Items		Itemset	soporte	LS	Itemset	soporte	LS
1 100 A C F H I		A	0.67	[1,6]	A	0.67	[1,6]
2 200 A B C G		B	1.0	[2,3]	C	1.0	[1,5]
3 300 B C D G I	scan **d**	C	1.0	[1,5]	D	0.75	[3,6]
4 400 AC I	---------->	D	0.75	[3,6]	G	0.6	[2,6]
5 500 C DEHI		E	1.0	[5,5]	H	0.4	[1,5]
6 600 ADFG		F	0.33	[1,6]	I	0.8	[1,5]
		G	0.6	[2,6]			
		H	0.4	[1,5]			
		I	0.8	[1,5]			

C_2		C_2			L_2		
Itemset		Itemset	soporte	LS	Itemset	soporte	LS
{A,C}		{A,C}	0.6	[1,5]	A,C}	0.6	[1,5]
{A,D}		{A,D}	0.25	[3,6]	{A,G}	0.4	[2,6]
{A,G}		{A,G}	0.4	[2,6]	{A,I}	0.4	[1,5]
{A,H}		{A,H}	0.2	[1,5]	{C,D}	0.67	[3,5]
{A,I}		{A, I}	0.4	[1,5]	{C,G}	0.5	[2,5]
{C,D}		{C,D}	0.67	[3,5]	{C,H}	0.4	[1,5]
{C,G}	scan **d**	{CG}	0.5	[2,5]	{C,I}	0.8	[1,5]
{C,H}	---------->	{C,H}	0.4	[1,5]	{D,G}	0.5	[3,6]
{C, I}		{C, I}	0.8	[1,5]	{D, I}	0.67	[3, 5]
{D, G}		{D,G}	0.5	[3,6]	{H, I}	0.4	[1,5]
{D,H}		{D, H}	0.33	[3,5]			
{D, I}		{D,I}	0.67	[3,5]			
{G, H}		{G, H}	0.0	[2,5]			
{G, I}		{G,I}	0.25	[2,5]			
{H, I}		{H,I}	0.4	[1,5]			

Figura 3.4: Ejemplo 3.3 (continuación)

C_3		C_3			L_3		
Itemset		Itemset	soporte	LS	Itemset	soporte	LS
{A,C,G}		{A,C,G}	0.25	[2,5]	{A,C,I}	0.4	[1,5]
{A,C,I}		{A,C,I}	0.4	[1,5]	{C,D,I}	0.67	[3,5]
{C,D,G}	scan **d**	{C,D,G}	0.33	[2,3]	{C,H,I}	0.4	[1,5]
{C, D, I}	-------->	{C, D, I}	0.67	[3, 5]			
{C,H,I}		{C,H,I}	0.4	[1,5]			

C_4
Itemset
∅

3.3.2 Generación de Reglas

Para generar las reglas de sociación temporales, es necesario hallar todos los subconjuntos para todo itemset frecuente [AS94]. Luego, dado un itemset frecuente $\boldsymbol{Z}$, debemos hallar, para cada subconjunto propio $\boldsymbol{X}$ de $\boldsymbol{Z}$, las reglas $\boldsymbol{X} \Rightarrow (\boldsymbol{Z} - \boldsymbol{X})$ $[t_1, t_2]$ tal que $s(\boldsymbol{Z}, l_{\boldsymbol{Z}}, \mathbf{d}) / s(\boldsymbol{X}, l_{\boldsymbol{Z}}, \mathbf{d}) \geq \theta$.

Uno de los problemas que encontramos en el cálculo de la confianza, de acuerdo con la definición 9,

$$conf(\boldsymbol{X} \Rightarrow \boldsymbol{Y}, [t_1, t_2], \mathbf{d}) = s(\boldsymbol{X} \cup \boldsymbol{Y}, l_{\boldsymbol{X} \cup \boldsymbol{Y}}, \boldsymbol{d}) / s(\boldsymbol{X}, l_{\boldsymbol{X} \cup \boldsymbol{Y}}, \boldsymbol{d}),$$

donde $l_{\boldsymbol{X} \cup \boldsymbol{Y}} = \{[t_1, t_2]\}$, es la determinación de $s(\boldsymbol{X}, l_{\boldsymbol{X} \cup \boldsymbol{Y}}, \mathbf{d})$. Evidentemente, $s(\boldsymbol{X}, l_{\boldsymbol{X} \cup \boldsymbol{Y}}, \mathbf{d})$ puede ser igual a $s(\boldsymbol{X}, l_{\boldsymbol{X}}, \mathbf{d})$, puesto que $l_{\boldsymbol{X} \cup \boldsymbol{Y}} \subseteq l_{\boldsymbol{X}}$. Pero en la Fase 1T habíamos computado $s(\boldsymbol{X}, l_{\boldsymbol{X}}, \mathbf{d})$, y no $s(\boldsymbol{X},$

$l_{X \cup Y}$, **d**). Puesto que, si ***XY*** es un itemset frecuente de tamaño k tendremos 2^k subconjuntos posibles, deberíamos recomputar la frecuencia para 2^k-2 itemsets in $l_{X \cup Y}$, y repetir este cálculo en cada k-ésima pasada, con k > 1. Una forma de evitar esto es usar una estimación en su lugar. En el caso más simple, si consideramos que todos los itemsets ***X*** tienen una distribución temporal uniforme, la probabilidad de aparición de ***X*** en cualquier subconjunto de l_X, en particular en $l_{X \cup Y}$, será siempre la misma. Luego es posible estimar s(***X***, $l_{X \cup Y}$, **d**) por medio de s(***X***, l_X, **d**).

A modo de ejemplo consideremos el itemset frecuente CDI de Figura 1. Desde este itemset podemos generar la regla CI $\Rightarrow$ D [3, 5]. Supongamos nuevamente que $\theta = 0.7$. Para determinar si la regla es válida, computamos su confianza como 0.67 / 0.8 = 0.84. El soporte de CI que utilizamos para el cómputo es 0.8; pero dicho valor corresponde al intervalo [1, 5] y no el [3, 5] que ahora se requiere. Bajo el supuesto simplificador de frecuencia uniforme, el computo realizado sería válido. Sin embargo, tratándose de reglas temporales, consideramos que precisamente las variaciones temporales son de interés. Por lo que habría que recalcular el soporte de los diversos subconjuntos de cada posible regla. Este es uno de los problemas que exhibe este modelo y que se resuelve en el Modelo Temporal Generalizado del Capítulo 4.

La modificación de un algoritmo tal como [AS94] para obtener todas las reglas posibles, dado un itemset frecuente Z, es inmediata.

3.4 Conclusión

En este capítulo, hemos introducido el tiempo en el problema del descubrimiento de reglas de asociación, dando lugar a lo que llamamos

Reglas de Asociación Temporales. Cada ítem, itemset y regla tiene ahora asociado un lifespan, el que proviene del tiempo explícitamente definido en las transacciones de la base de datos. Hemos introducido también el concepto de soporte temporal. Esto da lugar al descubrimiento de nuevas reglas que, debido a la falta del soporte necesario, no fueron descubiertas con el punto de vista tradicional. Ahora, con el concepto de tiempo, consideramos las reglas que tienen soporte suficiente en su lifespan, en la medida que también tengan soporte temporal.

Uno de los problemas relacionados al descubrimiento de reglas temporales que es frecuentemente mencionado, es el gran número de reglas que se pueden llegar a generar. Una solución es que el usuario puede establecer cuáles fechas son suficientemente antiguas, de manera que las reglas con lifespan previo a esas fechas serían consideradas obsoletas y no presentadas al usuario. Más aun, si el algoritmo usado para generar los itemsets frecuentes encuentra ítems o itemsets antiguos, los puede eliminar directamente, lo que agregaría una poda adicional, ya que reduciría el tamaño de la base de datos de transacciones.

Para mostrar la incidencia del tiempo en el monto y calidad de las reglas obtenidas, hemos extendido, a modo de ejemplo, el algoritmo A priori para generar los itemsets frecuentes.

Capítulo 4

Reglas de Asociación Temporales Generalizadas

En este capítulo introducimos el concepto de reglas de asociación temporales generalizadas, asociando a cada regla un conjunto de intervalos de tiempo. Estos intervalos corresponden al lifespan de los itemsets involucrados en la regla. Además de soporte y confianza, incluimos el lifespan de cada itemset y una serie de tiempo que expresa su comportamiento temporal. Representamos la serie de tiempo por medio de ciertas características obtenidas del análisis de wavelets u onditas, lo que permite ahorrar memoria cuando se las almacena y nos posibilita realizar consultas por similitud. Presentamos también un algoritmo eficiente para el descubrimiento de todos los itemsets frecuentes y diseñamos algunas estrategias de optimización derivadas de la temporalidad intrínseca de los itemsets. El descubrimiento de subintervalos frecuentes dentro del lifespan de cada itemset no solo extiende el modelo del Capítulo 3, sino que generaliza el modelo basado en calendarios ya que permite hallar todas las reglas las reglas que se cumplen en cualquier patron temporal.

4.1 Introducción y Motivación

El problema del descubrimiento de reglas de asociación se originó en la necesidad de descubrir patrones interesantes en datos de transacciones de un supermercado. Puesto que los datos transaccionales son temporales, esperamos hallar patrones que dependen del tiempo. Por ejemplo, cuando

recolectamos datos sobre la venta de productos en un supermercado, el tiempo de la venta-o la compra- queda estampado en la transacción.
Como agregado a la motivación planteada en el Capítulo 3, consideremos el caso de productos que se venden solamente en verano. Es posible que esos productos no sean frecuentes ni aun considerando solo su período de vida. Sin embargo, en la estación en que estos productos se venden pueden resultar altamente frecuentes y así participar en reglas útiles e interesantes. Para poder capturar esta característica es que acudimos al concepto amplio de lifespan como un conjunto de subintervalos. Luego, consideramos el caso de productos que pueden ser frecuentes solamente en algunos subintervalos estrictamente contenido en su período de vida pero no en el intervalo íntegro correspondiente a su lifespan.

Denominamos *Reglas de Asociación Temporales Generalizadas* a este nuevo tipo de reglas.

El modelo de reglas de asociación temporales introducido en el Caítulo3 es una extensión del modelo no temporal. La idea básica es limitar la búsqueda para itemsets frecuentes al tiempo de vida de los miembros del itemset. Por otra parte, para evitar considerar como frecuente a un itemset con un período de vida muy corto, por ejemplo, un ítem que se vende una sola vez, se introdujo el concepto de soporte temporal. Luego, cada regla tiene asociado un marco temporal, correspondiente al tiempo de vida de los ítems participantes en la regla. Si el período del tiempo de vida de una regla excede el mínimo estipulado por el usuario, analizamos si la regla es frecuente en ese período. Este concepto nos permite hallar reglas que, con el punto de vista tradicional de frecuencia, no sería posible descubrir.

En este capítulo, extendemos el modelo temporal anterior de la siguiente manera:

- El lifespan de un itemset puede incluir un conjunto de subintervalos. Los subintervalos son aquellos tales que el itemset dado:

    a) Tiene soporte temporal maximal y

    b) es frecuente.

- Este nuevo modelo resuelve los dos problemas siguientes:

    a) Itemsets no frecuentes en el lifespan completo, sino sólo en ciertos subintervalos

    y

    b) el descubrimiento de todo itemset frecuente en, al menos, subintervalos que resultan de la intersección de los lifespans de sus componentes. De esta manera aseguramos la validez de la propiedad de antimonotonía de Apriori [AS94]. Esta última caracteriza este tipo de regla temporal y permite su comparación con ventajas sobre otros modelos.

Este es el motivo por el cual llamamos *generalizadas* a las reglas formadas a partir de esta clase de itemsets frecuentes.

- Introducimos el concepto de historia de un itemset. Esto sirve a los siguientes propósitos:

    a) Computar los subintervalos contenidos en el lifespan de los itemsets,

b) Analizar su comportamiento durante su tiempo de vida, desde el punto de vista temporal y

c) Comparar el comportamiento de un itemset versus el comportamiento de otros itemsets. Para esto, modelamos la historia de los itemsets como series de tiempo y proponemos aplicar a esas series los conceptos de consultas por similitud. Proponemos también usar wavelets para representar las series de tiempo de un itemset, con lo que ahorramos memoria y permitimos varias clases de análisis y comparaciones.

- Definimos un algoritmo eficiente, basado parcialmente en el bien conocido DIC [BTU97], que toma en cuenta las características temporales intrínsecas de los itemsets.

El capítulo continua de la siguiente forma. La sección 4.2 compara este modelo con otros existentes. Las secciones 4.3 y 4.4 presentan el modelo temporal generalizado y el descubrimiento de reglas temporales, respectivamente. La sección 4.5 describe cómo hallar los subintervalos contenidos en el lifespan de un itemset, propone una caracterización de un itemset y discute la transformada wavelet a aplicar. Las secciones 4.6 y 4.7 plantean brevemente una representación alternativa a la de los subintervalos frecuentes y argumentamos por qué nuestras reglas son más generales que las propuestas en el modelo basado en calendarios, respectivamente. Finalmente, la sección 4.8 está dedicada a las conclusiones.

4.2 Trabajos relacionados

Los trabajos previos en data mining que incluyen aspectos temporales estuvieron relacionados al análisis de secuencias de eventos. El objetivo usual fue descubrir regularidades en la ocurrencia de ciertos eventos e interrelaciones temporales entre los diferentes eventos. Los trabajos relacionados referidos específicamente al descubrimiento de reglas de asociación temporales [ÖRS98, RMS98, LNWJ01, LCL03], han sido descriptos en el Capítulo 2 y serán comparados exhaustivamente con nuestro enfoque en el Capítulo 6.

Nuestro enfoque se basa en tomar en cuenta el período de vida o lifespan de los ítems, siendo éste el período transcurrido entre la primera y la última vez que el ítem aparece en las transacciones de la base de datos. Computamos el soporte de un itemset en el intervalo definido por su lifespan, o subintervalos contenidos en él, y definimos soporte temporal como la duración del lifespan. Consideramos la historia de un itemset, registrada en un histograma, como una serie de tiempo, teniendo así la posibilidad de realizar diferentes clases de análisis, basados en su transformada wavelet. Nuestro enfoque difiere de los otros en que no es necesario imponer intervalos o calendarios puesto que el lifespan es intrínseco a los datos.

4.3 El Modelo Temporal Generalizado

A los fines de completitud, reescribimos en esta sección unas pocas definiciones que ya fueron introducidas en el capítulo anterior.

Sea T = { t_0, t_1, t_2, . . . } un conjunto de tiempos, infinito numerable, sobre el cual definimos un orden lineal $<_T$, donde $t_i <_T t_j$ significa que t_i ocurre antes o es más temprano que t_j. Asumimos que T is isomorfo a N, los números naturales y restringimos nuestra atención a intervalos cerrados $[t_i, t_j]$.

A continuación definimos los conceptos de ítem, de carácter intensional, y de transacción, de tipo extensional.

Definición 4.1: Sea **R** = $\{A_1, \ldots, A_p\}$, donde los A_i's son llamados *ítems*, una *base de datos de transacciones* **d** es una colección de subconjuntos de **R**.

Cada transacción **s** en **d** es un conjunto de ítems tales que **s** $\subseteq$ **R**. Asociado a **s** tenemos una marca de tiempo o *timestamp* t_s que representa el tiempo válido de la transacción **s**.

Ejemplo 4.1.a: **R** = {A, B, C, D, E, F, G, H, I}. **d** es la colección de diez transacciones con tids 100, ...,1000 y timestamps 1, ..., 10, respectivamente (ver figura 4.1(a)).

Consideramos que **d** está ordenada temporalmente, es decir, en orden ascendente de sus timestamps. Todo ítem tiene un período de vida o *lifespan* en la base de datos, lo que explícitamente representa la duración temporal de la información del ítem, es decir, el tiempo en el cual el ítem es relevante al usuario. El lifespan de un ítem A_i está dado por un intervalo $[t_1, t_2]$, con $t_1 \leq t_2$, donde t_1 es el timestamp de la primera transacción en **d** que contiene A_i, y t_2 es el timestamp de la última transacción en **d** que contiene A_i.

Figura 4.1: Ejemplo 4.1 – La base de datos de transacciones y el conjunto de 1-itemsets candidatos

(a) La base de datos **d**

T	Tid	Items
1	100	ABCFHI
2	200	ABCG
3	300	CDI
4	400	ACI
5	500	DEHI
6	600	AF
7	700	BCI
8	800	CHG
9	900	BE
10	1000	ACD

b) El lifespan(LS) y soporte (Sup) para los items (o 1-itemsets), y lifespan frecuente (FLS): los subintervalos y correspondientes soportes. Este es el conjunto de itemsets candidatos.

Itemset	LS/Sup	FLS:Subintervalos/Soporte
A	[1,10],0.5	<[1,10], 0.5>
B	[1,9], 0.44	<[1,4],0.5>,<[6,9], 0.5>
C	[1,10], 0.7	<[1,10], 0.7>
D	[3,10], 0.37	<[3,6], 0.5>
E	[5,9], 0.4	--------
F	[1,6], 0.33	--------
G	[2,8], 0.29	--------
H	[1,8], 0.37	<[5,8], 0.5>
I	[1,7], 0.71	<[1,7], 0.71>

$\tau = 3$, $\sigma = 0.5$

Definición 4.2: Sea A_i un ítem de **R**, el *lifespan* de A_i es el tiempo sobre el cual éste aparece en la base de datos de transacciones **d**.

Con cada ítem A_i y base de datos **d**, asociamos un lifespan definido por un intervalo de tiempo $[A_i.t_1, A_i.t_2]$, o simplemente $[t_1, t_2]$ si asumimos A_i. $l : A_i \rightarrow 2^T$ es una función que asigna un lifespan a cada ítem A_i en **R**. Nos referiremos a este lifespan como l_{Ai}. En particular, éste pude ser definido como el intervalo minimal I (con respecto a la inclusión de conjuntos) que satisface que para cada tiempo t, si t está asociado a una transacción que contiene el ítem A, luego t está en I. Además, definimos l_d, el lifespan de **d**, como $l_d = \cup\, l_{Ai}$, $\forall i$.

Ejemplo 4.1.b: observemos, por ejemplo, l_A = [1,10], l_D = [3,10], l_F = [1,6], y así siguiendo, en Figura 4.1(b) en la columna encabezada por LS/Sup.

Figura 4.2: Ejemplo 4.1 – Los 1-itemsets frecuentes y los 2-itemsets candidatos

(a) Los 1-itemsets frecuentes, sus lifespans(LS) y sus lifespans frecuentes y correspondiente soporte.

Itemset	LS	FLS:Subintervalos/soporte
A	[1,10]	<[1,10],0.5>
B	[1,9]	<[1,4],0.5>, <[6,9],0.5>
C	[1,10]	<[1,10], 0.7>
D	[3,10]	<[3,6],0.5>
H	[1,8]	<[5,8],0.5>
I	[1,7]	<[1,7], 0.71>

(b) Los 2-itemsets candidatos, lifespans y soporte, y lifespans frecuentes y soporte.

Itemset	LS/Sop	FLS:Subintervalos / Soporte
AB	[1,9], 0.22	<[1,4],0.5>
AC	[1,10], 0.4	<[1,6], 0.5>
AD	[3,6], 0.0	
AH	[5,8], 0.0	
AI	[1,7], 0.28	<[1,4], 0.5>
BC	[1,9], 0.33	<[1,4], 0.5>
BI	[1,7], 0.28	
CD	[3,10], 0.25	
CH	[1,8], 0.25	
CI	[1,7], 0.57	<[1,7], 0.57>
DI	[3,7], 0.4	<[3,6], 0.5>
HI	[1,7], 0.28	

$\tau = 3$, $\sigma = 0.5$

Definición 4.3: Sea $\boldsymbol{X} \subseteq \mathbf{R}$ un conjunto de ítems y **s** una transacción en **d**, **s** *contiene* $\boldsymbol{X}$, o $\boldsymbol{X}$ *se verifica* en **s**, si $\boldsymbol{X} \subseteq$ s.

El conjunto de transacciones en **d** que contienen $\boldsymbol{X}$ se indica con $V(\boldsymbol{X}) = \{ \mathbf{s} \mid \mathbf{s} \in \mathbf{d} \wedge \boldsymbol{X} \subseteq \mathbf{s}\}$ (omitimos **d** a efectos de mayor claridad). Si el cardinal de $\boldsymbol{X}$ es k, $|\boldsymbol{X}| = k$, $\boldsymbol{X}$ se denomina un k-itemset.

Podemos estimar el lifespan de un k-itemset $\boldsymbol{X}$, con $k > 1$, por [t, t'] donde $t = \max\{t_1 | [t_1, t_2]$ es el lifespan de un ítem A_i en $\boldsymbol{X}\}$ y $t' = \min\{ t_2 | [t_1, t_2]$ es el lifespan de un ítem A_i en $\boldsymbol{X}\}$.

Ejemplo 4.2.a: En la Figura 4.2 (b) tenemos los 2-itemsets candidatos con sus lifespans computados, tal como $l_{AB} = [1,9]$, $l_{AD} = [3,6]$, $l_{DI} = [3,7]$, etc..

Definición 4.4: Sea $X \subseteq \mathbf{R}$ un conjunto de ítems y l_X su lifespan. Si **d** es el conjunto de transacciones de la base de datos, luego d_{l_X} es el subconjunto de transacciones de **d** cuyos timestamps $t_i \in l_X$.

Con $|d_{l_X}|$ indicamos el número de transacciones de d_{l_X}.

La incorporación del tiempo nos permite determinar si un itemset es frecuente, computando el cociente entre el número de transacciones que contienen el itemset y el número de transacciones en la base de datos, tal que su tiempo válido está incluido en el lifespan del itemset.

Definición 4.5: Dado un itemset ***X***, el *soporte temporal* de ***X*** es la amplitud del lifespan de ***X***, que indicamos con $|l_X|$.

También definimos un *umbral* para el soporte temporal: si l_d es el lifespan de la base de datos y $|l_d|$ es su duración, luego el umbral del soporte temporal τ es una fracción de $|l_d|$. Por otra parte, el usuario podría especificar un instante t_o, tal que cualquier ítem cuyo lifespan es $[t_1, t_2]$ y $t_2 < t_o$ es considerado obsoleto.

En ciertos casos, un itemset puede no ser frecuente en el intervalo correspondiente a su periodo de vida completo, sino solamente en uno o más subintervalos dentro de ese período. Luego, dicho itemset podría participar en reglas interesantes en esas porciones de su lifespan. Esos subintervalos no dependen estrictamente de los datos sino de los parámetros, es decir, el soporte temporal τ y el soporte mínimo σ, provistos por el usuario. Sin embargo, nosotros no estaremos interesados

en todo posible subintervalo, sino sólo en aquellos que son maximales con respecto al soporte temporal y son frecuentes en su marco temporal. Otra forma de definir los subintervalos es tomando los subconjuntos de unidades de tiempo contiguas, de acuerdo a la granularidad seleccionada, por ejemplo día o semanas, en los cuales el itemset tiene suficiente frecuencia y cuya unión no es menor que τ. En lo que sigue usaremos la primer opción. La segunda opción será analizada en la sección 4.5.

En el ejemplo hemos usado $\tau = 3$ y $\sigma = 0.5$. También podríamos haber fijado $t_o = 5$ y de esta manera ningún ítem hubiera sido considerado obsoleto puesto que el límite superior de sus lifespans son todos mayores que 5.

Definición 4.6: Dado un itemset X, *el lifespan frecuente* de X (fl_X) es el conjunto de subintervalos, contenidos en su lifespan, en los que el itemset es frecuente.

Podemos ejemplificar este último caso con el ítem B que tiene $fl_B = \{[1, 4], [6, 9]\}$, en figura 4.1(b).

Debido a que las operaciones de conjuntos son válidas sobre lifespans, el lifespan l_X del k-itemset X, donde X es la unión de los (k-1)-itemsets V y W con lifespans l_V y l_W, respectivamente, está dado por $l_X = l_V \cap l_W$. Esto mismo no es siempre válido para los lifespans frecuentes, debido a las restricciones de frecuencia mínima.

Definición 4.7: El *soporte* de X en **d** sobre su lifespan l_X, indicado como $s(X,l_X)$ (nuevamente omitimos **d** a los fines de mayor claridad), es el conjunto de fracciones de transacciones en **d** que contienen X en todo intervalo maximal correspondiente a l_X. Para cada subintervalo [t, t'] computamos el soporte como $|V(X, [t, t'])| / |d_{[t, t']}|$. Dado un umbral de

soporte $\sigma \in [0, 1]$ y un umbral de soporte temporal τ, X es *frecuente* en su lifespan l_X si existe al menos un subintervalo [t,t'] en l_X tal que $s(X,[t, t']) \geq \sigma$ y $|[t, t']| \geq \tau$, y [t, t'] maximal en l_X. En este caso, decimos que X tiene *soporte mínimo* en l_X.

En algunos casos, fl_X contiene un único subintervalo y éste puede ser igual o más pequeño que el período de vida de X.

El ejemplo en la figura 4.2 (a) muestra diferentes casos: el itemset A con lifespan [1,10], es frecuente en su período de vida completo y su soporte es 0.5; el itemset B con lifespan [1,9], es frecuente en dos subintervalos, esto es, [1,4] y [6,9], ambos con soporte 0.5; el ítem D con lifespan [3,10] es frecuente sólo en el subintervalo [3,6], con soporte 0.5. En la figura 4.3 (a) podemos observar, por caso, el 2-itemset AB, con lifespan [1,9], que es frecuente en [1,4] y su soporte es 0.5; el itemset DI, con lifespan [3,7] y lifespan frecuente [3,6] y soporte 0.5.

En este punto podemos dar una explicación formal de por qué la propiedad de antimonotonía de Apriori se satisface, como mencionáramos en la introducción del presente capítulo.

Teorema 4.1: *Sea X un itemset no frecuente en su lifespan l_X, luego no existe ningún itemset Y frecuente en su lifespan l_y, con $X \subset Y$.*

Demostración: Probamos el resultado por reducción al absurdo. Puesto que X no es frecuente en su lifespan, l_X no contiene ningún subintervalo [t, t'] tal que $s(X,[t, t']) \geq \sigma$ con $|[t, t']| \geq \tau$ y [t, t'] maximal en l_X. Supongamos ahora que Y es frecuente en l_y y así existe al menos un subintervalo [t1,t1'] en l_Y tal que $s(Y,[t1, t1']) \geq \sigma$ y $|[t1, t1']| \geq \tau$, y [t1, t1'] maximal en l_Y. l_X debe contener el intervalo [t1, t1'], porque cualquier transacción que contenga Y contiene X. Ahora, se sigue de la definición de soporte s que

$|V(Y, [t1, t1'])| / |d_{[t1, t1']}| \geq \sigma$. Puesto que $|V(Y, [t1, t1'])|$ representa el numero de transacciones que contiene Y y $X \subset Y$, luego $s(X,[t1, t1']) = |V(X, [t1, t1'])| / |d_{[t1, t1']}| \geq |V(Y, [t1, t1'])| / |d_{[t1, t1']}| \geq \sigma$. Esto conduce a una contradicción, puesto que hemos encontrado un intervalo ($[t1, t1'] \subset l_X$) donde $s(X,[t1, t1']) \geq \sigma$, es decir, X es frecuente y habiamos asumido que X no era frecuente en l_X. □

Hemos agregado la Tabla 4.1 con la notación principal utilizada a efectos de clarificación.

Tabla 4.1: Notación

l_X	lifespan del itemset X
$\mathbf{d}l_X$	subconjunto de transacciones en **d** con timestamps en l_X
τ	umbral para el soporte temporal
σ	umbral de soporte o soporte mínimo
θ	umbral de confianza o confianza mínima
t_o	umbral de obsolescencia
fl_X	lifespan frecuente para el itemset X
$s(X, l_X)$	soporte de X en su lifespan l_X

Estamos ahora en condiciones de definir formalmente las Reglas de Asociación Temporales Generalizadas.

Definición 4.8: Una *Regla de Asociación Temporal Generalizada* para **d** es una expresión de la forma $X \Rightarrow Y$: $fl_{X \cup Y}$, donde $X \subseteq R$, $Y \subseteq R \setminus X$ y $fl_{X \cup Y}$ es el lifespan frecuente de $X \cup Y$, en una granularidad determinada por el usuario.

Ejemplo 4.2: dado el itemset frecuente *ABC*, podemos considerara diferentes reglas tales como $AB \Rightarrow C$: {[1,4]}, $AC \Rightarrow B$: {[1,4]}, etc.. También podríamos considerar $I \Rightarrow C$: {[1,7]} y $C \Rightarrow I$: {[1,7]}

Necesitamos, además, extender la noción de confianza de una regla para el caso en que su validez se extienda a un conjunto de subintervalos. Notemos que cada uno de los intervalos contenidos en el lifespan del itemset que da origen a las reglas tiene asociado un valor de soporte; luego la confianza se traducirá también en un conjunto de valores.

Definición 4.9: La *confianza de una regla* $X \Rightarrow Y$: $fl_{X \cup Y}$, indicada por *conf*($X \Rightarrow Y$: $fl_{X \cup Y}$) es la probabilidad condicional que una transacción de **d**, aleatoriamente seleccionada en el lifespan frecuente $fl_{X \cup Y}$, que contiene X también contenga Y.

La probabalidad condicional varía de acuerdo al subintervalo considerado dentro de $fl_{X \cup Y}$. Luego ésta se expresa como:

$$conf(X \Rightarrow Y : fl_{X \cup Y}) = \{ s(X \cup Y, [t, t']) / s(X, ([t, t']) \mid [t, t'] \subseteq fl_{X \cup Y} \}$$

donde

$$fl_{X \cup Y} = \{[t_1, t_2] / |[t_1, t_2]| \geq \tau \text{ y } s(X \cup Y, [t_1, t_2]) \geq \sigma \text{ y } \neg\exists [t_j, t_k]([t_1, t_2] \subset [t_j, t_k] \text{ y } s(X \cup Y, [t_j, t_k]) \geq \sigma) \}.$$

Ejemplo 4.3: computamos la confianza de la regla $C \Rightarrow I$: {[1,7]} como

$$Conf(C \Rightarrow I : \{[1,7]\}) = s(CI, [1,7]) / s(C, [1,7]) = 0.57 / 0.71 = 0.80.$$

Definición 4.10: La regla de asociación temporal generalizada $X \Rightarrow Y$: $fl_{X \cup Y}$ *vale* en **d** con *soporte* $s_1, ..., s_p$, *soporte temporal* $|fl_{X \cup Y}|$ y *confianza* $c_1, ..., c_p$, si $s_1\%$, ..., $s_p\%$ de las transacciones de **d** en $l_{X \cup Y}$ = { subintervalo$_1$, ..., subintervalo$_p$} contiene $X \cup Y$ y $c\%_1$, ..., $c\%_p$ de las

transacciones de **d** que contienen X también contienen Y, en el conjunto de intervalos de tiempo [t, t'] tal que [t, t'] $\subseteq$ $fl_{X \cup Y}$.

Ejemplo 4.4: la regla $C \Rightarrow I$: {[1,7]} se satisface en **d** con soporte 0.57, soporte temporal 7 (la amplitud del intervalo [1,7]), y confianza 0.80.

Figura 4.3: Ejemplo 1 – Los 2-itemsets frecuentes, los 3-itemsets candidatos y los 3-itemsets frecuentes

(a) Los 2-itemsets frecuentes, sus lifespans y y lifespans frecuentes y soporte.

Itemset	LS	FLS:Subintervalos/ Soporte
AB	[1,9]	<[1,4], 0.5>
AC	[1,10]	<[1,6], 0.5>
AI	[1,7]	<[1,4], 0.5>
BC	[1,9]	<[1,4], 0.5>
CI	[1,7]	<[1,7], 0.57>
DI	[3,7]	<[3,6], 0.5>

(b) los 3-itemset candidatos, sus lifespans frecuentes y soporte.

Itemset	LS/Sup	FLS:Subintervalos/ Soporte
ABC	[1,9], 0.22	<[1,4], 0.5>
ACI	[1,7], 0.28	<[1,4], 0.5>

(c) Los 3-itemsets frecuentes con sus lifespans frecuentes

L_3={ABC:[1,4], ACI:[1,4]}

(d) No hay ningún 4-itemset candidato.

$C_4 = \varnothing$

$\tau = 3$, $\sigma = 0.5$

4.4 Descubrimiento de Reglas Temporales Generalizadas

Como ya se dijo, el descubrimiento de todas las reglas de asociación en un conjunto de transacciones **d** puede ser realizado en dos fases [AIS93]. En lo que sigue, introducimos una serie de modificaciones para adecuarlas para el descubrimiento de reglas de asociación temporales generalizadas.

Fase 1TG. Hallar todo itemset $X \subseteq \mathbf{R}$ tal que X es *frecuente* en su lifespan l_X, es decir,. $s(X,[t,t']) \geq \sigma$, $|[t, t']| \geq \tau$ y $|[t,t']|$ maximal, para $[t, t']$ en l_X, o sea, $fl_X \neq \varnothing$.

Fase 2TG. Usar los itemsets frecuentes X para hallar las reglas: verificar para todo $Y \subset X$, con $Y \neq \varnothing$, si la regla $X \setminus Y \Rightarrow Y$: $fl_{X \cup Y}$ se satisface con suficiente confianza, en otros términso, excede la confianza mínima θ establecida en el intervalo [t, t] para todo [t, t'] en $fl_{X \cup Y}$.

4.4.1 Un Algoritmo Temporal para el Descubrimiento de Itemsets Frecuentes

Cualquiera de los algoritmos propuestos en la literatura [AS94a, BMUT97, PCY95, SON95] para el descubrimiento de reglas de asociación puede ser modificado convenientemente para su aplicación en el problema de las reglas de asociación temporales. Hemos seleccionado el algortimo DIC porque usualmente **d** es muy grande y éste optimiza el número de pasadas sobre **d.** Hemos llamado a este algoritmo *Temporal Dynamic Itemset Counting.*

Algoritmo 4.1 (TDIC): Hallar los k-itemsets frecuentes en su lifespan, con k = 1, 2, ...

Entrada: la base de datos de transacciones **d,** donde cada registro contiene una transacción, la que, a su vez, tiene asociado un timestamp en base al cual **d** está ordenada en forma ascendente, σ, τ, Δt y **M**(número de transacciones de cada porción en que se divide la base de datos).

Salida: el conjunto de todo itemset frecuente en su lifespan frecuente.

Procedimiento

Como en la notación original, L_k representa el conjunto de k-itemsets frecuentes. Cada miembro de este conjunto tendrá asociados los siguientes campos:

i) Identificación del itemset.

ii) Límites inferior y superior del tiempo de vida del item.

iii) Cuenta de soporte (Fr) del itemset en $[t_1, t_2]$.

iv) Número total de transacciones (FTr) halladas en el intervalo $[t_1, t_2]$.

v) Arreglo de contadores. Cada contador se asigna a un intervalo $[t_1 + j.\ \Delta t,\ t_1 + (j+1).\ \Delta t]$, con $j = 0, 1, \ldots$. Este arreglo mantendrá un histograma con el numero de transacciones que contengan el itemset en cada intervalo. Δt es definido por el usuario y puede ser expresado en diferentes granularidades.

C_k es el conjunto de k-itemsets candidatos; en otras palabras, itemsets potencialmente frecuentes que tienen asociada la misma información que los miembros de L_k. Además, C_k usa un campo auxiliar para computar correctamente el número total de transacciones en $[t_1, t_2]$ cuando t_2 no ha sido todavía definido, y un marcador para indicar si el itemset ha completado una pasada o no.

TRAIN es el conjunto que contiene todo itemset cuyo soporte está todavía en proceso de recuento.

Tdic(**d**, M, *F*) comienza hallando los 1-itemsets y, después de leer M transacciones, analiza si son frecuentes. A continuación, combina los 1-itemsets frecuentes para formar 2-itemsets candidatos. Estos últimos son analizados a su vez al final de la siguiente porción de M transacciones

correspondientes a sus lifespans, para verificar si también son frecuentes y, en este caso, son combinados para formar 3-itemsets candidatos, y así sucesivamente. Cuando el algoritmo completa una pasada, si durante esta última generó algún itemset nuevo de tamaño mayor que los existentes al comienzo de esa pasada, se inicia nuevamente por el comienzo de **d**. Podría haber algunos itemsets no frecuentes en el lifespan completo, sino sólo en algunos subintervalos. Para esos casos usamos el algoritmo descripto en la sección siguiente, para hallar los intervalos maximales contenidos en sus lifespans y generar nuevos candidatos a partir de ellos.

Los itemsets candidatos C_k de tamaño k, basados en los itemsets frecuentes L_{k-1} de tamaño k-1 obtenidos hasta el momento, son generados por medio de la función *apriori-gen* del Capítulo 3, subsección 3.3.1, que tiene ahora dos argumentos, L_{k-1} y L'_{k-1}, donde este último contiene los itemsets frecuentes hallados en la última porcion de M transacciones. El lifespan de un k-itemset con $k > 1$ se obtiene de la siguiente forma: si el k-itemset u se obtiene combinando los k-1-itemsets v and w, luego el lifespan de u es el conjunto de intervalos

$$\{[u.t_1, u.t_2] / (\exists [v.t_1, v.t_2], [w.t_1, w.t_2])([v.t_1, v.t_2] \cap [w.t_1, w.t_2] \neq \varnothing \text{ and } u.t_1 = \max\{v.t_1, w.t_1\} \text{ and } u.t_2 = \min\{v.t_2, w.t_2\})\}.$$

Luego, se lee la base de datos de transacciones para computar el soporte de los itemsets candidatos de TRAIN, para lo cual se utiliza la función *subset*. Ésta determina si cada miembro c de TRAIN está contenido en la transacción s. El timestamp t de s debe satisfacer $t \in l_c$.

A continuación se incluye la versión en pseudocódigo del procedimiento del algoritmo TDIC:

Algoritmo Temporal Dynamic Itemset Counting

1. pass ← 1; kf ← 0; k ← 1;

```
2. while kf <> k do begin
3.      kf = k
4. while not end of d do begin
5.      read dM                 /* dM es un grupo de M transacciones de d
                                   ordenadas por sus timestamps */
6.      foreach s ∈ dM do begin
7.       if pass = 1 then
8.            check if each 1-itemset c ∈ s exists in C1 otherwise insert c
9.              in C1 and TRAIN;
10.     Cs ← subset (TRAIN, s) /* Cs contiene todo itemset c en TRAIN
                                   tal que c aparece en s */
11.     foreach c ∈ Cs do begin
12.           c.Fr ← c.Fr + 1;
13.           if s.t > c.t2 then begin
14.                 c.t2 ← s.t;
15.                 FTr ← FTr + FTr';
16.                 FTr' ← 0;
17.           endif;
18.       end;
19.     foreach c ∈ TRAIN do
20.           if s.t ∈ [c.t1, c.t2] then
21.                 c.FTr' ← c.FTr'+ 1;
22.           else
23.             if c is a 1-itemset then add 1 to c.FTr' ;
24.   end;
25. end;
26. for j = pass to k do begin
27.     LL = { c / c ∈ Cj ∧ c.Fr ≥ σ × FTr ∧ (c.t2 - c.t1 + 1) ≥ τ}
                   /* obtiene los j-itemsets frecuentes */
28.     if |Lj| > 1 then
```

29. $L_j' \leftarrow LL - L_j$
30. **else**
31. $L_j' \leftarrow LL$; /* L_j' contiene los j-itemsets nuevos */
32. $L_j \leftarrow LL$;
33. **endfor**;
34. **if** $|L_k| > 1$ **then**
35. $k \leftarrow k + 1$;
36. **for** j = pass + 1 to k **do begin**
37. $C_j' \leftarrow$ apriori-gen(L_{j-1}, L_{j-1}')
38. **If** $L_{j-1} <> L_{j-1}'$ **then**
39. $C_j' \leftarrow C_j' \cup$ apriori-gen (L_{j-1}', L_{j-1});
40. $C_j \leftarrow C_j \cup C_j'$;
41. TRAIN $\leftarrow$ TRAIN $\cup C_j'$;
42. **endfor**;
43. **foreach** c $\in$ TRAIN **do** /* este paso elimina todo itemset desde TRAIN */
44. **if** c.complete = Yes **then** /* que ha completado una pasada sobre **d** */
45. delete c from TRAIN;
46. **endwhile**;
47. pass $\leftarrow$ pass + 1;
48. a-posteriori (L_{pass});
49. **endwhile**;
50. **for** j = pass + 1 to k **do**
51. a-posteriori(L_j);
52. **endfor**;
53. $F = \cup_{i=1, \ldots, k} L_i$;
54. **end**.

a-posteriori(L_{pass}) encuentra los itemsets frecuentes tal que no son frecuentes en todo su lifespan sino sólo en algunos subintervalos maximales. Un bosquejo de este algoritmo y su version pseudocódigo se describe en la subsección 4.5.1.

Apriori-gen(L, L') es una ligera modificación de apriori-gen de [3].

A efectos de completitud probaremos que el algoritmo TDIC encuentra todos los itemsets frecuentes dentro de su lifespan, sea éste un intervalo único o un conjunto de subintervalos.

Teorema 4.2: TDIC computa correctamente todos los itemsets frecuentes en su lifespan.

Demostración: probaremos por inducción sobre k que L_K es computado correctamente para todo k. Para k = 1 es correcto que L_1 contiene exactamente los items frecuentes en su lifespan, sea en uno o más subintervalos.
Para k > 1, asumimos que L_{k-1} ha sido computado correctamente. Luego *Apriori-gen* genera un conjunto de candidatos C_K para el que $L_K \subseteq C_K$. En prueba de ello, mostramos que todo X frecuente en su lifespan pertenece a C_K.
Por Teorema 4.1 sabemos que ningún itemset de tamaño h, frecuente en su lifespan, puede contener algún itemset de tamño h – 1 no frecuente en su propio lifespan.
Dado Z y l_Z en L_K, consideremos los subconjuntos de Z de tamaño k – 1 X, con su l_X asociado, e Y con su correspondiente l_Y, donde ambos coinciden en los primeros k – 2 componentes. Luego, como X e Y están en L_{K-1}, A priori-gen genera Z en C_K. □

4.4.2 Generación de las Reglas Temporales Generalizadas

En el caso general, para generar las reglas, resulta necesario hallar todos los subconjunto para todo itemset frecuente([AS94]). Es decir, dado un itemset frecuente Z debemos hallar, para cada X subconjunto propio de Z, las reglas $X \Rightarrow (Z - X) : fl_Z$ tal que $conf(X \Rightarrow (Z - X)) \geq \theta$.

De igual manera que en la sección 3.2 del capítulo anterior, uno de los problemas que encontramos en el cómputo de la confianza, de acuerdo con la definición 4.9 de la sección anterior,

$$conf(X \Rightarrow Y : fl_{X \cup Y}) = s(X \cup Y, fl_{X \cup Y}) / s(X, fl_{X \cup Y}),$$

es la determinación de $s(X, fl_{X \cup Y})$. También en este caso, $s(X, fl_{X \cup Y})$ puede no ser igual a $s(X, fl_X)$, puesto que $fl_{X \cup Y} \subseteq fl_X$, y una alternativa es usar $s(X, fl_X)$ como una estimación de $s(X, fl_{X \cup Y})$, asumiendo una distribución temporal uniforme para X.

Observemos que, en el caso general, una regla puede tener un conjunto de valores de confianza, dependiendo del número de subintervalos contenidos en su lifespan. Además, cuando los diferentes subconjuntos de itemsets son frecuentes en subintervalos pero no en el lifespan completo, y usamos la estimación para $s(X, fl_{X \cup Y})$, la confianza resulta próxima a 1. En este caso resultará conveniente usar el histograma para estimar $s(X, fl_{X \cup Y})$. Como un ejemplo de lo recien expresado, observando el contenido de Figuras 4.1 a 4.3, analicemos el itemset AC con soporte 0.5 en el intervalo [1, 5]. Una de las dos reglas posibles es C $\Rightarrow$ A con soporte 0.5 en [1, 6]. Veamos ahora qué ocurre con el cómputo de la confianza. Si

tomáramos directamente el soporte de C, éste es 0.7, pero en [1, 10]. Si asumiéramos distribución uniforme de las apariciones de C en la base de datos de transacciones, la confianza de la regla sería 0.5 / 0.7 = 0.71. Si contáramos con el histograma para C, con $\Delta t = 1$, veríamos que C aparece 4 veces en [1, 6] por lo que su soporte en ese subintervalo es 4 / 6 = 0.67. Luego la confianza de la regla en su periodo de validez, es decir en [1, 6], resulta 0.5 / 0.67 = 0.75.

4.5 Caracterización Temporal de los Itemsets

Durante el proceso de búsqueda de los itemsets frecuentes se construyen histogramas que asociamos con cada itemset. En estos histogramas acumulamos el número de transacciones en las cuales el itemset aparece, en su período de vida completo. La amplitud del intervalo para los histogramas, llamado *unidad de tiempo*, es porporcionado por el usuario o podría ser estimado automáticamente.

Hemos usado un algoritmo, descripto en la section anteior, para descubrir todos los itemsets frecuentes. TDIC requiere, para aquellos itemsets no frecuentes en la totalidad de su período de vida, otro algoritmo que describimos más abajo. TDIC, en la búsqueda de itemsets frecuentes, genera los histogramas como un subproducto de esa tarea. El intervalo de tamaño fijo de los histogramas es la unidad de tiempo dada. Para los itemsets no frecuentes, empleamos el algoritmo *a-posteriori* a efectos de hallar los intervalos frecuentes maximales que están estrictamente contenidos en sus periodos de vida. A continuación damos un bosquejo de este último algoritmo.

4.5.1 Algoritmo *a-posteriori*

Algoritmo 4.2: Hallar los intervalos frecuentes maximales en l_X, para un itemset X, esto es lf_X.

Entrada: arreglo S con información recolectada en el histograma, arreglo Q con total de transacciones para toda unidad de tiempo, τ y σ.

Salida: conjunto de intervalos frecuentes maximales, lf_X. Este conjunto puede ser vacío, en cuyo caso el itemset no es frecuente en absoluto.

Procedimiento:
Buscamos S desde el final hacia el principio, tratando de hallar intervalos tales que el total del número de transacciones acumuladas que contengan, dividido el número de las transacciones correspondientes de Q resulte mayor que σ. Si hallamos un intervalo satisfactorio, marcamos el final de un intervalo y buscamos nuevamente desde el final del arreglo hasta la marca del intervalo anterior; de lo contrario avanzamos la marca una unidad y repetimos el proceso.

También para este algoritmo incluimos a continuación la versión en pseudocódigo del procedimiento.

Algoritmo para hallar los subintervalos frecuentes maximales (a-posteriori)
/* el intervalo unitario número 1 de S se corresponde con el intervalo unitario j de Q. sum1 es la suma de los intervalos del histograma. sum2 es el número total de transacciones en el mismo período de tiempo */
/* el histograma tiene n intervalos unitarios */
h ← 1; g ←0;
while n ≥ h **do begin**

$m \leftarrow 0$; $g \leftarrow g + 1$;

sum1 $\leftarrow \Sigma_{i = h \text{ to } n} S(i)$;

sum2 $\leftarrow \Sigma_{k = j + h - 1 \text{ to } j + n - 1} Q(k)$;

fr $\leftarrow$ sum1 / sum2;

while fr $< \sigma$ and $(n - m) * |\text{TimeUnit}| \geq \tau$ **do begin**

fr $\leftarrow$ (sum1 - S(n - m)) / (sum2 - Q(j + n - m))

$m \leftarrow m + 1$;

endwhile

if $(n - m) * |\text{TimeUnit}| \geq \tau$ **then begin**

hl $\leftarrow$ h; h $\leftarrow$ n - m; hs $\leftarrow$ h;

$\text{Interval}_g \leftarrow$ <[lwr bound T. Unit j + hl - 1, upper bound T. Unit j + hs - 1], fr>;

$m \leftarrow 0$;

else $h \leftarrow h + 1$;

endwhile.

4.5.2 Representación y uso de los histogramas

Los histogramas describen, en una cierta medida, el comportamiento temporal de los itemsets. De hecho, podemos considerar los histogramas de los itemsets como series de tiempo, los que resultan patrones útiles para explorar. En esta sección describimos en forma suscinta esta idea y definimos una transformación de los datos, con una doble finalidad: primero, ahorrar memoria usando una representación más compacta de las series de tiempo y segundo, con esta clase de representacón, obtener una caracterización para esas series de manera de poder comparar una con otra, haciendo posible la búsqueda de comportamientos similiares entre los itemsets.

Consideramos una unidad de tiempo Δt, tal como día, semana, mes o año, y un itemset $\boldsymbol{X}$ con un intervalo $[t_1, t_2]$ asociado, correspondiente al lifespan del itemset. La tarea de acumular el número de transacciones que contienen el itemset deseado en diferentes puntos en el tiempo produce una secuencia que puede ser definida por un conjunto de pares $\{(v_1, f_1), ..., (v_p, f_p)\}$, donde $v_1 = t_1$, $v_2 = v_1 + \Delta t$, ..., $v_p + \Delta t = t_2$, y f_i es el numero de transacciones que contienen $\boldsymbol{X}$ en el período entre v_i y $v_i + \Delta t$. Luego, consideramos $F_{\boldsymbol{X}} = f_1, f_2, ..., f_p$ la serie de tiempos asociada con el itemset $\boldsymbol{X}$.

Después de ejecutar nuestros algoritmos obtenemos todos los itemsets frecuentes más sus series de tiempo. Sobre esas secuencias podríamos realizar otras tareas de minado de datos para responder a interrogantes como los siguientes[GK95]:

- ¿Cuáles son los itemsets que exhiben un comportamiento similar que el del itemset Z?
- ¿Cuáles son los meses, o cualquier otro período de tiempo, que muestran patrones similiares a un particular mes para el itemset Z?
- ¿Cuáles itemsets han mostrado en los últimos dos años patrones similares al itemset Z ?
- ¿Cuáles itemsets ha mostrado en cuáles meses en los últimos dos años patrone similares al itemset Z en este mes?

Un enfoque simple es considerar la distancia euclideana entre dos series de tiempo, por ejemplo F_X y F_Y, y las llamamos similares si E_Distancia(F_X, F_Y) es menor que un cierto umbral. Pero las series de tiempo son habitualmente extensas, de manera que el cómputo de la distancia resulta demasiado largo. Para responder esos interrogantes y evitar los problemas mencionados, se han propuesto varios enfoques[AFS93, GK95, RM97]. Éstos resuelven este otro tipo de tarea de

data mining, conocida como *consultas por similitud para datos de series de tiempo,* como sigue. La idea básica es caracterizar cada serie de tiempo por medio de *k* atributos. Luego, toda secuencia resulta un punto en un espacio *k*-dimensional. Extrayendo las k características o atributos de toda secuencia, podemos mapear las secuencias a un espacio k-dimensional. Las secuencias deberían ser mapeadas a puntos tales que la distancia euclideana en el espacio k-dimensional es menor o igual a la distancia real entre las dos secuencias. Diversos estudios [AFS93, RM97]usan la Transformada Discreta de Fourier (DFT) para la extracción de características, es decir, dado F_X transformarla a DTF(F_X) y tomar los k primeros coeficientes de Fourier como las deseadas k características. Luego, dos secuencias F_X y F_Y son similares si E_Distancia(DTF(F_X),DTF(F_Y)) < ε, donde ε es el umbral proporcionado por el usuario.

Otros interrogantes serían:

- ¿Cuál es la tendencia dada por el patrón de ventas del itemset ?
- ¿Hay algún ciclo?
- ¿Hay variaciones estacionales?

que son típicas en el análisis de series de tiempo.

En este trabajo proponemos usar *wavelets* para representar nuestras series de tiempo. Las wavelets son una familia de funciones matemáticas de bases ortogonales que satisfacen ciertos requerimientos y son usadas en la representación de datos u otras funciones. Éstas separan los datos en diferentes componentes de frecuencia y estudian cada componente con resolución acorde a su escala. El análisis de wavelets tiene ventajas con respecto a los métodos tradicionales de Fourier en los casos en que la señal contiene discontinuidades, espigas agudas y otros tipos de características no suavizadas. Las funciones wavelest están localizadas en el espacio mientras que las funciones seno y coseno de Fourier no. Esta

característica y la localización de frecuencia hace que estas funciones, cuando transformadas en el dominio wavelet, resulten dispersas. Ambas características son útiles cuando se trata de eliminar ruido de las series de tiempo.

De acuerdo a nuestros dos objetivos sólo tenemos que conservar almacenados los valores de las características que representan cada secuencia. Para esto, debemos computar la descomposición wavelet de las secuencias y conservar los *m* coeficientes más significativos correspondientes a cada descomposición. Usualmente m << p así que satisfacemos nuestro primer objetivo en relación a ahorrar memoria.

Para hallar secuencias similares a una dada, aplicamos un enfoque similar al de la Transformada de Fourier [CF99, GK95, PM02], tomando en cuenta la distancia entre secuencias y comparándola con un umbral dado. Para esto deberíamos primero normalizar las secuencias para considerar problemas de escalado y desplazamiento [GK02].

Para implementar la Transformada Wavelet, debemos primero seleccionar un conjunto de funciones base buscando un equilibrio entre compacidad y suavidad. En general, esto depende de la naturaleza de la secuencia y se denomina una *base de formas de onda adapatadas*. La construcción de las bases de wavelets puede realizarse por un *análisis multi-resolución* que consiste de una secuencia de espacios de aproximación sucesivos. En segundo término, tenemos que computar el ancho de banda de la secuencia o el número de espacios a ser transformados, yendo a través de los espacios, desde el más fino al más grueso. Si hay *p* puntos en la secuencia F_X, luego puesto que $2^{\Delta j} = p - 1$ así resulta $\Delta j = \lfloor \log_2(p - 1) \rfloor$. Luego computamos los coeficientes correspondientes a la parte gruesa y a la parte de detalle, de la secuencia original. Algunos coeficientes de la parte de detalle que son más pequeños que un umbral dado, son considerados

ruido y, en consecuencia, descartados. Podríamos aún descartar todo coeficiente de la parte de detalle si sólo necesitáramos considerar la tendencia global de la secuencia.

El número de coeficientes de la parte gruesa depende del nivel deseado de detalle. Si este nivel es j, luego el número de coeficientes es $p / 2^j$. Los coeficientes son luego insertados en una estructura de índice, tal como un R-Tree, para soportar la búsqueda de las series de tiempo coincidentes.

4.6 Otra forma de representación para los subintervalos frecuentes

Una representación alternativa a los subintervalos frecuentes maximales es la dada por la densidad de frecuencia estricta de los histogramas, para cada itemset. En este caso, los subintervalos a considerar en el lifespan son aquellos que resultan de los Δt en los que el itemset es frecuente (intervalos unitarios frecuentes) y cada subintervalo se obtiene a partir de la unión de los intervalos unitarios frecuentes contiguos

Esta otra forma de representación de los subintervalos exhibe una correspondencia directa entre los subintervalos frecuentes y los histogramas. La ventaja que esta alternativa presenta, es la posibilidad de ampliar el alcance del presente modelo temporal permitiendo la obtención inmediata de reglas temporales basadas en el modelo de calendarios. Esta posibilidad muestra que el modelo de reglas basado en lifespan es más general que el modelo basado en calendarios.

4.7 Importancia de los histogramas

Una de la metas planteadas en el descubrimiento de reglas de asociación es la de, una vez halladas las reglas, no acceder nuevamente a la base de datos de transacciones más allá de lo estrictamente necesario.

Al finalizar la generación de los itemsets nuestro algoritmo produce, como ya se dijo en la Sub-sección 4.4.1, un histograma para cada uno de ellos. Si esta generación se realiza con un umbral de soporte suficientemente bajo σ_{Min}, luego es posible generar reglas con diversos umbrales, siempre que sean superiores a σ_{Min}, directamente desde los histogramas sin necesidad de recurrir a las transacciones originales. De hecho, los histogramas constituyen una representación compacta de la base de datos **d**, la que se implementa de acuerdo con las técnicas descriptas en la Sección 4.6.

Aplicar patrones de calendario correspondientes a esquemas basados en calendario, tales como los de [LNWJ01], resulta sumamente simple y de muy bajo costo. Esto consiste solo en buscar en cada histograma, fijado un umbral de soporte, cuáles son los itemsets que cumplen con el patrón dado.

4.8 Conclusión

En este capítulo hemos presentado un modelo para el descubrimiento de Reglas de Asociación Temporales Generalizadas. Cada itemset tiene un lifespan asociado, el que resulta del tiempo explícitamente definido en la base de datos de transacciones. En particular, cada itemset frecuente tiene un lifespan frecuente asociado, es decir, un conjunto de intervalos en los que resulta frecuente.

Hemos propuesto la caracterización temporal de los itemsets por medio de una serie de tiempo y su representación usando características extraídas por la aplicación de una transformada wavelet.

Hemos descripto nuestro algoritmo temporal que genera los itemsets frecuentes. Las optimizaciones estan basadas en el ordenamiento temporal de las transacciones y la definición de los itemsets. El algoritmo genera, como un subproducto, un histograma que describe la historia del itemset. También hemos propuesto otro algoritmo para buscar los subintervalos frecuentes, basado en los datos de los histogramas.

El algoritmo TDIC cuenta con dos implementaciones. La primera es una versión Java, realizada como una extensión a WEKA, y ha sido aplicada a experimentación con bases de datos relativamente pequeñas. Por otro lado tenemos una versión PL/SQL implementada sobre Oracle 9 que ha sido utilizada con bases de datos más grandes conteniendo datos sobre comercio exterior.

Cabe destacar que los usuarios potenciales de este tipo de información, analistas de comercio exterior, han hallado a la historia de los itemsets de tanto interés como las reglas. Por lo que las series temporales que representan esas historias estarían sujetas a diversos tipos de análisis, tales como los propuestos en la subsección 4.5.2.

Capítulo 5

Experimentación

Para evaluar el desempeño de TDIC y analizar el tipo de reglas que es posible descubrir, hemos conducido diversos experimentos. Usamos dos tipos de conjuntos de datos que denominamos base de datos real y base de datos sintética. Presentaremos, en primer término, la experimentación realizada con una base de datos real conteniendo información de importación de productos o posiciones arancelarias. Esta base de datos fue proporcionada por la Dirección de Aduanas de la República Argentina y contiene información del período 1995-2002. En segundo término, analizaremos resultados de la experimentación realizada sobre una base de datos sintética.

5.1 Experimentación sobre una base de datos real

Para esta experimentación se ha utilizado un computador con un procesador Pentium IV de 1.5 GHZ, 256 MB de memoria y un disco de 40 GB. El software utilizado fue el sistema operativo Windows 2000, el sistema de gestión de bases de datos Personal ORACLE 9i, Java 1.2.2 y la aplicación se desarrolló en PL/SQL.

Comenzaremos mostrando el tipo de reglas descubiertas. La siguiente es una regla cuyo lifespan está definido por un conjunto de intervalos frecuentes:

Pasadores Clavijas Chavetas $\Rightarrow$ ***Tuercas***
Sop = {0.0402, 0.0408}, conf = {88%, 93%}
lifespan: {[22/1/98, 6/2/98],[27/3/98, 3/4/98]}

Esta regla expresa que las órdenes de importación que incluyen pasadores, clavijas y chavetas, de un cierto tipo, tienden a incluir también tuercas, igualmente de un determinado tipo. Esta regla se verifica en dos intervalos frecuentes: el primero desde el 22/1/98 al 6/2/98, período en el que exhibe un soporte de 4.02% y una confianza de 88%, y el segundo desde 27/3/98 al 3/4/98, cuando su soporte fue de 4.08% y su confianza de 93%.

Los datos utilizados para este experimento representan importaciones realizadas en la Aduana Argentina (con excepción de las aduanas de Ezeiza y de Buenos Aires) durante el período 24/05/1995 – 12/01/2002. Para caracterizar las importaciones tomamos en cuenta los siguientes datos:

- fecha en que se produjo la importación
- posiciones arancelarias involucradas en la misma

La posición arancelaria es una codificación internacional que se utiliza para describir la mercadería. Por ejemplo la posición arancelaria 8301.30.00.000T representa "candados, cerraduras y cerrojos (de llave, combinación o eléctricos), de metal común; cierres y monturas cierre, con cerradura incorporada, de metal común; llaves de metal común para estos artículos". Esta información suele completarse con datos referidos a la marca, modelo y versión de la mercadería.

A los fines de la experimentación se ha fraccionado la base de datos en tres segmentos de aproximadamente 10.000, 20.000 y 50.000 transacciones, respectivamente.

La tabla 5.1 detalla cada uno de los tres segmentos que se utilizaron en la experimentación. La fecha inicial es 01/01/98. Hasta el 28/05/98 se contaron 10001 transacciones que se construyeron de 84.746 filas de la

tabla que hasta ese momento incluía 3.554 ítems diferentes. Los demás segmentos se van integrando en forma acumulativa.

Tabla 5.1: conformación de las bases de datos

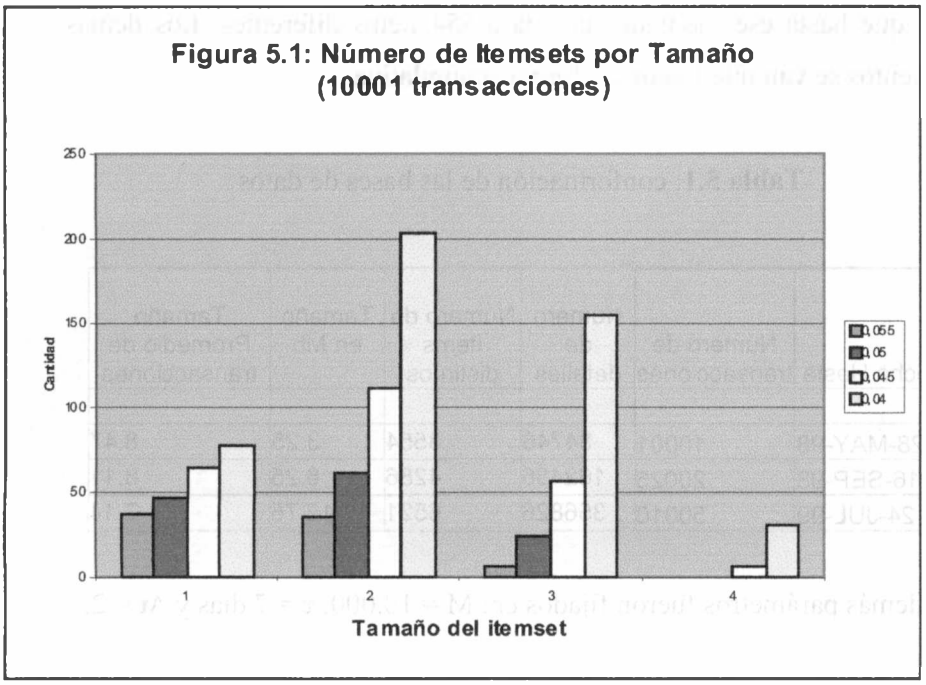
Fecha Hasta	Número de transacciones	Número de detalles	Número de ítems distintos	Tamaño en Mb	Tamaño Promedio de transacciones
28-MAY-98	10001	84746	3554	3.25	8.47
16-SEP-98	20025	162466	4286	6.25	8.11
24-JUL-99	50010	356826	6521	12.75	7.14

Los demás parámetros fueron fijados en: M = 10.000, τ = 7 días y Δt = 2.

5.1.1 Primer Experimento: Sobre Características de la Base de Datos Real

Esta primer serie de pruebas está destinada a explorar las características del conjunto de transacciones. Podremos observar el número de itemsets descubiertos, para distintos valores de soporte mínimo y distintos tamaños de la base de datos.

Los tres gráficos siguientes, figuras 5.1, 5.2 y 5.3, muestran el resultado de procesar 10001, 20025 y 50010 transacciones, respectivamente, con diversos valores de soporte. Se ha graficado el número de itemsets descubiertos versus el tamaño de dichos itemsets, para valores de soporte mínimo 0.04, 0.045, 0.05 y 0.055. En los dos primeros gráficos se puede observar que, como era de esperar, la mayoría de los itemsets se verifica para el tamaño 2 y esa tendencia se agudiza para los valores menores de soporte.



En esta primer parte de la experimentación, hemos contabilizado los itemsets que son frecuentes en todo el período, es decir, del 01/01/98 al 28/05/98. Estos itemsets corresponden, aproximadamente, a los descubiertos por un algoritmo no temporal tal como Apriori estándar. Los resultados de esa contabilización se muestran en la tabla 5.2. Se observa que el número de itemsets descubiertos tomando en cuenta el tiempo es de un orden de magnitud mayor que en el caso no temporal.

Tabla 5.2: Itemsets descubiertos a priori vs. TDIC

Soporte Mínimo	Itemsets frecuentes en todo el período	Itemsets descubiertos por TDIC
4.0%	38	466
4.5%	27	240
5.0%	20	131
5.5%	12	79

Figura 5.2: Número de Itemsets por tamaño (20025 transacciones)

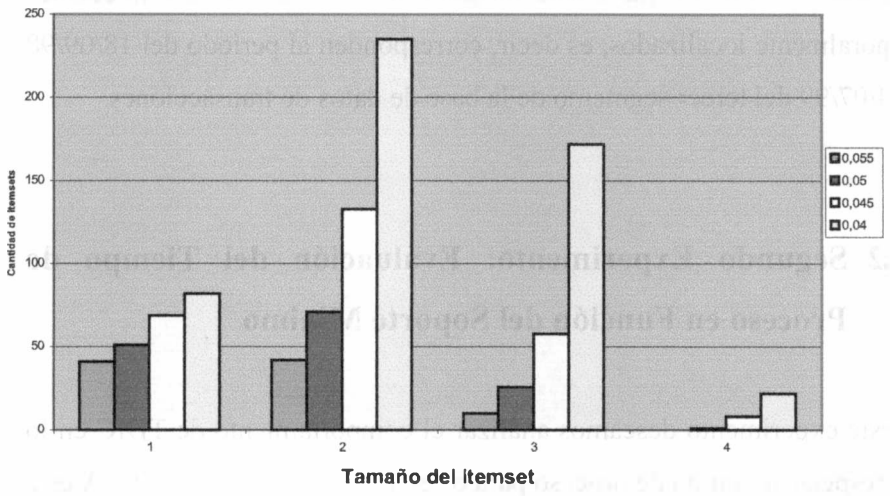
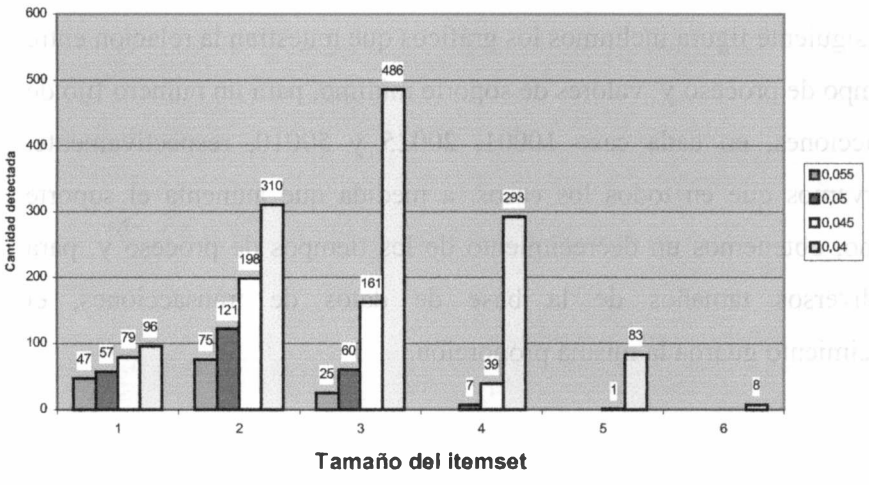


Figura 5.3: Número de itemsets por tamaño (50.010 transacciones)



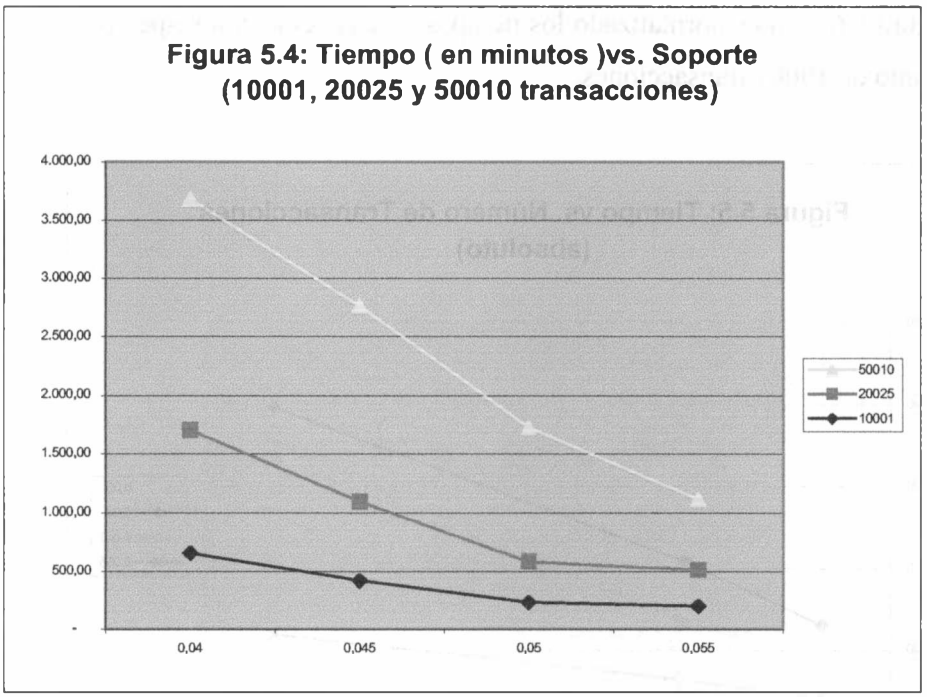
En el gráfico de la figura 5.3 se observa que, para un valor de soporte relativamente bajo de 4%, los itemsets generados más numerosos corresponden al tamaño 3 y que incluso los de tamaño 4 tienen un número comparable a los de tamaño 2. Para valores de soporte mínimo mayor se

verifica el comportamiento general, es decir, la mayoría de los itemsets son de tamaño 2. Debido a que esto no se observó en los gráficos anteriores, se infiere que existe un gran número de itemsets frecuentes temporalmente localizados, es decir, corresponden al período del 18/09/98 al 24/07/99 del tercer segmento de la base de datos de transacciones.

5.1.2 Segundo Experimento: Evaluación del Tiempo de Proceso en Función del Soporte Mínimo

En este experimento deseamos analizar el comportamiento de TDIC en lo que respecta a tiempo de proceso para diferentes valores de soporte. A este efecto hemos realizado cuatro corridas de nuestro algoritmo, correspondiente a cada valor de soporte considerado, para cada tamaño de la base de transacciones.

En la siguiente figura incluimos los gráficos que muestran la relación entre el tiempo de proceso y valores de soporte mínimo, para un número fijo de transacciones, en cada caso 10001, 20025 y 50010, respectivamente. Observamos que en todos los casos, a medida que aumenta el soporte mínimo, obtenemos un decrecimiento de los tiempos de proceso y, para los diversos tamaños de la base de datos de transacciones, el decrecimiento guarda la misma proporción.



5.1.3 Tercer Experimento: Evaluación de la Escalabilidad de TDIC

En este experimento deseamos analizar el desempeño de escalamiento de TDIC.

Nuevamente, se han considerado cuatro corridas de TDIC, una para cada valor de soporte mínimo, por cada segmento de la base de transacciones.

En la figura 5.5 mostramos el resultado de la experimentación en la que examinamos el desempeño de escalamiento de TDIC. Se han obtenido resultados de tiempos de ejecución para los tres conjuntos de datos reales, de tamaño 10.001, 20.025 y 50.010, respectivamente. Podemos observar el desempeño de TDIC a medida que los valores de |**d**| se incrementan. En

la figura 5.6 hemos normalizado los tiempos de ejecución con respecto al conjunto de 10001 transacciones.

Figura 5.5: Tiempo vs. Número de Transacciones (absoluto)

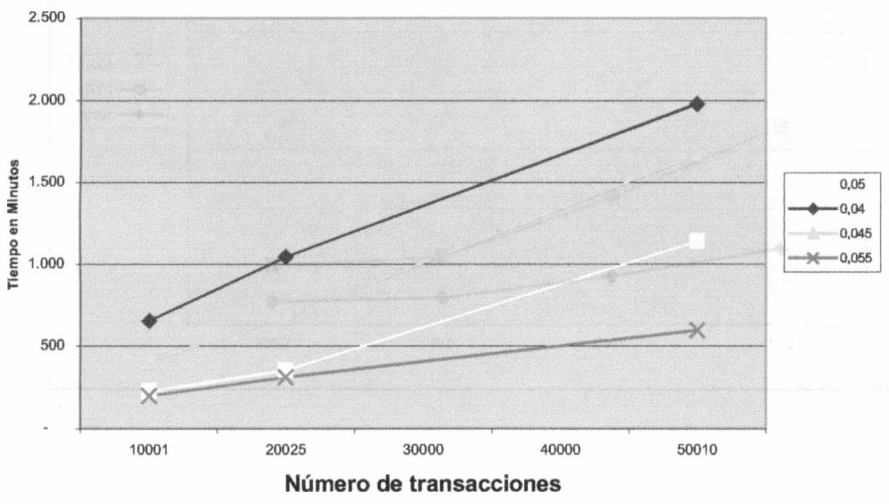
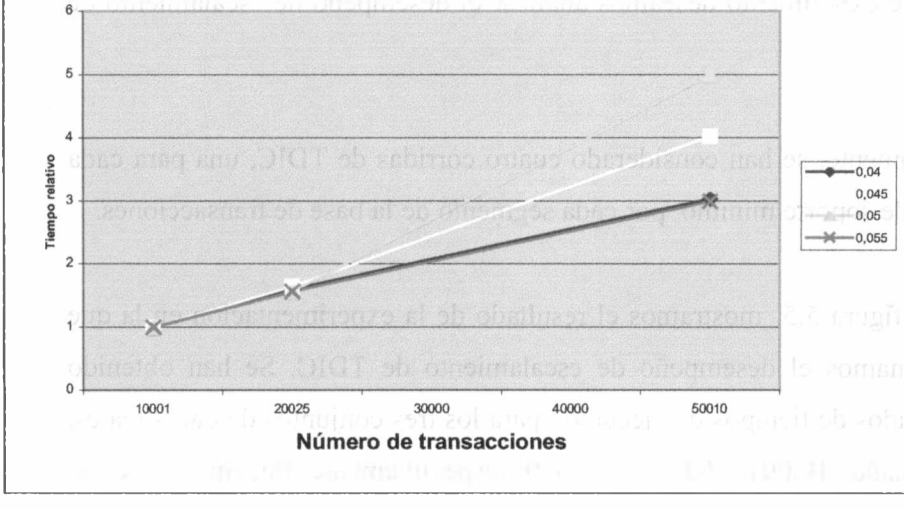


Figura 5.6: Tiempo vs. Número deTransacciones (Relativo)



Observamos que el tiempo se incrementa linealmente con respecto al tamaño de la base de datos de transacciones, lo que muestra la escalabilidad de TDIC.

5.2 Experimentación sobre una base de datos sintética

El objetivo de esta experimentación es analizar el comportamiento de TDIC sobre una base de datos de transacciones sintéticas de amplia utilización en la experimentación en el descubrimiento de reglas de asociación. Se desea analizar el efecto de M, el tamaño de cada porción en que se particiona la base de datos, para diversos valores de soporte mínimo y la escalabilidad del algoritmo.

Para esta experimentación se ha utilizado un computador con un procesador Pentium III de 800 MHZ y 1,5 GB de memoria, disco rígido IDE de 30 GB de 7200 rpm. El sistema operativo empleado fue Windows NT 4.0 y JAVA el lenguaje de desarrollo.

Con la finalidad de obtener resultados experimentales confiables, el método empleado para generar las transacciones sintéticas es similar al de [AS94]. A efectos de obtener una representación del tiempo razonable, simulamos los tiempos de arribo a una cola con un generador de desvíos pseudo-aleatorios con distribución exponencial negativa $f(t)= \lambda.e^{-\lambda t}$, donde la media es $1/\lambda$, mientras $F(t) = e^{-\lambda t}$; por lo que los tiempos entre arribos resultan $t = - 1/ \lambda . \log_e F(t)$, siendo F(t) un valor pseudo-aleatorio, entre 0 y 1, con distribución uniforme. Para la generación de los desvíos t utilizamos $\lambda = 35$, es decir, una tasa de 35 arribos por minuto.

En la tabla 5.3 resumimos el significado de varios parámetros utilizados en los experimentos, de acuerdo con el modelo de generación de datos

desarrollado en [AS94] y que, prácticamente, constituye el estándar para la experimentación en reglas de asociación..

Tabla 5.3: Parámetros para la generación de datos

Notación	Significado
$\|D\|$	Número de transacciones de la base de datos
|T|	Tamaño promedio de las transacciones
|I|	Tamaño promedio de los itemsets maximales potencialmente frecuentes.
|L|	Número de itemsets frecuentes maximales potencialmente frecuentes.
N	Número de ítems
M	Número de transacciones en cada porción de la base de datos de transacciones

Se generaron diversas bases de transacciones, las que presentan las diferentes características en función de:

Tamaño Promedio de Transacción
Tamaño Promedio de Itemset Maximal Potencialmente Frecuente
Cantidad de Transacciones

Siguiendo el modelo de Agrawal y Srikant, se generaron seis bases de datos de transacciones sintéticas, variando los valores de cada una de las características enunciadas.

La tabla 5.4 resume los parámetros utilizados para cada versión de base de datos de transacciones utilizada.

Las bases de transacciones fueron generadas utilizando un conjunto de 1000 ítems y 2000 itemsets potencialmente frecuentes. Además, se utilizó *mean correlation* 0,5 y *mean corruption* 0,5.

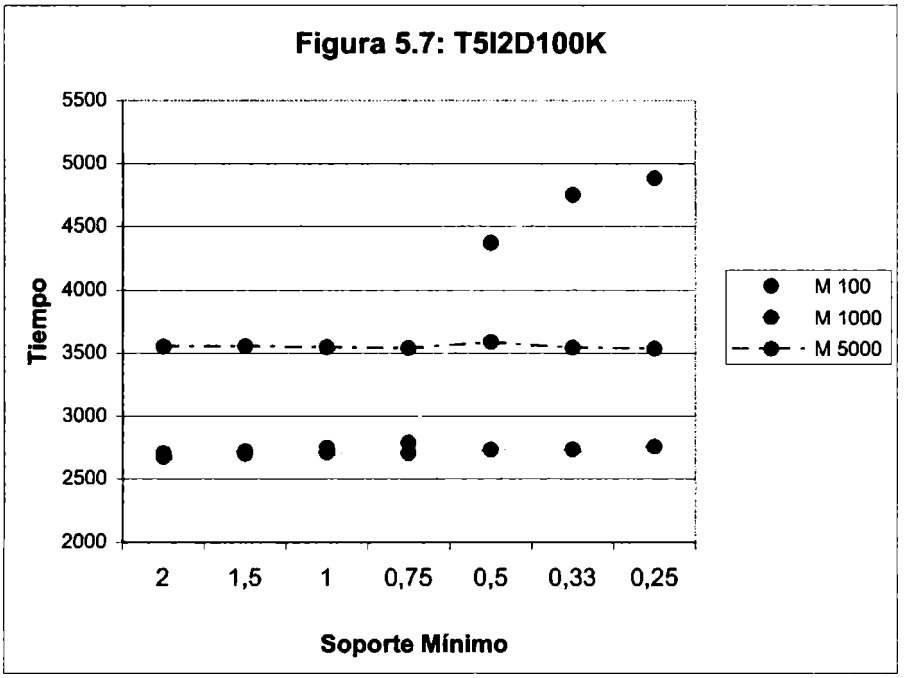
Tabla 5.4: Valores de parámetros

Nombre	|T|	|I|	|D|	Tamaño en Megabytes
T5.I2.D100K	5	2	100K	4.28
T10.I2.D100K	10	2	100K	7.53
T10.I4.D100K	10	4	100K	7.57
T20.I2.D100K	20	2	100K	11.74
T20.I4.D100K	20	4	100K	12.73
T20.I6.D100K	20	6	100K	13.29

En las siguientes subsecciones presentamos los resultados de una serie de experimentos realizados sobre las bases de datos de transacciones sintéticas, generadas de acuerdo con los valores de parámetros de la tabla 5.4. El segundo experimento prueba la escalabilidad de nuestro algoritmo, para lo cual se han utilizado bases de transacciones de diverso tamaño y varios valores de soporte mínimo. Todos los tiempos son expresados en segundos.

5.2.1 Primer Experimento: Evaluación del Tiempo de Proceso para Diversos Conjuntos de Datos, Soporte Mínimo y M

Se desea evaluar el tiempo de proceso en función de M. El primer experimento consistió en ejecutar TDIC para diversos juegos de conjuntos de datos, soporte mínimo y tamaño de M.



En la figura 5.7 observamos que el tiempo de ejecución para el M más pequeño (M=100), crece y supera a los de M superiores a partir de un determinado valor de soporte mínimo. En este caso 0.75%. Este comportamiento responde a que con valores altos de soporte, siempre encuentra pocos itemsets frecuentes, por lo que la generación de candidatos, aunque sea realizada frecuentemente, insume poco tiempo. Cuando los valores de soporte mínimo son más pequeños, necesariamente habrá más itemsets con soporte mínimo, por lo que la realización frecuente del paso de generación de candidatos incidirá significativamente en el tiempo de proceso.

Observamos en la figura 5.8 que los mejores tiempos se obtienen para el M más pequeño, es decir M = 100. Sin embargo, la curva crece para valores más pequeños de soporte mínimo. Observamos que para valores mayores de M, los tiempos se mantienen aproximadamente constantes, independientemente del soporte mínimo.

Figura 5.8: T10I2D100K

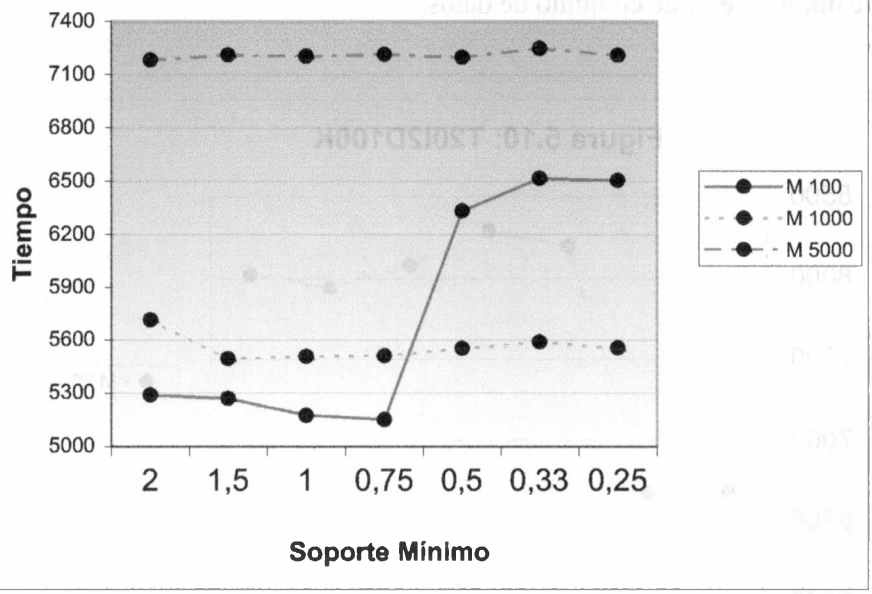
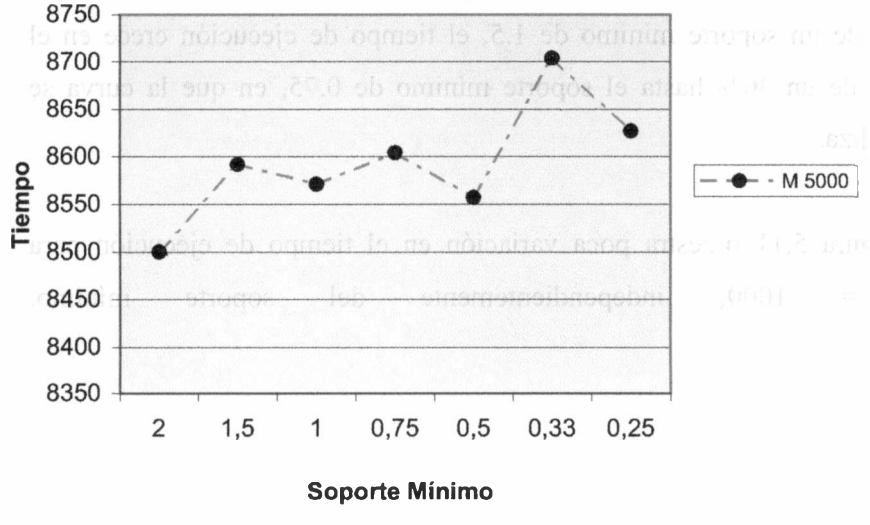
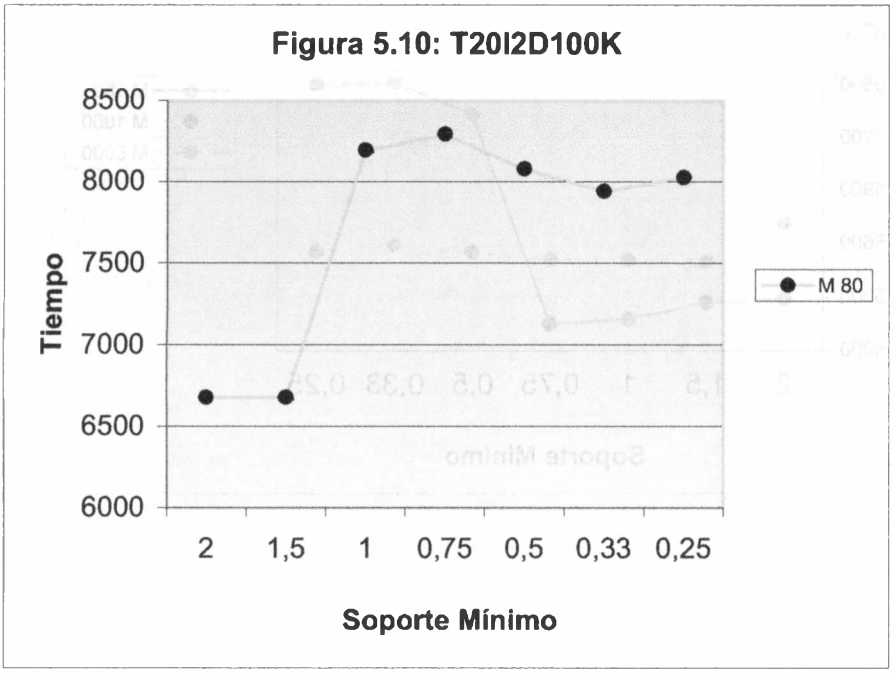


Figura 5.9: T10I4D100K



Del gráfico de la figura 5.9 concluimos que, si bien fluctúan, los tiempos para valores altos de M crecen suavemente a medida que se reduce el soporte mínimo, en este conjunto de datos.



En este caso, la figura 5.10 muestra que para un valor pequeño de M, a partir de un soporte mínimo de 1.5, el tiempo de ejecución crece en el orden de un 30% hasta el soporte mínimo de 0.75, en que la curva se estabiliza.

La figura 5.11 muestra poca variación en el tiempo de ejecución para M = 1000, independientemente del soporte mínimo.

Figura 5.11: T20I4D100K

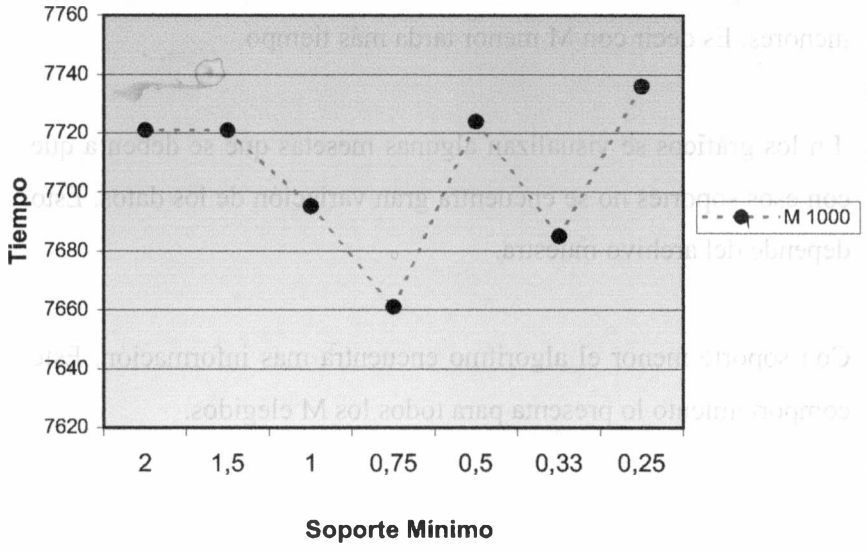
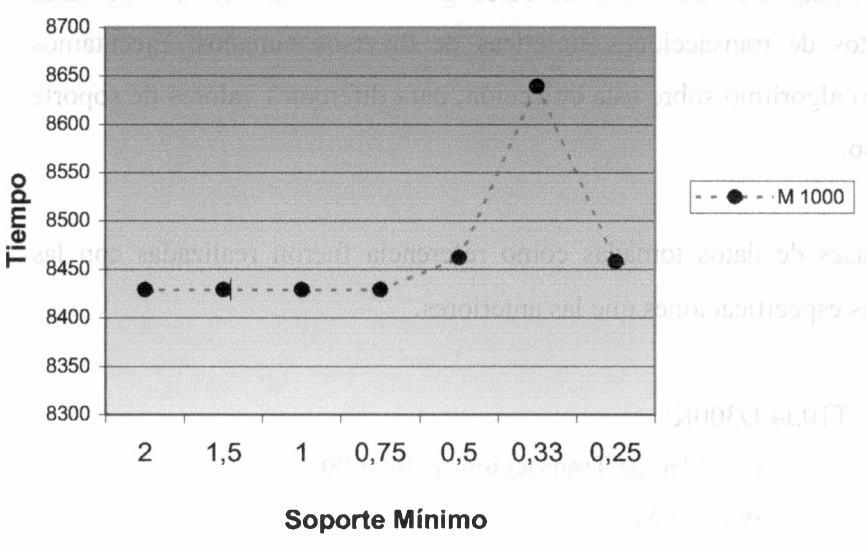


Figura 5.12: T20I6D100K



En este último gráfico (figura 5.12), se confirma la tendencia de la figura anterior.

Resumiendo, las conclusiones a las que arribamos son:

- Los tiempos de ejecución aumentan a medida que se eligen M's menores. Es decir con M menor tarda más tiempo.

- En los gráficos se visualizan algunas mesetas que se deben a que con esos soportes no se encuentra gran variación de los datos. Esto depende del archivo muestra.

- Con soporte menor el algoritmo encuentra mas información. Este comportamiento lo presenta para todos los M elegidos.

5.2.2 Segundo Experimento: Evaluación de la Escalabilidad de TDIC para Diversos Conjuntos de Datos y Soportes

Para evaluar la escalabilidad de TDIC generamos una colección de bases de datos de transacciones sintéticas de diversos tamaños. Ejecutamos nuestro algoritmo sobre esta colección, para diferentes valores de soporte mínimo.

Las bases de datos tomadas como referencia fueron realizadas con las mismas especificaciones que las anteriores.

- T10.I4.D300K:
  Cantidad de Transacciones: 300.000
  Peso:22 Mb.

- T10.I4.D500K:
  Cantidad de Transacciones: 500.000

Peso:37.5 Mb.

- T10.I4.D700K:

  Cantidad de Transacciones: 700.000

  Peso:57 Mb.

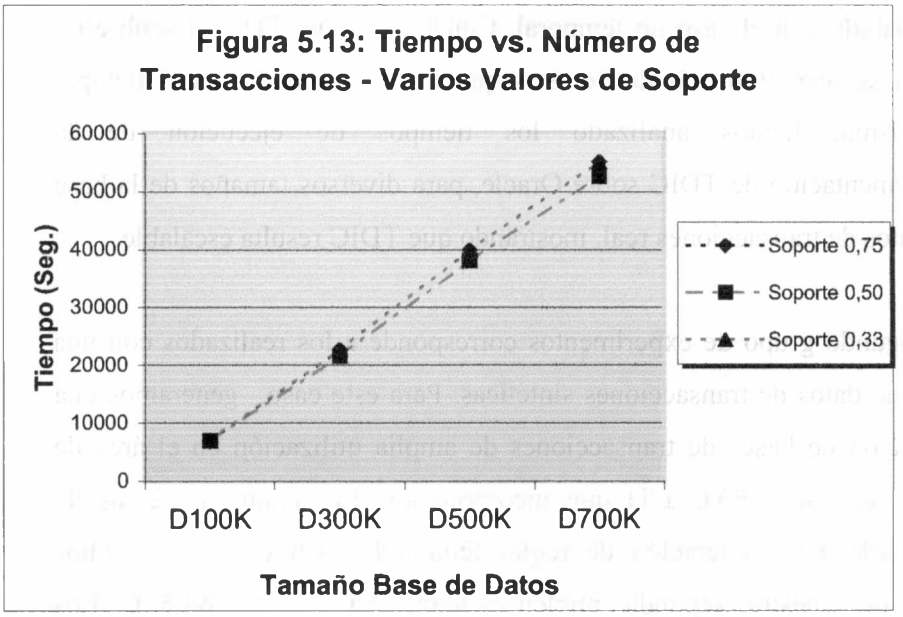


Figura 5.13: Tiempo vs. Número de Transacciones - Varios Valores de Soporte

Se puede observar que el tiempo de ejecución se conserva proporcional a la cantidad de transacciones, con lo que concluimos que, también en el caso de datos sintéticos, TDIC resulta escalable.

5.3 Conclusión

El objetivo de este capítulo ha sido analizar el comportamiento de nuestro algoritmo TDIC, en varios aspectos. El primero fue verificar que el algoritmo captura el aspecto temporal, en el sentido de descubrir itemsets localizados en el tiempo, y descubre itemsets que los algoritmos no temporales no descubren. El segundo fue probar que TDIC escala

linealmente o menos. El tercero, observar el comportamiento con diversos valores de M o porciones de la base de datos de transacciones.

Para esto hemos llevado a cabo una serie de experimentos. Los primeros fueron realizados sobre una base de datos real, donde hemos analizado la cantidad de itemsets generados para distintos valores de soporte mínimo y comparado con el caso no temporal. Concluimos que TDIC descubre un orden de magnitud más de itemsets que si no se considerara el tiempo. Asimismo, hemos analizado los tiempos de ejecución de la implementación de TDIC sobre Oracle, para diversos tamaños de la base de datos de transacciones real, mostrando que TDIC resulta escalable.

El segundo grupo de experimentos corresponde a los realizados con una base de datos de transacciones sintéticas. Para este caso, generamos una colección de bases de transacciones de amplia utilización en el área de reglas de asociación, a la que incorporamos timestamps a efectos de adecuarla a la generación de reglas temporales. En este caso, hemos utilizado nuestra segunda implementación, basada en WEKA. Los experimentos realizados muestran la incidencia de M, el tamaño de las porciones de la base de transacciones al final de las cuales se realiza el recuento de itemsets frecuentes, observándose que, para pequeños valores de M el tiempo de ejecución aumenta sensiblemente con la reducción del soporte mínimo. La experimentación con cuatro bases de datos sintéticas de diverso tamaño y varios valores de soporte mínimo, prueba que también en este caso TDIC resulta escalable.

Capítulo 6

Comparación con otros Enfoques de Reglas de Asociación Temporales

Nuestro modelo de reglas de asociación, basado en los lifespans de los itemsets, comparte con todos los demás modelos temporales la idea básica de representar las reglas dentro de un marco temporal.

Como las reglas de asociación temporales son una extensión de las reglas no temporales, se han considerado, para su descubrimiento, las mismas restricciones o parámetros del caso no temporal, es decir, el soporte y la confianza.

En nuestro caso, hemos advertido sobre lo erróneo de considerar, en muchísimas situaciones, que los ítems a evaluar puedan aparecer en cualquier momento del período que abarca la base de datos de transacciones. Esto se ha traducido, en el modelo no temporal, en la forma de cálculo del soporte. El valor de soporte de un itemset se asimila a la probabilidad que al extraer al azar una transacción de la base de transacciones, esa transacción contenga al itemset. Sin embargo, como ya se expresó, intuimos que ciertos ítems, por ejemplo nuevos productos, tienen *probabilidad cero* de aparecer en ciertas regiones de la base de transacciones asociadas a tiempos previos a la existencia de esos ítems.

Una característica común a todos los enfoques de reglas temporales es la forma de cálculo del soporte: el valor de soporte resulta la probabilidad de encontrar el ítem, o itemset, en una transacción extraída al azar pero cuyo timestamp corresponda a un determinado período, en el que se toma en consideración dicho itemset. Nuestro enfoque difiere de los expuestos en

que el período a considerar es el período de vida o lifespan del ítem o itemset, originalmente introducido en [CT85].

En el concepto amplio de lifespan [CC93], también llamado elemento temporal en [GV85] y [Ga88], consideramos que un itemset puede exhibirse, en una base de datos de transacciones, en forma no continua dentro de su período de vida. Esto nos conduce a la posibilidad de tomar en cuenta subintervalos dentro de un lifespan: períodos en que un itemset se manifiesta o aparece en transacciones, dentro de su lifespan.

6.1 Comparación con las Propuestas Basadas en el Enfoque de Calendarios

En el enfoque de calendarios se trata de descubrir las reglas que se cumplen, de acuerdo con los parámetros de soporte y confianza, en determinados momentos. Dichos momentos son especificados por el usuario por medio de un álgebra de calendarios (Reglas de Calendario[RMS98]), en la forma de expresiones de calendario (Reglas Temporales [CPH98]) o esquemas de calendario (Reglas Temporales basadas en Calendarios [LNWJ01]). En todos los casos, el usuario desea saber qué reglas se cumplen en los intervalos especificados. Todos los estudios mencionados han elaborado formalismos orientados a resolver *cómo especificar patrones de tiempo* en los cuales buscar reglas que se cumplan.

Nuestro enfoque difiere fundamentalmente en que deseamos descubrir *cuándo* se cumple cada regla, es decir, *cuáles son los patrones de tiempo* en que se cumplen las reglas.

En el ejemplo que los autores proponen en [LNWJ01] sobre *huevos* y *café*, la regla de asociación sería *huevos* ⇒ *café* con un soporte de 40% entre las siete y las once de la mañana, mientras que fuera de ese horario el soporte puede reducirse a 0.005%. Luego, para encontrar esta regla en el horario de la mañana, el usuario debe tener un buen conocimiento de sus datos. Este conocimiento debería permitirle establecer en su esquema la situación "todos los días hábiles de 7 a 11 horas". Pero, ¿cómo sabe el usuario que fuera de ese horario el soporte de la regla es 0.005%?. Para ello tendría que definir un esquema de calendario complementario al original y realizar el proceso de búsqueda de itemsets frecuentes, en todos los intervalos cubiertos por el esquema.

El siguiente ejemplo de [LNWJ01] también se refiere a una base de datos de transacciones en un supermercado. En él se analiza la venta de *pavo* y *pastel de calabaza* y expresa puntualmente que si miramos las transacciones en la semana previa al Día de Acción de Gracias, podemos descubrir que la mayoría de las transacciones contienen *pavo* y *pastel de calabaza*. Es decir, que la regla de asociación

Pavo ⇒ *Pastel de Calabaza*, tiene alto soporte y alta confianza en las transacciones realizadas en la semana previa al día de Acción de Gracias. El problema que se plantea, es que el usuario debe tener un buen conocimiento previo del dominio, de lo contrario no estaría en condiciones de fijar el esquema que le permitiría descubrir este tipo de regla. Por otra parte, si el usuario definió realizar el mining en esa semana, ¿cómo sabe si la regla no se cumple en alguna otra semana del año?.

En resumen, el enfoque basado en calendarios, en todas sus variantes, requiere una gran participación del usuario. En cierto sentido, se asemeja al descubrimiento dirigido ya que el usuario conduce el proceso de mining fijando los intervalos en los cuales se desea descubrir reglas. Además, la información que se brinda puede ser confusa: es posible saber qué reglas

se cumplen en los intervalos prescriptos, luego el usuario puede interpretar que *sólo* se cumplen en esos intervalos, lo cual no está asegurado. Por ejemplo, una regla que, de acuerdo al patrón de búsqueda fijado, se cumple los primeros días del mes como *asado* $\Rightarrow$ *vino*, induciría al usuario a promover la venta de vinos rebajando el precio del asado durante la primeros días del mes. Sin embargo, es posible que esa regla también se verifique los primeros días de cada quincena. Como dicha información no fue proporcionada, el usuario pierde la oportunidad de promover los vinos en la segunda quincena. En otro tipo de ejemplos, como las reglas temporales entre síntomas y tratamientos, la falta de una información completa puede inducir a diagnósticos y/o tratamientos erróneos.

En nuestro modelo de Reglas de Asociación Temporales Generalizadas, nuestros algoritmos permiten hallar todas las reglas con todos los intervalos en que resultan válidas, *sin ningún conocimiento previo de posibles patrones temporales*.

El concepto de historia de un itemset permite analizar temporalmente las variaciones en soporte de un conjunto de ítems. La idea de su representación por medio de una serie temporal nos habilita a realizar diversos tipos de análisis. Entre éstos se cuentan los siguientes: análisis de tendencia, estacionalidad y ciclos para predicción de valores futuros; análisis de similitud y correlación entre historias de itemsets para detectar itemsets de comportamiento similar o interdependiente.

En nuestra propuesta, la historia de los itemsets producida como resultado del proceso de mining, se conserva para análisis posteriores como los mencionados en el párrafo anterior. Pero este conjunto de historias constituye una representación, en cierto sentido, de la base de datos de transacciones. Luego, sobre esta nueva representación, la posibilidad de súper imponer calendarios o esquema de calendario, es inmediata. De esta

forma podemos obtener reglas basadas en calendarios con mínimo costo computacional.

6.2 Comparación con el Modelo de Base de Datos de Publicaciones

Mucho de lo expresado en la comparación con el modelo basado en calendarios resulta válido también para este modelo.

Sin embargo, este modelo no usa expresiones basadas en calendarios para el mining de la base de datos de transacciones y por sus características, resulta el más próximo a nuestro enfoque.

En [LCL03] los autores expresan que los diversos trabajos, tales como Ale y Rossi [AR00], Bettini et al.[BWJ98], Chen et al.[CPH98], Li et al.LNWJ01] y otros, propuestos para explorar el problema del descubrimiento de reglas de asociación temporal, fallan en el problema que proponen. Puntualmente expresan que esos trabajos no consideran el período de exhibición individual de cada ítem, en las transacciones. De esa manera, no son aplicables para resolver los problemas de mining en una base de datos de publicaciones.

Recordemos que el modelo planteado por Lee et al., también en [LLC01] de Noviembre de 2001, fundamenta el algoritmo Progressive Partition Miner. El algoritmo considera que la base de datos de transacciones **d** se encuentra particionada. Ellos llaman $db^{t,n}$ a la parte de **d** formada por una región contigua desde la partición P_t a la partición P_n. Un ítem $x^{x.start,n}$ es designado como un ítem temporal de x, lo que significa que $P_{x.start}$ es la partición inicial de x y n es el número de partición de la última partición de la base de datos, a un determinado momento.

En particular, definen un itemset temporal maximal $X^{t,n}$, en una base de datos parcial $db^{t,n}$, tal que t es el último número de partición de comienzo de todos los ítems que integran X en **d**, y n es el número de partición de la última partición en $db^{t,n}$. Vemos que esta definición es muy similar a la definición 4 del Capítulo 3 referida al lifespan de un k-itemset, aparecida en nuestra publicación [AR00] de Marzo de 2000.

Lee et al.expresan el período de exhibición de un itemset en términos del *Maximal Common* exhibition *Period*, o MCP de los ítems que aparecen en el itemset. Este concepto es similar al de amplitud de lifespan de [AR00].

Nuevamente en [LCL03], los mismos autores introducen en su modelo un nuevo parámetro. Éste es un umbral, denotado por *min_leng*, para el período de exhibición de un ítem, ya que el usuario podría no estar interesado en ítems o itemsets con período de exhibición muy breve. Observamos que *min_leng* es idéntico a nuestro umbral denominado *soporte temporal*, denotado por τ.

Finalmente, en su comparación con trabajos relacionados en [LCL03], los autores argumentan que, en su trabajo, se asume que cada ítem tiene el mismo tiempo de corte del período de exhibición del ítem, es decir, el n de (t, n) y, en consecuencia, es diferente de la definición previa de "lifespan", originalmente publicada en [AR00], la que puede tener diferentes momentos de finalización de los períodos de exhibición de los ítems. Aclaran también que la formulación del problema con el mismo período de finalización, les permite derivar algoritmos muy eficientes y efectivos para el mining de reglas de asociación temporales.

El último párrafo puede ser una justificación válida, en términos de eficiencia de los algoritmos. Sin embargo, en términos de modelo

temporal, nuestra versión, de [AR00] y su extensión [AR02], es más precisa, ya que permite modelizar también la situación en que una publicación deja de exhibirse, por ejemplo, por encontrarse agotada.

Esta breve comparación nos ha permitido mostrar que nuestras reglas de asociación temporales generalizadas, son efectivamente más generales que las reglas propuestas en los otros enfoques analizados. Además, a pesar de su generalidad, permiten brindar información más precisa de los fenómenos temporales subyacentes. Un aspecto importante a considerar es que nuestro enfoque requiere el mínimo conocimiento previo de todos los enfoques. El usuario sólo debe proporcionar los umbrales de soporte σ, confianza θ y soporte temporal τ, y, eventualmente, el intervalo unitario Δt para la construcción de los histogramas.

Para finalizar este Capítulo, deseamos puntualizar que, como se puede inferir de lo expresado en el Capítulo 4 y en esta comparación, es posible obtener reglas basadas en calendarios con suma facilidad, en base a la información proporcionada por nuestro modelo. Por otra parte, otros autores, tales como Zimbrao et al. en [ZMTA02], consideran que el enfoque de calendarios y nuestro modelo básico, de [AR00], son complementarios.

En [ZMTA02] los autores consideran que existen, básicamente, dos enfoques al descubrimiento de reglas de asociación temporales. El primero trata de descubrir reglas de asociación que ocurren cíclicamente, o de acuerdo a algún patrón temporal especificado por medio de un calendario. Los trabajos en [ORS98], [RMS98], [CPH98] y fundamentalmente [LNWJ01] responden al primer enfoque. El otro enfoque trata de hallar reglas de asociación durante el tiempo de vida de los itemsets. El trabajo en [AR00] corresponde a este segundo enfoque. Como conclusión de este trabajo, los autores proponen un algoritmo que permite el descubrimiento

de reglas estacionales, utilizando el concepto de lifespan de los itemsets, que no habrían sido descubiertas por medio del enfoque de calendarios solamente.

6.3 Conclusión

En esta comparación, hemos mostrado las diferencias entre nuestro enfoque, basado en el lifespan de los itemsets, y otros enfoques al descubrimiento de reglas de asociación temporales. A este efecto, categorizamos los trabajos en dos grandes grupos, los que responden al enfoque de calendarios, donde el usuario debe tener un conocimiento previo de los patrones de calendario a utilizar, y otros, como el de [LCL03], para el que mostramos similitudes con nuestro trabajo y del que sería, eventualmente, un caso particular.

Como resultado de la comparación mostramos que nuestro enfoque es el que menos conocimiento previo requiere, las reglas obtenidas son las más generales y es posible aún combinar los dos enfoques, el del lifespan de los itemsets y el de calendarios.

Capítulo 7

Conclusiones y Líneas de Trabajo Futuro

El descubrimiento de reglas de asociación, originado en el análisis de canasta de mercado, tiene por finalidad, entre otras, explicar el comportamiento y el interés de clientes de un supermercado. En otro tipo de aplicaciones, permite el análisis de asociaciones de manera clara y precisa.

Sin embargo, el modelo inicial de Agrawal et al.[AIS93] presenta una serie de dificultades. Nosotros consideramos que parte de esas dificultades se solucionan incorporando la variable tiempo al modelo inicial, con lo que se obtienen representaciones más precisas de la realidad. Para ello incluimos la dimensión temporal, la que se encuentra explícitamente representada por los timestamps asociados a las transacciones de la base de datos.

7.1 Resumen y Contribuciones

Esta tesis está enfocada en la investigación de un modelo temporal para las reglas de asociación, que permite representar más fielmente las asociaciones y descubre reglas ignoradas por el modelo estándar, no temporal. Las principales contribuciones están enunciadas en la Sección 1.3 del Capítulo 1 Introducción. Los Capítulos 3 y 4 presentan el Modelo Temporal Básico y las Reglas de Asociación temporales Generalizadas, respectivamente. En particular, nuestro trabajo se resume en los siguientes elementos:

- Definición de un modelo de regla de asociación basado en el concepto de lifespan de los itemsets. Este modelo no requiere de ningún conocimiento previo, tal como los necesarios para la definición de intervalos de búsqueda, ciclos o calendarios. Entre los nociones introducidas para la elaboración de este modelo se cuentan la de *obsolescencia*, según la cual ítems antiguos, es decir, aquellos con un fin de período de vida anterior a un parámetro t_o, carecen de interés y pueden ser eliminados de la consideración. Otra noción es la de *soporte temporal*, de acuerdo con la cual sólo resultan de interés los ítems o itemsets cuyo lifespan supera un mínimo establecido τ. Finalmente la noción de *subintervalo frecuente*, lo que permite considerar períodos dentro del lifespan de un itemset dentro del cual éste resulta frecuente y, por lo tanto, de interés para el usuario.

- Definición de la noción de historia de un itemset. Este concepto permite representar el comportamiento temporal de cada itemset, en el sentido de variaciones de frecuencia en el tiempo, dentro de su lifespan. Asimismo, permite representar la base de transacciones íntegra de una manera compacta y eficiente. Esto facilita realizar sobre esta representación diversos tipos de análisis, incluso descubrir con suma facilidad reglas basadas en el enfoque de calendarios. Se ha propuesto transformar la representación de la historia, considerada como una serie temporal, por medio de un análisis wavelet, habiendo argumentado que dicha transformada resulta totalmente adecuada al tipo de dato de nuestro problema.

- Un algoritmo eficiente para el descubrimiento de las reglas temporales, que aprovecha las características temporales

intrínsecas de los itemsets. Este algoritmo, llamado Temporal Dynamic Itemset Counting reduce la cantidad de lecturas de la base de datos de transacciones, en relación a Apriori, y ha sido posible incorporarle numerosas optimizaciones, basadas en las características temporales de los itemsets.

- Experimentación con datos reales provenientes de la Dirección de Aduana de la República Argentina, donde los ítems son posiciones arancelarias y las transacciones representan órdenes de importación. La particularidad de este tipo de datos es que son estrictamente temporales, es decir, muy pocas reglas de interés podrían descubrirse con el modelo estándar, no temporal. Hemos evaluado tiempos de ejecución y escalabilidad para diversos valores de soporte mínimo, número de reglas descubiertas, número de itemsets descubiertos para distintos valores de soporte, etc.

- Experimentación con una colección de bases de datos sintéticas, consideradas estándar en el área de reglas de asociación. Hemos adjudicado timestamps a las transacciones de dichas bases de transacciones, de acuerdo con nuestro propio modelo temporal. Evaluamos ciertos parámetros de nuestro algoritmo y probamos, también para este caso, que TDIC resulta escalable.

- Comparación del modelo basado en los lifespans de los itemsets con diversos trabajos y enfoques de descubrimiento de reglas de asociación temporales.

7.2 Líneas de Trabajo Futuro

Existen diversas líneas de trabajo futuro, de las cuales solo mencionaremos las que se encuentran en un cierto estado de elaboración.

7.2.1 Descubrimiento Incremental de Reglas de Asociación Temporales

Existen diversos trabajos sobre el descubrimiento incremental de reglas no temporales[CHW96, GGR01, TBAR97]). El enfoque general es evitar el reprocesamiento de la base de datos de transacciones y sólo procesar las nuevas transacciones que se incorporan a la base. Sin embargo, en todos los casos se ha requerido realizar al menos una lectura de la base de datos completa, con el costo que ello implica.

En nuestro caso, la idea es utilizar la historia de los itemsets, con su representación comprimida, tanto para eliminar el efecto de transacciones antiguas u obsoletas, y por ende irrelevantes, como para incorporar el efecto de nuevas transacciones. De esta manera, pueden surgir nuevas reglas de asociación, como así también algunas de las existentes pueden perder validez. La restricción, en este modelo de actualización, es evitar el acceso a la base de transacciones original.

7.2.2 Búsqueda de Patrones en las Reglas Temporales

En esta línea, estamos interesados en descubrir ítems con ciertas características, tales como los ítems sustitutos, en el Análisis de Canasta. Bajo ciertas condiciones es posible hallar pares de reglas que difieren en precisamente un ítem y son complementarias con respecto al lifespan de esos dos ítems. En este caso, decimos que un ítem es sustituto del otro.

Ésta, y otras características, resulta posible descubrirlas analizando el conjunto de las reglas de asociación temporales generadas desde una base de transacciones.

7.2.3 Análisis de Dependencias de Tendencias

El problema de las dependencias de tendencias ha sido presentado y analizado en [Wij99]. Se trata de analizar una base de datos histórica y descubrir *todas* las dependencias de tendencia. Para esto hallamos todos los itemsets frecuentes, en su lifespan, y luego analizamos sus tendencias temporales.

7.2.4 Reducción del Conjunto de Reglas de Asociación Temporales Descubiertas

Este trabajo es una extensión del realizado en [MA03] sobre reglas no temporales. En el citado trabajo se utiliza Defeasible Logic, una variante de lógica no monótona, para representar las reglas de asociación. La finalidad es reducir el número de reglas descubiertas por medio de esquemas de reglas de inferencia, llamadas "assumptions", que expresan intuiciones sobre propiedades observadas por las reglas de asociación, y "defeaters" que expresan excepciones a las "assumptions" descubiertas desde los datos.

La extensión consiste en aplicar estos mismos conceptos a reglas temporales, para lo cual analizamos diversos enfoques de "Description Logics" adecuados al caso temporal [AF00].

7.2.5 Integración de Reglas Temporales Basadas en Calendarios y Reglas Basadas en Lifespan

Dado que, en el proceso de descubrimiento de los itemsets frecuentes, hemos generado la historia de los mismos, es posible analizar si las reglas se ajustan a patrones temporales descriptos por medio de esquemas de calendario[LNWJ01].

En [ZMTA02] se muestra que resulta ventajoso combinar ambos enfoques. Pero nuestra propuesta difiere de la citada, en que aplicamos las técnicas basadas en calendarios para buscar regularidades sobre las reglas temporales ya descubiertas en base a nuestro modelo.

8. Referencias

[AFS93] Agrawal, R. – Faloutsos, C. – Swami, A.: Efficient similarity search in sequence databases. Proc. FODO Conf. 1993

[AIS93] Agrawal, R.-Imielinski, T.-Swami, A.: Mining Association Rules Between Sets of Items in Large Databases. Proc. ACM SIGMOD:207-216. 1993.

[AS94] Agrawal, R.-Srikant, R.: Fast Algorithms for Mining Association Rules. IBM Res. Rep. RJ9839, IBM Almaden. June 1994.

[AS94a] Agrawal, R.-Srikant, R.: Fast Algorithms for Mining Association Rules. Proc 20th BLVD. Conference, 1994.

[AS95] Agrawal, R.-Srikant, R.: Mining Sequential Patterns. Proc. IEEE Int'l Conference on Database Engineering: 3 - 14. 1995.

[AR00] Ale, J. – Rossi, G.: An Approach to Discovering Temporal Association Rules. . Proc. ACM 15th Symposium on Applied Computing, pages 294-300. March 2000.

[AR02] Ale, J.- Rossi, G.: The Itemset's Lifespan Approach to Discovering General Temporal Association Rules. Proc. ACM 2nd Workshop on Temporal Data Mining, held in KDD-2002, pages 1-10. July 2002.

[All85] Allen, J.: Maintaining Knowledge about Temporal Intervals. In *Readings in Knowledge Representation*, pages 509-521. Morgan Kaufmann Publishers. 1985.

[AF00] Artale, A. – Franconi, E.: A survey of temporal extensions of Description Logics. Annals of Mathematic and Artificial Intelligence. 2000.

[BWJ98] Bettini, C.-Wang, X.-Jajodia, S.-Lin, J.: Discovering Frequent Event Patterns With Multiple Granularities In Time Sequences. IEEE TOKDE Vol.10 N°2: 222-237. April 1998.

[BMU97] Brin, S.- Motwani, R. - Ullman, J. - Tsur, S.: Dynamic Itemset Counting and Implication Rules for Market Basket Data. Proc. ACM SIGMOD: 255-264. 1997.

[CSD98] Chakrabarti, S.-Sarawagi, S.-Dom, B.: Mining surprising patterns Using Temporal Description Length. Proc. 24th VLDB Conference 1998.

[CF99] Chan, K.-Fu, A.: Efficient Time Series Matching by Wavelets. Proc. IEEE 15th Intl.Conf. on Data Engineering, 1999.

[CSS94] Chandra, R.- Segev, A.-Stonebraker, M.: Implementing Calendars and Temporal Rules in Next Generation Databases. Proc. Int'l Conference in Data Engineering. Houston, February 1994.

[CPH98] Chen, X.-Petrounias, I.-Heathfield,H.: Discovering Temporal Association Rules in Temporal Databases. Proc. Int'l Workshop IADT'98. July 1998.

[CHY96] Chen, M.- Han, J.- Yu, P.: Data Mining: An Overview from Database Perspective. IEEE Transactions on Knowledge and Data Engineering, December 1996.

[CHW96] Cheung, D.-Han, J.-Ng, V.-Wong, C.: Maintenance of Discovered Association Rules in Large Databases: An Incremental Updating Technique. Proc. of 1996 Int'l Conf. On Data Engineering. Feb.1996.

[CC93] Clifford, J.- Croker, A.: The Historical Relatinal Data Model (HRDM) Revisited. In *Temporal Databases: Theory, Design and Implementation* (Tansel et al. eds.). Benjamín Cummings Publishing. 1993.

[CD96] Cukierman, D.- Delgrande, J.: A Language to Express Time Intervals and Repetition. Proc. Int'l Workshop on Temporal Representation and Reasoning, Florida, April 1996.

[CT85] Clifford, J.- Tansel, U.: On an Algebra for Historical Relational Databases: Two Views. In Proc. Of ACM SIGMOD, Austin, May 1985.

[Dau92] Daubechies, I.: Ten Lectures on Waveletes. SIAM, Philadelphia, 1992.

[FPS96] Fayyad, U.- Piatetsky-Shapiro, G.- Smyth, P.:From Data Mining to Knowledge Discovery. In *Advances in KD and Data Mining*. The MIT Press. 1996.

[FPS96a] Fayyad, U.- Piatetsky-Shapiro, G.- Smyth, P.: The KDD Process for Estracting Useful Knowledge from Volumes of Data. ACM Communications. Vol 39, N°11. Nov. 1996.

[Gad88] Gadia, S.: The rol of Temporal Elements in Temporal Databases. IEEE Data Engineering. December 1988.

[GV85] Gadia, S.- Vaishnav, J.: A Query Language for a Homogeneous Temporal Database. Proc. ACM Symposium on Principles of Database Systems, March 1985.

[GGR01] Ganti, V. Gehrke, J.- Ramakrishnan, R.: DEMON: Mining and Monitoring Evolving Data. IEEE Transactions on Knowledge and Data Engineering. Vol. 13 Number 1, January/February 2001.

[GK95] Goldin, D.-Kanellakis, P: On similarity queries for time series databases: Constraint specification and implementation. First Intl. Conf . on the Principles and Practice of Constraint Programming, pages 137-153. 1995.

[Gra95] Graps, A.L.; An Introduction to Wavelets, IEEE Computational Sciences and Engineering, Volume 2, Number 2, Summer 1995, pp 50-61.

[HF95] Han, J.- Fu, J `` Discovery of Multiple-Level Association Rules from Large Databases", Proc. of 1995 Int'l Conf. on Very Large Data Bases (VLDB'95), Zürich, Switzerland, September 1995, pp. 420-431

[HPY00] Han, J.-Pei, J. – Yin, Y.:Mining Frequent Patterns without Candidate Generation. Proc. ACM SIGMOD Int'l Conference. May 2000.

[HGN00] Hipp, J.- Güntzer, U.- Nakhaeizadeh. G.: Algorithms for Association Rule Mining – A General Survey and Comparison. ACM SIGKDD Exploration, July 2000.

[HKMT95] Holsheimer, M.- Kerstein, M.- Mannila, H.- Toivonen, H.: A Perspective on Databases and Data Mining. KDD'95. AAAI. August 1995.

[Kim96] Kimball, R.: *The Data Warehouse Toolkit.* JohnWiley & Sons. 1996.

[LMF86] Leban, B.- McDonald, D.-Forster, D.: A Representation for Collections of Temporal Intervals. Proc. AAAI 5th Int'l Conference on Artificial Intelligence. 1986.

[LCL03] Lee, C.H. – Chen, M.S. – Lin, C.R.: Progressive Partition Miner: An Efficient algorithm for Mining General temporal Association Rules. IEEE Trans. On Knowledge and Data Engineering, Vol. 15 Number 4 pages 1004-1017, July/August 2003.

[LLC01] Lee, C.H – Lin, C.R. – Chen, M.S.: On Mining General Temporal Association Rules in a Publication Database. Proc. of the IEEE Int'l Conference on Data Mining(ICDM-01). November 2001.

[LC00] Li, S.- Chou, S.: Multi-resolution spatio-temporal data mining for the study of air pollutant regionalization. In Proc. 33rd Hawai Int'l conference on System Science. 2000.

[LLZO02] Li, T.- Li, Q.- Zhu, S.- Ogihara, M.: A Survey on Wavelet Applications in Data Mining. ACM SIGKDD Exploration, pages 49-68, Vol. 4, Issue 2. December 2002.

[LNWJ01] Li, Y.- Ning, P.- Wang, X.- Jajodia, S.: Discovering Calendar-based Temporal Association Rules. Proc. 8th Int'l Symposium on Temporal Representation and Reasoning. 2001.

[Man96] Mannila, H.:Methods and Problems in Data Mining. Proc. 6th Int'l Conference in Database Theory. January 1997.

[MTV95] Mannila, H.- Toivonen, H.- Verkamo, A.: Discovering Frequent Episodes in Sequences. In First International Conference on Knowledge Discovery and Data Mining (KDD'95), 210 - 215, Montreal, Canada, August 1995.

[MT96] Mannila, H.- Toivonen, H.: Discovering generalized episodes using minimal occurrences. In Second International Conference on Knowledge Discovery and Data Mining (KDD'96), 146-151, Portland, Oregon, August 1996.

[MTV97] Mannila, H.- Toivonen, H.- Verkamo, A.: Discovery of frequent episodes in event sequences. Data Mining and Knowledge Discovery, 1(3): 259 - 289, November 1997. (Preliminary Report C-1997-15, University of Helsinki, Department of Computer Science, February 1997.)

[MVW98] Matias, Y.- Vitter, J.- Wang, M.: Wavelet-Based Histograms for Selectivity Estimation. Proc. ACM SIGMOD, pages 448-459, Seattle, June 1998.

[MA03] Minuto Espil, M.- Ale, J.: Embedding Association Rules in a Defeasible Description Logic Framework. XXIII

Conferencia Internacional de la Sociedad Chilena de Ciencia de la Computación. II Workshop de Bases de Datos. Chillán, Chile, Noviembre 2003.

[NS92] Niezette, M.- Stevenne, J.:An Efficient symbolic Representation of Periodic Time. Proc 1st. Int'l Conference on Information and Knowledge Management, Baltimore, 1992.

[ÖRS98] Özden, B.-Ramaswamy, S.-Silberschatz, A.: Cyclic Association Rules. ICDE 1998.

[PCY95] Park, J.S.-Chen, M.S.-Yu, P.S.: An Effective Hash Based Algorithm for Mining Association Rules. Proc ACM SIGMOD: 175-186. 1995.

[Pas00] Pasquier, N.: Mining Association Rules Using Formal Concept Analysis. Proc. ICCS'2000 conference, pp 259-264, Shaker-verlag, august 2000.

[PBTL98] Pasquier, N.- Bastide, Y.- Taouil, R.- Lakhal, L.: Pruning Closed Itemset Lattices for Association Rules. Proc. BDA Conference, pages 177-196. October 1998.

[PBTL99] Pasquier, N.- Bastide, Y.- Taouil, R.- Lakhal, L.: Eficiente Mining of Association Rules Using closed Itemset Lattices. Information Systems, Vol. 24 Number 1, pages 25-46, March 1999.

[PW00] Percival, D.- Walden, A.: *Wavelet Methods for Time Series Análisis*. Cambridge University Press. 2000.

[PM02] Popivanov, I.- Miller, R.: Similarity Search over Time-Series Data Using Wavelets. Proc. 18th Int'l Conference on Data Engineering. 2002.

[RM97] Rafiei, D.-Mendelzon, A.: Similarity-based queries for time series data. Proc. of the ACM-SIGMOD pages 13-23. 1997.

[RMS98] Ramaswami, S.-Mahajan, S- Silberschatz, A.: On the Discovery of Interesting Patterns in Associations Rules. Proc. 24th VLDB Conference. 1998.

[RS02] Roddick, J.- Spiliopoulou, M.: A Survey of Temporal Knowledge discovery Paradigms and Methods. IEEE Trans. On Knowledge Discovery and Data Engineering. Vol.14 Number 4, July/August 2002.

[RHS01] Roddick, J.- Hornsby, K.-Spiliopoulou, M.: YABTSSTDM R: Yet Another Bibliography of Temporal and Spatio-Temporal Data Mining Research. ACM 1st Workshop on Temporal Data Mining held in KDD-2001, San Francisco, 2001.

[SON95] Savasere, A.- Omiecinski, E.- Navathe, S.: An Eficcient algorithm for Mining Association Rules in Large Databases. Proc. 21th Int'l Conference on VLDB. 1995.

[SSU96] Silberschatz, A.- Stonebraker, M.- Ullman, J.: Database Research: Achievements and Opportunities into the 21st Century. ACM SIGMOD Record Volume 25 Number 1. March 1996.

[SA95]. Srikant, R.- Agrawal, R.: Mining Generalized Association Rules. Proc. 21st VLDB Conference. Zurich. 1995.

[SA96] Srikant, R.- Agrawal, R.: Mining Sequential Patterns: Generalization and Performance Improvements. 5th Int'l Conference on Extending Database Technology. Avignon. March 1996.

[SA96a]. Srikant, R.-Agrawal, R.: Mining Quantitative Association Rules In Large Relational Databases. Proc. ACM SIGMOD. 1996.

[TCG+93] Tansel, A. et al: *Temporal Databases: Theory, Design, and Implementation.* Benjaming/Cummings. 1993.

[TBAR97] Thomas, S.- Bodagala, S.- Alsabti, K.- Ranka, S.: An Efficient Algorithm for the Incremental Updation of Association Rules in Large Databases. Journal of Knowledge Discovery and Data Mining. 1997.

[Uth96] Uthurusamy, R.: From Data Mining to Knowledge Discovery: Currents Challenges and Future Directions. In *Advances in KD and Data Mining*. The MIT Press. 1996.

[Wij99] Wijsen, J.:Temporal Dependencies with Order Constraints. A Time Center Technical Report, December 1999.

[Wil82] Wille, R.: Reestructuring lattice theory: an approach based on hierarchies of concepts. *Ordered Sets* (I. Rival, ed.). pages 445-470. 1982.

[ZPOL97] Zaki, M. – Parthasarathy, S.- Ogihara, M.-Li, W.: New Algorithms for Fast Discovery of Association Rules. Proc 3rd. Int'l Conf. on KDD (KDD-97), August 1997.

[ZO98] Zaki, M.- Ogihara, M.: Theoretical Foundations of Association Rules. ACM SIGMOD 3rd. Workshop in DMKD, June 1998.

[ZMTA02] Zimbrao, G.- Moreira de Souza, J. – Texeira de Almeida, V. –Araújo da Silva, W.: An Algorithm to Discover Calendar-based Temporal Rules with Item's Lifespan Restriction. ACM 2nd Workshop on Temporal Data Mining held in KDD-2002, Edmonton, July 2002.