Descubrimiento de Patrones Similares Frecuentes para la Minería de Reglas de Asociación sobre Datos Mezclados

Por

Ansel Yoan Rodríguez González

Tesis sometida como requisito parcial para obtener el grado de

DOCTOR EN CIENCIAS EN LA ESPECIALIDAD DE CIENCIAS COMPUTACIONALES

en el

Instituto Nacional de Astrofísica, Óptica y Electrónica
Tonantzintla, Puebla
Marzo 2011

Supervisada por:

Dr. José Francisco Martínez Trinidad
Investigador titular del INAOE

El destino sólo está escrito hasta el presente y el futuro se crea en cada instante.

A mi hija Anna Karla.

Agradecimientos

Sería incorrecto e ingrato expresar que esta tesis es el resultado de tres años de intenso trabajo y sacrificio. Esta tesis es resultado de un largo camino recorrido, que comenzó cuando por primera vez abrí los ojos y vi la luz en este mundo y no concluye con la defensa de la misma. Durante ese largo camino, muchas personas han compartido mi andar. Algunos durante todo el camino, otros durante tramos, y unos pocos han se han incorporado para no abandonarlo jamás. También están los que aunque físicamente ya no se encuentran entre nosotros, me siguen acompañando, pues los llevo en el corazón. Con sus acciones me han resultado en lo que hoy soy, y por tanto, aunque algunos anónimos, también son todos ellos autores de esta tesis.

Quiero agradecerles a todos y quisiera escribir todos sus nombres acá, pero las páginas necesarias serían muchas más que las de esta tesis. Véase más allá de las personas que explícitamente aparecerán en estos agradecimientos, cada palabra escrita en esta tesis como nombres anónimos. Sin ellos, esta tesis sería solamente hojas en blanco.

A Yuliet, mi esposa, por toda la comprensión, tener una paciencia inmensa y darme todo su amor, sin lo cual no hubiera podido enfocarme en esta investigación.

A Anna Karla, mi pequeña bebe, por darme toda la alegría del mundo, portarse muy bien y hasta dejarme dormir.

A Mercedes y Anselmo, mis padres, por la formación que me han dado, ser mis siempre mis maestros y haberme dedicado sus vidas.

A mi hermanita por su sinceridad, y porque la quiero mucho.

A mis abuelos, tíos y primos por tenerme siempre en cuenta y quererme mucho.

A Lourdes y Alberto por acogerme como un hijo.

A mis maestros y profesores, desde aquellos que me enseñaron las primeras letras y números hasta a los de mi Alma Mater, la Universidad de Habana.

A Heribo y Ernesto Bennette, a quienes les debo parte importante de mi formación profesional y confiaron siempre en mí.

Al Centro de Aplicaciones de Tecnologías de Avanzada, pues fue mi escuela como investigador y aprendí mucho allí.

Al Dr. José Ruiz Shulcloper, por iniciarme en el tema, dedicarme su tiempo y experiencia, y apoyarme durante gran parte del tiempo de esta tesis.

Al Dr. José Francisco Martínez Trinidad y al Dr. Jesús Ariel Carrasco Ochoa, mis asesores, por su oportuna guía en el desarrollo de esta tesis, sus críticas constructivas y todas las enseñanzas, y sobre todo por haberme dado siempre su apoyo.

Al los Doctores Eduardo Morales Manzanares, Carlos Alberto Reyes García, Leopoldo Altamirano Robles, Gustavo Rodríguez Gómez y Michael Berry, por su labor como sinodales.

A todos los mexicanos y amigos pues han estado en todo momento disponibles para brindar su ayuda, en especial Margarita Flores y Efrén Cielo, y Carmen Mesa.

A los cubanos del instituto pues siempre podemos contar unos con los otros.

A Víctor, Cori y toda su familia pues ya son también la mía.

Por último y no menos importantes:

Al INAOE por haberme permitido formar parte de su estudiantado y la formación que me ha brindado.

A CONACyT por el apoyo económico sin el cual hubiera sido imposible llegar a este punto.

A México por haberme acogido y brindado su cariño.

Y también todos aquellos que no confiaron en mí y en que este momento sería posible, o no dieron su apoyo o lo retiraron, pues me dieron razones y fuerzas para crecerme, ponerme nuevas metas y seguir adelante.

Resumen

La Minería de Reglas de Asociación es una tarea importante del descubrimiento de conocimiento en datos. La misma ha sido aplicada en mercadeo, análisis de crímenes, bioinformática, medicina, seguridad de redes, etc. El objetivo de la Minería de Reglas de Asociación es encontrar asociaciones interesantes de la forma "si antecedente entonces consecuente", entre combinaciones de los valores de los atributos que describen a los objetos de una colección de datos. Comúnmente, una regla de asociación es interesante si su frecuencia y su confianza[1] son mayores o iguales que umbrales de frecuencia y confianza especificados por el usuario.

Generalmente, minar reglas de asociación, consiste en: I) Encontrar todos los patrones frecuentes (descripciones de objetos cuya frecuencia es mayor o igual que un umbral de mínima frecuencia); II) Extraer las reglas de asociación interesantes a partir de los patrones frecuentes.

El primer paso (también llamado Minado de Patrones Frecuentes) es el más costoso computacionalmente. Como consecuencia, muchos trabajos se han enfocado en este problema. Los patrones frecuentes representan regularidades que aparecen en los datos. En dependencia del área de aplicación, estos patrones pueden ser interpretados como perfiles de usuarios, modus operandi, síndromes o factores de riesgo, entre otros. Los patrones frecuentes también han sido usados en otras tareas de minería de datos, diferentes del minado de reglas de asociación, como la clasificación y el agrupamiento.

En el enfoque tradicional de minado de patrones frecuentes y de reglas de asociación, las colecciones de datos están descritas exclusivamente por atributos Booleanos. Sin embargo, existen áreas de aplicación como geología, medicina, biología y sociología donde las colecciones de datos pueden contener objetos descritos simultáneamente por atributos numéricos y no numéricos (Datos Mezclados). Adicionalmente, en muchas aplicaciones, dos objetos casi nunca son exactamente iguales, y por lo tanto, para compararlos se utilizan funciones de semejanza diferentes de la igualdad.

En la literatura sólo se ha reportado un algoritmo para el minado de reglas de asociación usando funciones de semejanzas diferentes de la igualdad, el cual encuentra reglas de asociación ocultas para el enfoque tradicional. Sin embargo, el mismo fue diseñado para funciones de semejanza Booleana que cumplan que: si dos objetos no son semejantes

[1]Por confianza de una regla de asociación se entiende cuánto representa la frecuencia de la regla, de la frecuencia del antecedente de la regla.

respecto a un conjunto de atributos, tampoco lo son respecto a un superconjunto de éste. No obstante, existen problemas en los cuales la función de semejanza entre descripciones y subdescripciones de objetos no satisface esta propiedad. Más aún, existen problemas en los que las funciones de semejanza no son Booleanas.

Los algoritmos propuestos en esta tesis para minar patrones frecuentes usan funciones de semejanza menos restrictivas, que las usadas por el algoritmo existente. Además, se introducen nuevas propiedades que permiten podar el espacio de búsqueda de patrones similares frecuentes, así como una estructura de datos que reduce el número de evaluaciones de la función de semejanza, tanto para funciones de semejanza Booleana, como para funciones de semejanza no Booleana. También, se adapta el algoritmo *GenRules* para generar reglas de asociación interesantes a partir de patrones similares frecuentes.

De acuerdo con nuestros experimentos, los algoritmos propuestos obtienen conjuntos de patrones frecuentes de mayor calidad que los obtenidos por el algoritmo existente y por los algoritmos del enfoque tradicional de minado de patrones frecuentes. Además, el algoritmo propuesto, que permite el mismo tipo de funciones de semejanza que el algoritmo existente, es más rápido que éste.

Por otro lado, nuestros experimentos también muestran que al minar reglas de asociación a partir de patrones frecuentes usando como función de semejanza la igualdad (como en el enfoque tradicional) se pueden perder reglas de asociación interesantes y más aún, pueden generarse reglas de asociación que no serían reglas de asociación interesantes si se usara una función de semejanza diferente de la igualdad. Un efecto similar ocurre cuando la función de semejanza entre los objetos no es Booleana y las reglas de asociación son obtenidas a partir de los patrones encontrados mediante la Booleanización de esta función.

Abstract

Association Rule Mining is an important task in Knowledge Discovery from Data. It has been applied to marketing, crime analysis, bioinformatics, medicine, network security, etc. The aim of Association Rule Mining is finding interesting "if-then" rules between combinations of feature values that describe the objects in a dataset. Commonly, an association rule is considered interesting if its frequency and confidence[2] are greater than or equal to user-specified frequency and confidence thresholds.

Usually, mining association rules consists in: I) Searching frequent patterns (descriptions of objects whose frequency is greater than or equal to a minimum threshold frequency); II) Extracting interesting association rules from frequent patterns.

The first step (also called Frequent Pattern Mining) is the most computationally expensive. Consequently, many works have focused on this problem. Frequent patterns represent regularities that appear in the data. Depending on the application area, these patterns could be interpreted as user profiles, modus operandi, syndromes and risk factors, among others. Frequent patterns have also been used in other data mining tasks, different from association rule mining, such as classification and clustering.

In the traditional approach for mining frequent patterns and mining association rules, datasets are described only by Boolean features. However, there are application areas like geology, medicine, biology and sociology where datasets may contain objects described simultaneously by numerical and non numerical features (Mixed Data). Additionally, in many applications, two objects are almost never exactly equal, and therefore similarity functions different from the equality are used to compare objects.

The literature reports only one algorithm for association rule mining using similarity functions different from the equality, which finds association rules hidden for the traditional approach. Nevertheless, this algorithm was designed for Boolean similarity functions that satisfy that: if two objects are not similar with respect to a set of features, then they are not similar with respect to any superset of it. However, there are problems where the similarity function between object descriptions and subdescripciones does not satisfy this property. Moreover, there are problems where the similarity functions are not Boolean.

The algorithms proposed in this thesis for mining frequent patterns use similarity

[2]The confidence of a rule is how much represent the frequency of the rule respect to the frequency of the rule antecedent.

functions less restrictive than those used by the existing algorithm. In addition, new properties that allow to prune the search space of frequent similar patterns, a data structure that reduces the number of similarity function evaluations, for Boolean and non Boolean similarity functions, are introduced. Also, we adapt the *GenRules* algorithm to generate interesting association rules from frequent similar patterns.

According to our experiments, the proposed algorithms obtain sets of frequent patterns with higher quality than those obtained by both the existing algorithm and the traditional approach algorithms for mining frequent patterns. In addition, the proposed algorithm, which allows the same type of functions allowed by the existing algorithm is faster than it.

On the other hand, our experiments also show that mining association rules from frequent patterns using the equality as similarity function (as in the traditional approach), has as a consequence that interesting association rules could be lost and even more, some association rules, which would not be interesting association rules if a similarity function different from the equality were used, could be generated. A similar effect occurs when the similarity function between objects is not Boolean and association rules are obtained from patterns found through the Booleanization of this function.

Índice general

Índice de figuras

Índice de tablas

Índice de algoritmos

Capítulo 1

Introducción

Hoy en día, debido a los rápidos avances científicos y tecnológicos, se generan y almacenan diariamente grandes volúmenes de información digital como: correos electrónicos, llamadas telefónicas, música, videos, libros electrónicos, información de clientes o usuarios, entre otros. Dada esta situación, la cantidad actual de información existente en muchos contextos supera la capacidad humana de discernir información útil para realizar análisis y tomar decisiones.

Tal como en la minería tradicional donde se pretende encontrar minerales valiosos entre montones y montones de rocas, en la Minería de Datos se pretende encontrar o descubrir información útil en conjuntos de datos.

La Minería de Reglas de Asociación es una técnica de la Minería de Datos que consiste en encontrar asociaciones interesantes en forma de relaciones de implicación entre valores de los atributos de los objetos de un conjunto de datos. Numerosos y recientes estudios [Alatas et al., 2008; Huand et al., 2008; Jian-min and Xiao-ding, 2010; Kalpana and Nadarajan, 2008; LaRosa et al., 2008; López et al., 2008; Nan et al., 2009; Patil et al., 2010; Shen et al., 2010; Yunyan and Juan, 2010; Zhang et al., 2007] avalan su actualidad e importancia, así como su aplicación en diferentes ámbitos como: mercadeo, bioinformática, medicina y seguridad de redes entre otras áreas.

La Minería de Reglas de Asociación emergió en la década de los 90´s con una aplicación práctica, el análisis de información de ventas para el mercadeo [Agrawal et al., 1993; Agrawal and Srikant, 1994]. Mediante esta técnica se descubrían las relaciones interesantes en los datos recopilados a gran escala por los sistemas de terminales de punto de venta de supermercados. Los datos consistían en colecciones de transacciones, también conocidas como bases de datos transaccionales, donde cada transacción expresa qué productos compró un cliente. Un ejemplo de este tipo de colecciones se muestra en la tabla 1.1.

En este contexto, una regla de asociación podría ser "Si un cliente compra pan y leche, entonces también compra mantequilla", formalmente:

$$(Pan \wedge Leche) \rightarrow (Mantequilla)$$

Tabla 1.1: Ejemplo de una colección de transacciones.

ID	***Productos***
1	*Leche, Pan*
2	*Pan, Mantequilla*
3	*Cerveza*
4	*Leche, Pan, Mantequilla*
5	*Pan*
6	*Leche, Pan, Mantequilla*

Existen varias formas de medir el interés de una regla [Geng and Hamilton, 2006]. Sin embargo, generalmente el interés de una regla de asociación está dado por su soporte (denotado como *supp*) y su confianza (denotada como $conf$), entendiéndose por soporte la frecuencia de aparición, en la colección, de la combinación de productos involucrados en la regla. Por ejemplo, para la colección mostrada en la tabla 1.1 se tiene que:

$$supp((Pan \wedge Leche) \rightarrow (Mantequilla)) = supp(Pan \wedge Leche \wedge Mantequilla) = \frac{2}{6}$$

Por confianza de una regla entendemos cuánto representa el soporte de la regla, del soporte del antecedente de la regla. Por ejemplo para la colección mostrada en la tabla 1.1 se tiene que:

$$conf((Pan \wedge Leche) \rightarrow (Mantequilla)) = \frac{supp(Pan \wedge Leche \wedge Mantequilla)}{supp(Pan \wedge Leche)} = \frac{2}{3}$$

Se considera que una regla es interesante si su soporte y su confianza son mayores o iguales que ciertos umbrales de mínimo soporte y mínima confianza especificados. Este tipo de reglas fue denominado *Reglas de Asociación Binarias* debido a que los objetos de la colección (transacciones) están descritos exclusivamente por atributos Booleanos. Nótese que usualmente cada transacción de la colección se describe a través de atributos Booleanos, uno por cada producto, tal que cada atributo representa si el producto se compró o no. Varios han sido los algoritmos desarrollados para el minado de *Reglas de Asociación Binarias* [Agrawal et al., 1993; Agrawal and Srikant, 1994; Holt and Chung, 2001, 2002; Park et al., 1997; Savasere et al., 1995; Zaki et al., 1997].

La Minería de Reglas de Asociación consiste de dos pasos fundamentales: I) Encontrar las combinaciones frecuentes de valores de atributos (patrones frecuentes); II) Construir las reglas de asociación interesantes a partir de los patrones frecuentes. El primer paso es el paso crítico, pues no obstante a la simplicidad del dominio de los atributos, la cardinalidad del espacio de búsqueda de patrones frecuentes crece exponencialmente respecto al número de atributos. Este hecho afecta la eficiencia de la minería de reglas de asociación.

Existen otras situaciones que también dificultan el minado de reglas de asociación:

- Generalmente las colecciones de datos contienen objetos descritos simultáneamente por atributos numéricos y no numéricos (Datos Mezclados).
- Rara vez dos objetos del mundo real son idénticos, por lo cual, funciones de semejanza diferentes de la igualdad son comúnmente usadas para comparar descripciones de objetos.

Cuando los objetos son descritos por Datos Mezclados [Ruiz-Shulcloper, 2009], la cardinalidad del espacio de búsqueda de patrones frecuentes puede ser mayor que cuando los objetos son descritos por atributos Booleanos. En general, si la cardinalidad del dominio de los atributos es muy grande o infinita, entonces la probabilidad de que el valor de un atributo sea frecuente tiende a ser baja. Para atacar este problema, se han propuesto varios trabajos que conforman al que denominaremos enfoque de discretización duro, en los que el dominio de los atributos numéricos es discretizado para representar a los objetos mediante atributos Booleanos, es decir, el problema se transforma mediante la discretización del dominio de los atributos numéricos al enfoque tradicional de minado de reglas de asociación [Srikant and Agrawal, 1996; Salleb-Aouissi et al., 2007]. Sin embargo, discretizar por medio de conjuntos duros no resuelve el problema, pues algunas veces estas transformaciones son realizadas sin considerar la semántica de los datos y como consecuencia la naturaleza de los mismos es cambiada. Otro enfoque para minar patrones frecuentes en colecciones con datos mezclados, que considera la semántica de los datos, es al que denominaremos enfoque de discretización difuso. Este enfoque conlleva a la determinación de la función de pertenencia para cada uno de los conjuntos difusos por cada atributo. No obstante, para obtener buenos resultados las funciones de pertenencia deben ser definidas cuidadosamente por expertos humanos del área de estudio [Kuok et al., 1998; Hong and Lee, 2008].

Por otro lado, el concepto de semejanza es una herramienta metodológica, para el análisis de datos, utilizada en las ciencias poco formalizadas como la Geología [Gómez et al., 1994], Medicina [Ortiz-Posadas et al., 1994], Sociología [J. Ruiz-Shulcloper, 1981], etc., para tomar decisiones; en este contexto los objetos son comúnmente descritos por datos mezclados. Por ejemplo, se pudiera considerar que dos personas son semejantes en términos de sus edades si ellas son de la misma generación, lo cual podría ser equivalente a considerar que dos edades son semejantes si el valor absoluto de sus diferencias es a lo sumo 5 años. Observe que este criterio de semejanza es diferente al de intervalos de edades (grupos etarios). En este ejemplo, la semejanza es utilizada para comparar valores de un atributo en los objetos de estudio, no obstante, la semejanza puede ser usada para comparar objetos completos o partes de ellos. Por ejemplo, se puede considerar que dos objetos son semejantes si ellos son semejantes en todos los atributos o si ellos son semejantes en al menos 90 % de los atributos.

Los primeros avances en la Minería de Reglas de Asociación usando funciones de semejanza entre descripciones y subdescripciones de objetos se reportan en [Dánger et al., 2004]. El algoritmo propuesto fue diseñado sólo para un conjunto restringido de funciones de semejanza Booleana y la eficiencia del mismo no fue evaluada. No obstante,

dicho trabajo muestra cómo mediante la incorporación del concepto de semejanza en el cálculo de la frecuencia, pueden ser descubiertas reglas de asociación ocultas para el enfoque tradicional de minando de reglas de asociación, en el cual la igualdad es usada como función de semejanza para comparar descripciones y subdescripciones de los objetos.

La presente tesis se enfoca al estudio de la minería de Patrones Frecuentes y Reglas de Asociación usando funciones de semejanza para comparar descripciones y subdescripciones de los objetos. Es importante resaltar que en esta tesis no se utilizará el enfoque tradicional de minado de reglas de asociación, ni los enfoques discretización, sino que, se explorará el uso de funciones de semejanza entre objetos completos o partes de ellos para minar Patrones Frecuentes y Reglas de Asociación en colecciones de datos mezclados.

1.1. Problemática actual

Hasta el momento de inicio de esta tesis sólo se había propuesto un algoritmo para el minado de reglas asociación que usa funciones de semejanza diferentes de la igualdad. Mediante dicho algoritmo pueden encontrase reglas de asociación ocultas para el enfoque tradicional. Sin embargo, el mismo, sólo permite el uso de funciones de semejanza Booleana que cumplan que: si dos objetos no son semejantes respecto a un conjunto de atributos tampoco lo son respecto a un superconjunto de éste. No obstante, existen problemas en los cuales la semejanza entre descripciones y subdescripciones de objetos no cumple con la propiedad antes mencionada. Más aún, existen problemas en los que las funciones de semejanza no son Booleanas.

Por tal motivo, el problema que será resuelto en esta investigación es minar patrones frecuentes y reglas de asociación en colecciones de datos mezclados, considerando la semejanza entre descripciones y subdescripciones de objetos, permitiendo el uso de funciones de semejanza menos restrictivas que las permitidas por el único algoritmo existente.

1.2. Objetivo general

Con base en lo antes mencionado, el objetivo general de este trabajo de investigación es el siguiente:

> Diseñar algoritmos para extraer patrones frecuentes y reglas de asociación en colecciones de datos mezclados, incorporando el concepto de semejanza entre descripciones y subdescripciones de objetos, que permitan el uso de funciones de semejanza menos restrictivas que las permitidas por el único algoritmo existente.

1.3. Objetivos particulares

Los objetivos específicos de este trabajo de investigación son los siguientes:

1. Extender los conceptos de frecuencia y confianza, patrón frecuente y regla de asociación, incorporando el concepto de semejanza entre descripciones y subdescripciones de objetos con datos mezclados.

2. Desarrollar algoritmos de búsqueda de patrones similares frecuentes en colecciones de datos mezclados para funciones de semejanza Booleana y no Booleana, a partir de las extensiones de los conceptos de frecuencia y patrón frecuente.

3. Definir propiedades de poda del espacio de patrones similares frecuentes.

4. Diseñar algoritmos eficientes de búsqueda de patrones similares frecuentes en colecciones de datos mezclados para funciones de semejanza Booleana y no Booleana, que incorporen en su estrategia de búsqueda las propiedades de poda encontradas.

5. Proponer un algoritmo de búsqueda de reglas de asociación, a partir de los patrones similares frecuentes encontrados por los algoritmos anteriores.

1.4. Organización de esta tesis

La manera en que está organizado el contenido de este documento es la siguiente: En el capítulo 2 se describen los trabajos más relevantes relacionados con el minado de reglas de asociación. Estos trabajos son agrupados atendiendo al dominio de los atributos de las colecciones que procesan. Para cada caso, se presentan los conceptos básicos, se define el problema de la minería de reglas de asociación y se muestran las diferentes estrategias con las que el problema se ha abordado.

En el capítulo 3 se presentan los conceptos básicos requeridos para definir el problema de minado de patrones frecuentes usando funciones de semejanza Booleana. Se extienden los conceptos de frecuencia y confianza, patrón frecuente y regla de asociación, incorporando el concepto de semejanza Booleana entre descripciones y subdescripciones de objetos con datos mezclados. Se presentan y demuestran nuevas propiedades que permiten la poda del espacio de búsqueda de patrones similares frecuentes. Se introduce una nueva estructura de datos y se describe cómo ésta puede ser utilizada en la minería de patrones similares frecuentes. Además, se proponen tres nuevos algoritmos de minado de patrones similares frecuentes y se adapta el algoritmo de minado de reglas de asociación Binarias al minado de reglas de asociación incorporando el concepto de semejanza Booleana entre descripciones y subdescripciones de objetos con datos mezclados.

En el capítulo 4 se extienden los resultados obtenidos en el capítulo anterior, al minado de patrones similares frecuentes y reglas de asociación incorporando el concepto

de semejanza no Booleana entre descripciones y subdescripciones de objetos con datos mezclados.

El capítulo 5 muestra los resultados experimentales obtenidos al evaluar el desempeño de los algoritmos propuestos y una comparación experimental contra el enfoque tradicional y el único algoritmo existente para el minado de patrones frecuentes usando funciones de semejanza.

Finalmente, se exponen las conclusiones, las aportaciones del presente trabajo de investigación y algunas direcciones a seguir como trabajo futuro.

Capítulo 2

Estado del Arte

Como se comentó en el capítulo anterior, la minería de reglas de asociación se restringió inicialmente a colecciones de datos Booleanos. Posteriormente esta técnica se desarrolló también para colecciones de datos mezclados. En este capítulo se presentan los trabajos relacionados con la minería de reglas de asociación para ambos tipos de colecciones.

2.1. Reglas de Asociación en colecciones de datos Booleanos

El minado de Reglas de Asociación en colecciones de datos Booleanos (enfoque tradicional de minado de Reglas de Asociación) fue introducido en [Agrawal et al., 1993]. Formalmente se conceptualiza de la siguiente manera: sea $I = \{i_1, \ldots, i_m\}$ un conjunto de ítems, $T = \{t_1, \ldots, t_n\}$ un conjunto de transacciones, cada una contiene ítems del conjunto I, es decir, cada transacción t_i es un conjunto de ítems tal que $t_i \subseteq I$. Una regla de asociación es una implicación de la forma $X \rightarrow Y$, donde $X \subseteq I$, Y$\subseteq I$ y $X \cap Y = \emptyset$. El problema de minería de Reglas de Asociación Binarias consiste en encontrar todas las reglas interesantes a partir de un conjunto de transacciones.

Aunque existen varias formas de medir el interés de una regla de asociación [Geng and Hamilton, 2006], comúnmente se mide mediante su soporte y su confianza [Agrawal et al., 1993].

Definición 2.1. El soporte de un conjunto de ítems X en un conjunto de transacciones T, es la fracción de transacciones de T que contienen los ítems de X.

$$supp(X) = \frac{|\{t \in T \mid X \subseteq t\}|}{|T|}$$

Definición 2.2. Se dice que un conjunto de ítems X es frecuente en un conjunto de transacciones T, si y sólo si su soporte en T es mayor o igual que un umbral de mínimo

soporte $minSupp$.

Definición 2.3. El soporte de una regla de asociación $X \Rightarrow Y$ en un conjunto de transacciones T, es la fracción de transacciones de T que contienen a los ítems de $X \cup Y$.

$$supp(X \rightarrow Y) = supp(X \cup Y)$$

Definición 2.4. La confianza de una regla de asociación $X \Rightarrow Y$ es la fracción de transacciones de T que conteniendo a X, también contienen a Y.

$$conf(X \rightarrow Y) = \frac{supp(X \cup Y)}{supp(X)}$$

Definición 2.5. Se dice que una regla de asociación $X \Rightarrow Y$ es interesante en un conjunto de transacciones T, si y sólo si su soporte es mayor o igual que un umbral de mínimo soporte $minSupp$ y su confianza es mayor o igual que un umbral de mínima confianza $minConf$.

El proceso de minado de reglas de asociación consta de dos pasos fundamentales. Primero se buscan en el conjunto de transacciones, los conjuntos frecuentes de ítems [Agrawal et al., 1993; Agrawal and Srikant, 1994; Han et al., 2000; Pietracaprina and Zandolin, 2003; Gopalan and Sucahyo, 2004; Grahne and Zhu, 2005; Song and Rajasekaran, 2006; Kalpana and Nadarajan, 2008]. Nótese que para que una regla $X \Rightarrow Y$ sea interesante $X \cup Y$ debe ser un conjunto frecuente de ítems. Luego a partir de los conjuntos frecuentes de ítems son generadas las reglas de asociación interesantes [Agrawal and Srikant, 1994; Holt and Chung, 1999, 2001, 2002].

2.1.1. Generación de conjuntos frecuentes de ítems

La generación de los conjuntos frecuentes de ítems es el paso más costoso del proceso de minado de reglas de asociación y en él se enfocan la mayoría de los trabajos. Esto se debe a que el tamaño del espacio de búsqueda de los conjuntos frecuentes de ítems depende exponencialmente del tamaño del conjunto de ítems I ($|\{X \mid X \subseteq I, X \neq \emptyset\}| = 2^{|I|} - 1$). Sin embargo, debido a la siguiente propiedad, no es necesario recorrer todo el espacio de búsqueda para encontrar los conjuntos frecuentes de ítems.

Propiedad 2.1 (Clausura Descendente del soporte)**.** Todo subconjunto de un conjunto frecuente de ítems es frecuente, mientras que todo superconjunto de un conjunto no frecuente de ítems tampoco es frecuente.

Como consecuencia de esta propiedad, el espacio de búsqueda asociado con el retículo que forman los subconjuntos de ítems, es dividido por una frontera en dos subespacios: el subespacio que sólo contiene conjuntos frecuentes de ítems y el subespacio que sólo contiene conjuntos no frecuentes de ítems.

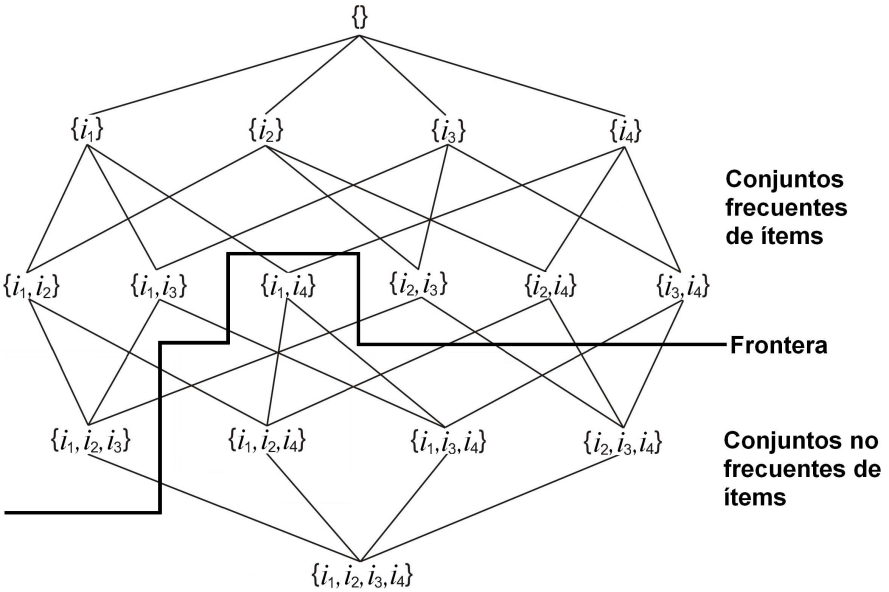


Figura 2.1: Retículo formado por el conjunto de ítems $I = \{i_1, i_2, i_3, i_4\}$.

La figura 2.1 muestra el retículo formado para un conjunto de ítems $I = \{i_1, i_2, i_3, i_4\}$, así como la frontera entre el subespacio que sólo contiene conjuntos frecuentes de ítems y el subespacio que sólo contiene conjuntos no frecuentes de ítems, para el conjunto de transacciones $T = \{\{i_1, i_2, i_3\}, \{i_2, i_4\}, \{i_3, i_4\}, \{i_1, i_2, i_3, i_4\}\}$ dado el umbral de mínimo soporte $minSupp = 0{,}5$.

Existen varias estrategias para recorrer el retículo que forman los subconjuntos de ítems. Estas estrategias pueden clasificarse atendiendo a la dirección de los recorridos en el retículo como:

- Descendentes: El recorrido se realiza desde el conjunto vacío hasta la frontera.
- Ascendentes: El recorrido se realiza en el sentido opuesto, desde el conjunto completo de ítems I hasta la frontera.

Estas estrategias a su vez pueden generar los conjuntos de ítems en dos formas:

- En amplitud: Se generan todos los conjuntos frecuentes de ítems de tamaño k antes de generar los conjuntos de ítems de tamaño $k + 1$ [Agrawal and Srikant, 1994; Savasere et al., 1995; Park et al., 1997; Holt and Chung, 1999, 2001, 2002].
- En profundidad: Recursivamente se generan los conjuntos ítems por cada rama de la estructura arbórea que se deriva del retículo [Jav; Zaki et al., 1997; Han et al., 2000; Borgelt, 2003; Pietracaprina and Zandolin, 2003; Sucahyo and Gopalan, 2003; Erwin et al., 2007].

El primer algoritmo que hace uso de la propiedad de Clausura Descendente del Soporte para podar el espacio de búsqueda de los conjuntos de ítems fue propuesto en [Agrawal and Srikant, 1994] y se conoce como *Apriori* (Algoritmo 2.1). A partir de éste se han derivado toda una clase de algoritmos denominados tipo *Apriori*.

Procedimiento `Apriori`(D, $minSupp$)

Input: D - Colección de transacciones, $minSupp$ - Umbral de mínimo soporte.
Output: F - Conjuntos frecuentes de ítems.

```
1  F ← ∅
2  L_1 ← {i | |{t ∈ D | i ∈ t}| ≥ minSupp}
3  k ← 2
4  while L_{k-1} ≠ ∅ do
5  |   C_k ← {c | JOIN(c, L_{k-1}) ∧ PRUNE(c, L_{k-1})}
6  |   foreach transaction t ∈ D do
7  |   |   C_t ← {c ∈ c_k | c ∈ t}
8  |   |   foreach candidate c ∈ C_t do
9  |   |   |   c.support ← c.support + 1
10 |   |   end
11 |   end
12 |   L_k ← {c ∈ c_k | c.support ≥ minSupp}
13 |   F ← F ∪ L_k
14 |   k ← k + 1
15 end
```

Algoritmo 2.1: Algoritmo *Apriori*.

En este algoritmo, L_k contiene los conjuntos frecuentes de ítems de tamaño k y C_k los conjuntos de ítems candidatos a frecuentes de tamaño k. Primero son obtenidos y almacenados en L_1 los conjuntos frecuentes de ítems de tamaño 1. Posteriormente, en cada iteración (k) del algoritmo, utilizando la propiedad de Clausura Descendente del Soporte (Propiedad 2.1), se generan los conjuntos de ítems de tamaño k, candidatos a frecuentes, combinando los conjuntos frecuentes de ítems de tamaño $k-1$; y a partir de los conjuntos de ítems de tamaño k candidatos a frecuentes, son seleccionados los conjuntos frecuentes de ítems de tamaño k. Este proceso se repite hasta que, al comienzo de una iteración, el conjunto de conjuntos frecuentes de ítems de tamaño $k-1$ sea vacío. Para generar los conjuntos de ítems candidatos a frecuentes se utilizan las operaciones `JOIN` y `PRUNE`:

$$\texttt{JOIN}(\{i_1, ..., i_k\}, L_{k-1}) \equiv \langle \{i_1, ..., i_{k-2}, i_{k-1}\} \in L_{k-1} \wedge \{i_1, ..., i_{k-2}, i_k\} \in L_{k-1} \rangle$$

$$\texttt{PRUNE}(c, L_{k-1}) \equiv \langle \forall s[s \subset c \wedge |s| = k-1] \Rightarrow [s \in L_{k-1}] \rangle$$

La operación `JOIN` consiste en tomar todos los pares de conjuntos de ítems de tamaño $k-1$ que coincidan en sus $k-2$ primeros ítems y generar conjuntos de ítems de tamaño k manteniendo los $k-2$ ítems comunes y adicionando, en orden lexicográfico, los $(k-1)$-

ésimos ítems de los dos conjuntos que se unen. La operación `PRUNE` consiste en aplicar la propiedad de Clausura Descendente del soporte para podar los conjuntos de ítems de tamaño k que tengan al menos un subconjunto no frecuente de ítems de tamaño $k-1$.

El algoritmo *Apriori* a pesar de podar el espacio de búsqueda mediante la propiedad de Clausura Descendente del soporte, presenta claras desventajas como son: la necesidad de mantener todo un nivel en memoria para generar los candidatos del siguiente nivel, y recorrer la colección de datos en cada iteración.

Otros algoritmos que hacen uso de la propiedad de Clausura Descendente son los basados en árboles [Han et al., 2000; Pietracaprina and Zandolin, 2003; Sucahyo and Gopalan, 2003, 2004; Ahmed and Coenen, 2006] y los derivados del algoritmo *ECLAT* [Zaki et al., 1997; Borgelt, 2003; Kim et al., 2003; Kalpana and Nadarajan, 2008].

Los algoritmos basados en árboles utilizan estructuras de datos arbóreas para almacenar de forma compacta la colección de datos y contar eficientemente las repeticiones de los conjuntos de ítems. Entre los algoritmos más significativos pertenecientes a esta clase se encuentran *FP-Growth* [Han et al., 2000], *Patricia Trie-Mine* [Pietracaprina and Zandolin, 2003], *CT-ITL* [Sucahyo and Gopalan, 2003], *CT-PRO* [Sucahyo and Gopalan, 2004] y *Apriori-TFP* [Ahmed and Coenen, 2006].

El algoritmo *FP-Growth* [Han et al., 2000], se basa en el crecimiento o extensión de los conjuntos frecuentes de ítems. En un primer recorrido de la colección de datos, se obtienen los conjuntos frecuentes de ítems de tamaño 1. En un segundo recorrido de la colección de datos, se inserta cada transacción, con los ítems ordenados descendentemente de acuerdo a su soporte, en una estructura de datos compacta denominada *FP-tree*, también conocida como árbol de prefijos. De esta forma, prefijos iguales, de transacciones diferentes, comparten la misma rama del árbol. Luego, a partir de esta estructura se generan los conjuntos de ítems frecuentes recorriendo recursivamente las ramas del árbol.

El algoritmo *Patricia Trie-Mine* [Pietracaprina and Zandolin, 2003], utiliza una estructura de datos, denominada *PatriciaTrie*, más compacta que la estructura *FP-tree*. *PatriciaTrie*, a diferencia de la estructura *FP-tree*, agrupa en cada nodo del árbol todos los nodos consecutivos que tienen igual valor de soporte. Los conjuntos de ítems frecuentes son generados de forma similar al algoritmo *FP-Growth*.

Los algoritmos *CT-ITL* y *CT-PRO* se basan en la estructura *FP-tree*. El algoritmo *CT-ITL* [Sucahyo and Gopalan, 2003] utiliza una estructura de datos denominada *CT-tree*, la cual modifica la estructura *FP-tree* para almacenar grupos de transacciones, mientras que, el algoritmo *CT-PRO* [Sucahyo and Gopalan, 2004], del mismo autor, utiliza una estructura denominada *CFP-tree*, que puede reducir a la mitad el número de nodos de la estructura *FP-tree*.

Otro algoritmo que utiliza estructuras de datos arbóreas es *Apriori-TFP* [Ahmed and Coenen, 2006]. En un primer recorrido de la colección de datos se construye una estructura de datos denominada *P-tree*, que almacena los soportes parciales de todos los conjuntos de ítems. El soporte parcial de un conjunto de ítems $I' \subset I$ es el número de transacciones de la colección, que teniendo ordenados lexicográficamente sus ítems, su prefijo es I'. A partir de la estructura anterior se construye una segunda estructura

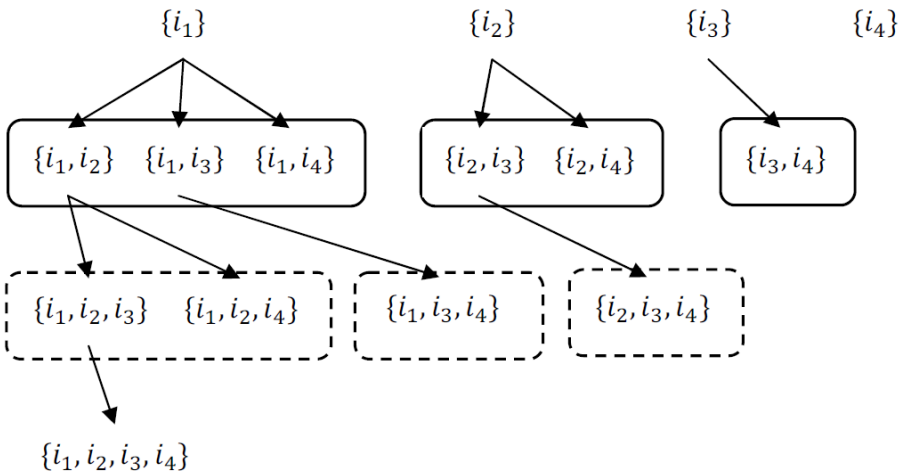


Figura 2.2: Clases de equivalencia.

de datos denominada *T-tree*. Al finalizar la construcción de la segunda estructura, los conjuntos frecuentes de ítems y sus soportes quedan almacenados en la misma. El soporte de cada conjunto de ítems es calculado a partir de su soporte parcial y del soporte parcial de cada conjunto de ítems que lo contiene.

Los algoritmos que utilizan estructuras de datos arbóreas, son muy eficientes en conjuntos de datos densos[1]. Sin embargo, en conjuntos de datos muy dispersos y grandes dichas estructuras resultan muy grandes debido a que compactan poco y los recorridos sobre ellas son muy costosos.

Los algoritmos derivados del algoritmo *ECLAT* [Zaki et al., 1997] definen subárboles de búsqueda, mediante las clases de equivalencia de los conjuntos de ítems.

Todos los conjuntos de ítems de tamaño k cuyos primeros $k-1$ ítems, según el orden lexicográfico, son iguales conforman una clase se equivalencia. Una clase de equivalencia de un conjunto $I' \subset I$ es el conjunto de los conjuntos resultantes de la unión de I' con cada ítem $X \in I$ lexicográficamente mayor que todo ítem en I'.

En la figura 2.2 se muestran para el conjunto de ítems $I = \{i_1, i_2, i_3, i_4\}$, las clases de equivalencia de los conjuntos de ítems $\{i_1\}$, $\{i_2\}$, $\{i_3\}$, $\{i_4\}$ con línea continua y el siguiente nivel de clases de equivalencia con línea punteada.

El algoritmo *ECLAT* [Zaki et al., 1997] se basa en el recorrido del retículo transformando recursivamente cada clase de equivalencia de tamaño k en clases de equivalencia de tamaño $k+1$. El soporte de cada nueva clase de equivalencia $I' \cup X$ se calcula a la vez que se obtiene, mediante la intersección de las listas de los identificadores de las transacciones que contienen a I' y a X respectivamente. Una debilidad de este algoritmo es que procesa un gran número de conjuntos de ítems. Una implementación eficiente del mismo es propuesta en [Borgelt, 2003], mientras en [Kim et al., 2003] un nuevo algoritmo

[1]Los conjuntos de datos densos, a diferencia de los dispersos, tienen un promedio de ítems por transacción del 30 % o más, con respecto al total de ítems.

basado en el concepto de unión virtual evita las relativamente costosas operaciones de unión usadas en el algoritmo original.

Siguiendo una estrategia similar de recorrido del retículo y haciendo uso de las clases de equivalencia en [Hernández-León et al., 2010] se propone el algoritmo *CA* en el cual las listas de identificadores de las transacciones que contienen a las clases de equivalencias son almacenadas de forma compacta en bloques de bits. La intersección de las mismas se realiza usando operadores lógicos sobre los bloques y sobre los índices de los bloques.

Otros algoritmos que recorren el retículo haciendo uso de las clases de equivalencia son *HybridMiner I* e *HybridMiner II* [Kalpana and Nadarajan, 2008]. Ambos alternan repetidamente una fase de búsqueda ascendente con una fase de búsqueda descendente, *HybridMiner I* comienza con la fase de búsqueda ascendente, mientras que *HybridMiner II* comienza con la fase de búsqueda descendente. En la fase de búsqueda ascendente, partiendo del conjunto I son buscados en amplitud todos los conjuntos frecuentes maximales de ítems [2] de tamaño $|I|, |I|-1, |I|-2...$, mientras no se encuentre al menos uno. En la fase de búsqueda descendente, partiendo de los conjuntos de ítems de tamaño 2 que contienen los ítems faltantes en los conjuntos frecuentes maximales de ítems obtenidos en la fase ascendente, son buscados en amplitud todos los conjuntos frecuentes minimales de ítems[3] de tamaño $2, 3, 4...$, mientras no se encuentre al menos uno. La posterior fase de búsqueda ascendente no tiene en cuenta aquellos conjuntos de ítems que contienen alguno de los conjuntos no frecuentes minimales de ítems encontrados en la fase de búsqueda descendente.

HybridMiner I e *HybridMiner II* logran una reducción del espacio de búsqueda de casi el 50 % respecto al algoritmo *ECLAT*. Sin embargo, la eficiencia de *HybridMiner II* es inferior a la de *ECLAT* y aunque la eficiencia de *HybridMiner I* es superior a la *ECLAT*, la diferencia es muy pequeña.

Una desventaja común en todos los algoritmos de minado de conjuntos frecuentes de ítems está relacionada con su aplicabilidad. En colecciones de datos donde los atributos que describen los objetos no son Booleanos, dichos algoritmos no se pueden emplear directamente. Para emplearlos es necesario transformar los datos originales en datos Booleanos, lo cual puede conllevar a perder información o a cambiar el problema original; y por tanto los resultados que se obtienen pueden no ser confiables.

2.1.2. Generación de reglas de asociación interesantes

El método más común de generación de reglas de asociación interesantes fue propuesto en [Agrawal and Srikant, 1994]. El mismo consiste en, por cada conjunto frecuente de ítems, generar todas las reglas posibles separando el conjunto de ítems en dos subconjuntos disjuntos (Algoritmo 2.2).

El algoritmo *GenRules*, al tener como entrada un conjunto de conjuntos frecuentes

[2]Un conjunto de ítems es frecuente maximal si no existe ningún super conjunto suyo que sea frecuente

[3]Un conjunto de ítems es frecuente minimal si no es superconjunto de ningún otro no frecuente.

Procedimiento `GenRules`(F, $minConf$)

Input: F - Conjuntos frecuentes de ítems, $minConf$ - Umbral de mínimo soporte.
Output: RA - Reglas de asociación interesantes.

```
1 RA ← ∅
2 foreach itemset Z ∈ F do
3 |   foreach itemset X ⊂ Z such that X ≠ ∅ do
4 |   |   if Z.support / X.support ≥ minConf then
5 |   |   |   RA = RA ∪ {X → (Z/X)}
6 |   |   end
7 |   end
8 end
```

Algoritmo 2.2: Algoritmo *GenRules*.

Tabla 2.1: Ejemplo de colección de datos mezclados.

Ω	*Edad*	*Auto*	*Casado*
1	23	*Compacto*	*No*
2	25	*Grande*	*No*
3	25	*Mediano*	*No*
4	29	*Mediano*	*No*
5	34	*Grande*	*Si*
6	38	*Lujoso*	*Si*

de ítems y estar diseñado para este tipo de datos, genera reglas de asociación que tienen la misma desventaja relacionada con su aplicabilidad, que los conjuntos frecuentes de ítems.

2.2. Reglas de Asociación en colecciones de datos mezclados

El conjunto de datos $\Omega = \{O_1, \ldots, O_n\}$ es una colección de datos mezclados [Ruiz-Shulcloper, 2009], si cada objeto de Ω está descrito por un conjunto $R = \{r_1, \ldots, r_m\}$ de atributos numéricos y no numéricos. Cada objeto de Ω se representa por una tupla $(v_{r_1}, \ldots, v_{r_m})$ donde $v_{r_j} \in D_{r_j}$ es el valor asociado al atributo r_j $(1 \leq j \leq m)$ y D_{r_j} es el dominio del atributo r_j; mientras que $O[r]$ denota el valor del atributo r en el objeto O. Un ejemplo de este tipo de colecciones se muestra en la Tabla 2.1.

Para la minería de reglas de asociación en colecciones de datos mezclados se han desarrollado tres enfoques, inicialmente el enfoque basado en discretización dura y el enfoque basado en discretización difusa; y más recientemente el enfoque basado en semejanza entre subdescripciones. A continuación se presentan particularidades de cada uno de ellos.

2.2.1. Enfoque basado en discretización dura

Srikant y Agrawal reportan en [Srikant and Agrawal, 1996] el primer algoritmo para el minado de reglas de asociación en colecciones de datos cuyos objetos son descritos por atributos numéricos y no numéricos y las denominaron reglas de asociación cuantitativas.

La solución de estos autores consiste en realizar un particionamiento del dominio de los atributos numéricos y combinar intervalos adyacentes para disminuir la pérdida de información inherente al particionamiento. Luego, el problema de minado de reglas de asociación en datos mezclados es transformado al problema de minado de reglas de asociación en datos Booleanos, haciendo corresponder un atributo Booleano por cada valor de cada atributo no numérico, así como un atributo Booleano por cada intervalo de cada atributo numérico del problema con datos mezclados. Finalmente, una variación del algoritmo *Apriori* [Agrawal and Srikant, 1994] (Algoritmo 2.1) es usado para el minado de los conjuntos frecuentes de ítems, a partir de los cuales son generadas las reglas de asociación interesantes mediante el algoritmo *GenRules* [Agrawal and Srikant, 1994] (Algoritmo 2).

En este contexto, un ejemplo de regla de asociación obtenida a partir de la colección de datos mezclados que se muestra en la tabla 2.1 sería:

$$(Edad \in [20, 30] \wedge Casado = No) \rightarrow (Auto = Mediano)$$

con soporte 0,33 y confianza 0,5.

Formalmente este tipo de reglas se conceptualizan de la siguiente manera [4] [Srikant and Agrawal, 1996]:

Definición 2.6. Sea Ω una colección de datos mezclados, una regla de asociación es una implicación de la forma $X \rightarrow Y$, donde X y Y son conjuntos de pares (r, c_r) con $r \in R$ y $c_r \subseteq D_r$ (c_r es un subconjunto de valores numéricos o es un solo valor no numérico), tal que, cada atributo sólo aparece a lo más una vez ya sea en X o en Y.

Análogamente a las reglas de asociación en datos Booleanos:

Definición 2.7. Se dice que una regla de asociación $X \rightarrow Y$ es interesante en una colección de datos mezclados Ω, si y sólo si su soporte es mayor o igual que un umbral de mínimo soporte $minSupp$ y su confianza es mayor o igual que un umbral de mínima confianza $minConf$.

Definición 2.8. Sea X un conjunto de pares (r, c_r) con $r \in R$ y $c_r \subseteq D_r$ tal que, $\forall (r, c_r) \in X$ y $\forall (r', c_{r'}) \in X$, si $r = r'$ entonces $c_r = c_r'$. Se dice que un objeto $O \in \Omega$ soporta a X si $\forall (r, c_r) \in X$, $O[r] \in c_r$. El soporte de X para una colección de datos

[4] Para lograr uniformomidad con el lenguaje que se ha empleado en las secciones anteriores y que se emplea en el resto del capítulo se ha variado la notación original empleada por los autores

mezclados Ω es la fracción de objetos de Ω que soportan a X.

$$supp(X) = \frac{|\{O \in \Omega \mid \forall (r, c_r) \in X, O[r] \in c_r\}|}{|\Omega|}$$

Tanto el soporte como la confianza de una regla de asociación en este contexto, son calculados igual que el contexto de las reglas de asociación en datos Booleanos, pero empleando la definición actual de soporte.

Definición 2.9. El soporte de una regla de asociación $X \rightarrow Y$ en una colección de datos mezclados Ω, es la fracción de objetos de Ω que soportan a $X \cup Y$.

$$supp(X \rightarrow Y) = supp(X \cup Y)$$

Definición 2.10. La confianza de una regla $X \Rightarrow Y$ en una colección de datos mezclados Ω, es la fracción de objetos en Ω que soportando a X, también soportan a Y.

$$conf(X \rightarrow Y) = \frac{supp(X \cup Y)}{supp(X)}$$

A partir de Srikant y Agrawal [Agrawal and Srikant, 1994] varios autores [Fukuda et al., 1996; Miller and Yang, 2002; Zhang et al., 1997; Mata et al., 2002a,b; Born and Schmidt-Thieme, 2004; Takashi et al., 2005; Ke et al., 2006; Karel, 2006; Salleb-Aouissi et al., 2007] se han centrado en cómo discretizar los atributos cuantitativos y en cómo reducir el número de reglas interesantes generadas, siempre teniendo en cuenta la relación entre el grado de discretización, el número de reglas, el tiempo de ejecución, y la pérdida de información inherente a la discretización.

La aplicabilidad de este enfoque es mayor que la del enfoque tradicional de minado de patrones frecuentes y reglas de asociación, pues puede verse como una generalización del anterior. Sin embargo, discretizar por medio de conjuntos duros no resuelve el problema, pues algunas veces estas transformaciones son realizadas sin considerar la semántica de los datos y como consecuencia la naturaleza de los mismos es cambiada. Además, en la práctica hay atributos numéricos que no deben ser discretizados, por ejemplo, en geociencias la Anomalía de Bouguer y su gradiente [Gómez et al., 1994]. En estos ejemplos prácticos, los especialistas del área, consideran que dos valores son equivalentes si el valor absoluto de su diferencia es menor que un umbral diferente de cero. Por tanto, para todo particionamiento del dominio de dichos atributos que se realice, siempre existen dos valores equivalentes que quedan separados por la frontera que define a dicho particionamiento. Más aun, dos valores pueden estar muy cercanos y no pertenecer al mismo conjunto duro, mientras que dos valores pueden estar mas lejanos y pertenecer al mismo conjunto duro.

2.2.2. Enfoque basado en discretización difusa

Los conjuntos difusos son una alternativa para discretizar el dominio de los atributos numéricos y también el dominio de los atributos no numéricos.

Los conjuntos difusos [Zadeh, 1965] pueden ser vistos como una generalización de los conjuntos duros (clásicos). A diferencia de los conjuntos duros, a los cuales los elementos pertenecen o no pertenecen, los conjuntos difusos permiten una gradación de la pertenencia de los elementos.

La pertenencia de un elemento del conjunto universo a un conjunto duro se define mediante una función binaria. Si el valor de la función evaluada en el elemento es 1 (*verdadero*), entonces el elemento pertenece al conjunto; si el valor de la función evaluada en el elemento es 0 (*falso*), entonces el elemento no pertenece al conjunto.

La teoría de conjuntos difusos generaliza las funciones de pertenencia, ampliando la imagen de las mismas a un intervalo especificado, típicamente $[0, 1]$. En la medida en que el valor de la función evaluada en el elemento esté más cercano a 1, el elemento pertenecerá más al conjunto, mientras que, en la medida en que el valor de la función evaluada en el elemento esté más cercano a 0, el elemento pertenecerá menos al conjunto. Formalmente, sea U el conjunto universo, un conjunto difuso A se caracteriza por una función de pertenencia $\mu_A : U \to [0, 1]$, tal que, $\forall a \in U$, $\mu_A(a)$ representa el grado de pertenencia del elemento a al conjunto difuso A.

Lee y Kwang reportan en [Lee and Lee-Kwang, 1997] el primer algoritmo para el minado de reglas de asociación usando discretización difusa. Se supone que por cada atributo se tienen conjuntos difusos asociados. A partir ellos son obtenidas reglas de asociación como la siguiente:

$$(Hamburguesa, PrecioMedio) \to (Coka, PrecioBajo)$$

donde $PrecioMedio$ es uno de los tres conjuntos difusos asociados al precio de la *Hamburguesa* ($PrecioBajo$, $PrecioMedio$, $PrecioAlto$) y $PrecioBajo$ es uno de los tres conjuntos difusos asociados al precio de la $Coka$. Sin embargo, los autores usan un umbral de pertenencia para transformar los conjuntos difusos en conjuntos duros; luego el problema es transformado al problema de minado de reglas de asociación en datos Booleanos, haciendo corresponder un atributo Booleano por cada uno de los conjuntos duros asociados a cada atributo y finalmente es usando un algoritmo de minado de reglas de asociación Binarias.

En [Kuok et al., 1998] también se supone que por cada atributo se tienen conjuntos difusos asociados, pero a diferencia del trabajo anterior los conjuntos difusos no son transformados en conjuntos duros. Los autores definen formalmente Regla de Asociación usando los conjuntos difusos (Definición 2.11 abajo) y caracterizan cuando una regla de asociación es interesante o no a partir del factor de significancia (denotado como $significance$) y el factor de certeza (denotado como *certain*). Ambos factores están definidos respectivamente como una extensión del soporte y la confianza.

Definición 2.11. Sea Ω una colección de datos mezclados, una regla de asociación usando conjuntos difusos es una implicación de la forma $X \to Y$, donde $X = \{A_1, \ldots, A_k\}$ es un conjunto de conjuntos difusos, tal que, todo A_i está asociado con un atributo $r_i \in R$, y $\forall A_i, A_j \in X$, si $A_i \neq A_j$ entonces $r_i \neq r_j$; y $Y = \{B_1, \ldots, B_l\}$ es un conjunto de conjuntos difusos, tal que, todo B_i está asociado con un atributo $r_i \in R$, y $\forall B_i, B_j \in X$, si $B_i \neq B_j$ entonces $r_i \neq r_j$.

Se dice que una regla de asociación $X \to Y$, usando conjuntos difusos, es interesante en una colección de datos mezclados Ω, si y sólo si su factor de significancia es mayor o igual que un umbral de mínima significancia y su factor de certeza es mayor o igual que un umbral de mínima certeza.

Definición 2.12. Sea $X = \{A_1, \ldots, A_k\}$ un conjunto de conjuntos difusos, tal que, todo A_i está asociado con un atributo $r_i \in R$, y $\forall A_i, A_j \in X$, si $A_i \neq A_j$ entonces $r_i \neq r_j$. El factor de significancia de X en Ω se define como:

$$significance(X) = \frac{\sum_{O \in \Omega} T^*(X, O)}{|\Omega|}$$

donde $T^*(\{A_1, \ldots, A_k\}, O)$ es el grado de pertenencia de O a la intersección de los conjuntos difusos $A_1, \ldots, A_k$.

Definición 2.13. El factor de significancia de $X \to Y$ en Ω se define como:

$$significance(X \to Y) = significance(X \cup Y)$$

Definición 2.14. El factor de certeza de $X \to Y$ en Ω se define como:

$$certain(X \to Y) = \frac{significance(X \cup Y)}{significance(X)}$$

Esta misma definición de regla de asociación fue utilizada en [Gyenesei, 2000], pero se denominó *soporte difuso* al factor de significancia y *confianza difusa* al factor de certeza.

En [Papadimitriou and Mavroudi, 2005] se extiende el enfoque de minado de conjuntos frecuentes de ítems, basado en árboles, al contexto de los conjuntos difusos y se presenta una estructura denominada árbol de patrones frecuentes difusos que es tan eficiente como las estructuras para el caso de conjuntos frecuentes de ítems. En colecciones de datos densas la estructura es pequeña y se puede recorrer eficientemente, en colecciones de datos muy dispersas y grandes dicha estructura resulta muy grande debido a que compacta poco y recorrerla es muy costoso.

Un algoritmo de aprendizaje de reglas de asociación basado en programación lógica inductiva fue presentado en [Serrurier et al., 2007]. El algoritmo maximiza la confianza de las reglas.

Se han propuesto otros trabajos que proponen algoritmos para el minado de reglas de asociación usando taxonomías de conjuntos difusos [De-Graaf et al., 2001; Chen

and Wei, 2002; Tzung-Pei et al., 2003; Shitong et al., 2005; Farzanyar et al., 2006]. Las taxonomías de conjuntos difusos comúnmente son definidas mediante árboles, donde los nodos representan conjuntos difusos y las aristas que los unen representan las funciones de pertenencia entre ellos. Por su parte, los algoritmos de minado de reglas de asociación usan las taxonomías de conjuntos difusos para calcular el grado de pertenencia de los valores de los atributos a cada uno de los conjunto difusos. El grado de pertenencia de un valor de un atributo a un conjunto difuso es calculado mediante la composición de las funciones de pertenencia representadas por las aristas que se encuentran entre el nodo que representa al conjunto difuso y los nodos hojas.

El enfoque basado en conjuntos difusos, a diferencia de las fronteras duras utilizadas para definir los intervalos en el enfoque de discretización, permite definir, tanto fronteras duras como fronteras difusas en dependencia de las funciones de pertenencia a los conjuntos difusos que sean usadas. Esto posibilita modelar de mejor manera las relaciones entre los valores de los atributos, respecto al enfoque basado en discretización. Sin embargo, dos valores lejanos y no semejantes pueden pertenecer a un conjunto difuso con valores de pertenencia parecidos. Por ejemplo, si se asume que la estatura media ideal de una persona es $170cm$, entonces tanto el valor de estatura $140cm$ como el valor de estatura $200cm$ (que no son valores de estatura cercanos) podrían pertenecer al conjunto difuso $EstaturaMedia$ con valores de pertenencia muy parecidos. Más aun, los grados de pertenencia a un conjunto difuso de dos valores cercanos y semejantes pueden ser menos parecidos que los grados de pertenencia a un conjunto difuso de dos valores lejanos y no semejantes. Por ejemplo, $140cm$ y $145cm$ pueden considerarse valores cercanos de estatura, mientras que sus valores de pertenencia al conjunto difuso $EstaturaMedia$ pueden considerarse menos parecidos que los valores de pertenencia al mismo conjunto difuso de $140cm$ y $200cm$, los cuales no son valores de estatura cercanos.

2.2.3. Enfoque basado en semejanza entre subdescripciones

El concepto de semejanza es comúnmente usado como herramienta para la toma de decisiones en disciplinas como Medicina [Ortiz-Posadas et al., 1994], Geología [Gómez et al., 1994], Sociología [J. Ruiz-Shulcloper, 1981], etc. En estos contextos las descripciones de los objetos no tienen que ser idénticas para ser consideradas como semejantes. Por ejemplo, se podría considerar que dos objetos (partes de objetos) son semejantes, si ellos (sus partes) son semejantes en al menos 90 % de los atributos.

Los algoritmos de minado de reglas de asociación, que utilizan la igualdad como función de semejanza para el conteo de las ocurrencias de los objetos o partes de ellos, pierden información valiosa, y como consecuencia patrones frecuentes y reglas interesantes.

Por ejemplo, para la colección de datos mezclados mostrada en la tabla 2.1, si se considera la igualdad como función de semejanza como en el enfoque tradicional de minado de patrones frecuentes y reglas de asociación, el umbral de mínimo soporte $minSupp = 0{,}66$ y el umbral de mínima confianza $minConf = 0{,}9$, entonces se obtiene un solo patrón frecuente ($Casado = No$) y no se obtienen reglas interesantes. Sin embar-

Tabla 2.2: Ejemplo de generación de patrones frecuentes y reglas de asociación obtenidas a partir de la tabla 2.1, para $minSupp = 0{,}66$ y $minConf = 0{,}9$.

Patrones frecuentes	*Reglas de asociación interesantes*
$(Edad = 25)$	$(Edad = 25) \rightarrow (Auto = Mediano)$
$(Edad = 29)$	$(Edad = 29) \rightarrow (Auto = Mediano)$
$(Auto = Mediano)$	$(Edad = 25) \rightarrow (Casado = No)$
$(Auto = Grande)$	$(Casado = No) \rightarrow (Edad = 25)$
$(Casado = No)$	$(Auto = Mediano) \rightarrow (Casado = No)$
$(Edad = 25, Auto = Mediano)$	$(Edad = 25) \rightarrow (Auto = Mediano, Casado = No)$
$(Edad = 29, Auto = Mediano)$	$(Edad = 25, Auto = Mediano) \rightarrow (Casado = No)$
$(Edad = 25, Casado = No)$	$(Edad = 25, Casado = No) \rightarrow (Auto = Mediano)$
$(Auto = Mediano, Casado = No)$	$(Auto = Mediano, Casado = No) \rightarrow (Edad = 25)$
$(Edad = 25, Auto = Mediano, Casado = No)$	$(Casado = No) \rightarrow (Edad = 25, Auto = Mediano)$

go, si se considera como función de semejanza para el atributo *Edad* que dos personas son semejantes en términos de sus edades, si ellas son de la misma generación, lo cual podría ser equivalente a considerar que dos edades son semejantes si el módulo de sus diferencias es a lo sumo 5 años; como función de semejanza para el atributo *Auto* consideramos que los autos compactos y los autos medianos son semejantes, los autos medianos y los autos grandes son semejantes, y los autos grandes y los autos lujosos también son semejantes; además consideramos como función de semejanza para el atributo *Casado* la igualdad y como función de semejanza entre objetos o partes de ellos que dos objetos (partes de objetos) son semejantes, si ellos (sus partes) son semejantes en todos los atributos; se tienen los patrones frecuentes y reglas de asociación interesantes mostrados en la tabla 2.2. Como puede apreciarse, al usar las funciones de semejanza anteriores, se producen patrones frecuentes y reglas de asociación ocultas para los algoritmos que utilizan como función de semejanza la igualdad.

En [Dánger et al., 2004], se propone un algoritmo (*ObjectMiner*), que utiliza funciones de semejanza en el conteo de las ocurrencias de partes de objetos, extendiendo para ello, los conceptos de soporte, confianza, regla de asociación, y regla de asociación interesante (ver Definiciones 2.16, 2.21, 2.18 y 2.19 abajo). Otros conceptos utilizados fueron el de subdescripción y subdescripción frecuente (ver Definiciones 2.15 y 2.17 abajo). El algoritmo propuesto se centra en la primera fase de la minería de reglas de asociación (obtener los patrones frecuentes) y es efectivo para funciones de semejanza Booleana que cumplen que si dos objetos no son semejantes respecto a un conjunto de atributos S_1, entonces tampoco lo son respecto a cualquier conjunto S_2, tal que $S_1 \subseteq S_2$.

Definición 2.15. Sea Ω una colección de datos mezclados, la subdescripción de un objeto O para un subconjunto de atributos $S \subseteq R$ denotada $I_S(O)$, es la proyección de los valores de O en términos de los atributos en S. Adicionalmente se denota $O[r]$ a la proyección de los valores de O en término del atributo r ($r \in R$).

Nótese que $I_{\{r\}}(O) = O[r]$.

Definición 2.16. Sea $I_S(O)$ una subdescripción de un objeto $O \in \Omega$, el soporte de $I_S(O)$ en Ω se define como:

$$supp(I_S(O)) = \frac{|\{O' \in \Omega \mid sim(I_S(O), I_S(O')) = 1\}|}{|\Omega|}$$

donde *sim* es una función de semejanza Booleana entre subdescripciones de objetos, la cual está basada en criterios Booleanos de comparación $C_r : D_r \times D_r \to [0, 1]$ uno para cada atributo.

Un ejemplo de este tipo de funciones es:

$$sim(I_S(O), I_S(O')) = \begin{cases} 1 & \text{si } \forall r \in S, C_r(O[r], O'[r]) = 1 \\ 0 & \text{en otro caso} \end{cases} \tag{2.1}$$

donde C_r es el criterio de comparación del atributo r. Dos ejemplo de criterios de comparación son:

$$C_{Edad}(x, y) = \begin{cases} 1 & \text{si } |x - y| \leq 5 \\ 0 & \text{en otro caso} \end{cases} \tag{2.2}$$

$$C_{Auto}(x, y) = \begin{cases} 1 & \text{si } (x = y) \vee \\ & \quad (x = Compacto \wedge y = Mediano) \\ & \quad (x = Mediano \wedge (y = Compacto \vee y = Grande)) \\ & \quad (x = Grande \wedge (y = Mediano \vee y = Lujoso)) \\ & \quad (x = Lujoso \wedge y = Grande) \\ 0 & \text{en otro caso} \end{cases} \tag{2.3}$$

Definición 2.17. Una subdescripción $I_S(O)$ de un objeto $O \in \Omega$ es frecuente si y sólo si su soporte es mayor o igual que un umbral de mínimo soporte *minSupp*.

Definición 2.18. Sea Ω una colección de datos mezclados, una regla de asociación es una implicación de la forma $X \to Y$, donde $X = I_{S_1}(O)$ y $Y = I_{S_2}(O)$, tal que, $O \in \Omega$, $S_1 \subset R$, $S_2 \subset R$, $S_1 \neq \emptyset$, $S_2 \neq \emptyset$ y $S_1 \cap S_2 = \emptyset$.

Definición 2.19. Se dice que una regla de asociación $X \to Y$ es interesante en una colección de datos mezclados Ω, si y sólo si su soporte es mayor o igual que un umbral de mínimo soporte *minSupp* y su confianza es mayor o igual que un umbral de mínima confianza $minConf$.

Definición 2.20. El soporte de una regla de asociación $I_{S_1}(O) \Rightarrow I_{S_2}(O)$ en una colección de datos mezclados Ω, se define como:

$$supp(I_{S_1}(O) \to I_{S_2}(O)) = supp(I_{S_1 \cup S_2}(O))$$

Definición 2.21. La confianza de una regla $I_{S_1}(O) \rightarrow I_{S_2}(O)$ en una colección de datos mezclados Ω se define como:

$$conf(I_{S_1}(O) \rightarrow I_{S_2}(O)) = \frac{supp(I_{S_1 \cup S_2}(O))}{supp(I_{S_1}(O))}$$

El algoritmo *ObjectMiner* (Algoritmo 2.3), inspirado en el algoritmo *Apriori*, trabaja como sigue: Primero son determinadas todas las subdescripciones frecuentes para cada atributo (subdescripciones de tamaño 1). Después, en cada iteración k (inicialmente $k = 2$) son combinadas dos a dos las subdescripciones frecuentes de tamaño $k - 1$, para obtener subdescripciones de tamaño k candidatas a frecuentes. En este paso, por cada par de subdescripciones, los conjuntos de índices de los objetos de la colección que contienen subdescripciones semejantes a una u otra subdescripción del par, son intersectados para crear un conjunto con los índices de los objetos candidatos a ser semejantes a la subdescripción resultante de la combinación de ambas subdescripciones (subdescripción candidata a frecuente). A partir de este conjunto y usando la función de semejanza son determinadas cuáles subdescripciones son semejantes a las candidatas a frecuentes, así como si las subdescripciones son o no frecuentes. Este proceso se repite incrementando k, mientras hayan sido obtenidas al menos dos subdescripciones frecuentes de tamaño $k - 1$.

Una vez obtenidas todas las subdescripciones frecuentes, a partir de ellas son generadas las reglas de asociación interesantes.

Una debilidad de este algoritmo es que no permite funciones de semejanza no Booleanas, o que no cumplan que si dos objetos no son semejantes respecto a un conjunto de atributos S_1 entonces tampoco lo son respecto a cualquier conjunto S_2, tal que $S_1 \subset S_2$, lo cual restringe su ámbito de aplicación. Un ejemplo simple de función de semejanza que no cumple las condiciones planteadas es considerar que dos objetos (o subdescripciones de los mismos) son semejantes, si ellos (o sus subdescripciones) son semejantes en al menos 90 % de los atributos. Para esta función puede darse el caso de que una subdescripción de un objeto respecto al 10 % de sus atributos no sea semejante a la subdescripción de un segundo objeto respecto a ninguno de los atributos de dicho 10 %, y que la subdescripción del mismo objeto respecto al restante 90 % de sus atributos sea semejante a la subdescripción del segundo objeto respecto todos los atributos de dicho 90 %. Como consecuencia los objetos no son semejantes respecto al mencionado 10 % de sus atributos y sin embargo son semejantes al tener en cuenta el total de atributos.

No obstante a esta debilidad, en [Dánger et al., 2004] se muestra cómo, mediante la incorporación del concepto de semejanza entre subdescripciones de objetos en el conteo de ocurrencias, pueden ser descubiertas reglas de asociación ocultas para el enfoque tradicional de minado de reglas de asociación y los enfoques basados en discretización dura y difusa; abriéndose así un nuevo enfoque de minado de Reglas de Asociación en colecciones de datos mezclados, lo cual motivó la investigación desarrollada en esta tesis doctoral.

Procedimiento `ObjectMiner`(Ω, C, sim, $minSupp$)

Input: Ω - Colección de objetos, C - Criterios de comparación,
sim - función de semejanza, $minSupp$ - Umbral de mínimo soporte.
Output: F - Conjunto de subdescripciones frecuentes.

```
1  L_1 ← SETFREQVALUES(Ω, C)
2  F ← L_1
3  k ← 2
4  while L_{k-1} ≠ ∅ do
5      featurePairs ← {(S_i, S_j) | ∃I_{S_i}(O) ∈ L_{k-1}, ∃I_{S_j}(O') ∈ L_{k-1}, |S_i ∩ S_j| = k}
6      foreach pair of feature sets (S_i, S_j) ∈ featurePairs do
7          foreach object O ∈ Ω do
8              if I_{S_i}(O) ∈ L_{k-1} and I_{S_j}(O) ∈ L_{k-1} then
9                  indexCandidates ← I_{S_i}(O).indexes ∩ I_{S_j}(O).indexes
10             end
11             I_{S_i∪S_j}(O).indexes ← ∅
12             foreach index l ∈ indexCandidates do
13                 if sim(I_{S_i∪S_j}(O), I_{S_i∪S_j}(O_l)) = 1 then
14                     I_{S_i∪S_j}(O).indexes ← I_{S_i∪S_j}(O).indexes ∪ {l}
15                 end
16             end
17             if |I_{S_i∪S_j}(O).indexes| ≥ minSupp then
18                 L_k ← L_k ∪ I_{S_i∪S_j}(O)
19             end
20         end
21     end
22     F ← F ∪ L_k
23     k ← k + 1
24 end
```

Algoritmo 2.3: Algoritmo *ObjectMiner*.

Procedimiento `SETFREQVALUES`(Ω, C, $minSupp$)

Input: Ω - Colección de objetos, C - Criterios de comparación,
$minSupp$ - Umbral de mínimo soporte.
Output: F_1 - Conjuntos de subdescripciones frecuentes de tamaño 1.

```
1  F_1 ← ∅
2  foreach feature r ∈ R do
3  |  foreach feature value v_r ∈ D_r, such that, ∃O ∈ Ω, v_r = O[r] do
4  |  |  v_r.indexes ← ∅
5  |  |  foreach object O_i ∈ Ω do
6  |  |  |  if C_r(v_r, O_i[r]) = 1 then
7  |  |  |  |  v_r.indexes ← v_r.indexes ∪ {i}
8  |  |  |  end
9  |  |  end
10 |  |  if |v_r.indexes| ≥ minSupp then
11 |  |  |  F_1 ← F_1 ∪ {v_r}
12 |  |  end
13 |  end
14 end
```

2.3. Síntesis y Conclusiones

En este capítulo se ha presentado el trabajo relacionado con el minado de reglas de asociación. Se ha hecho énfasis en los algoritmos de minado de reglas de asociación Binarias, particularmente en los algoritmos dedicados al primer y más costoso paso de este proceso, los algoritmos de minado de conjuntos de ítems frecuentes. Para colecciones de datos mezclados, fueron descritos los algoritmos fundamentales de los enfoques basados tanto en discretización dura como difusa, así como los inconvenientes del uso de los mismos.

Finalmente, se presentó un ejemplo para ilustrar que en problemas donde el concepto de semejanza es usado para comparar objetos, al minar reglas de asociación usando la igualdad pueden perderse patrones frecuentes y reglas de asociación interesantes. Además, se describieron los avances reportados en la literatura sobre el uso de funciones de semejanza para el minado de reglas de asociación y las debilidades de los mismos que motivaron el trabajo desarrollado en esta tesis.

Capítulo 3

Minado de Patrones Frecuentes usando Funciones de Semejanza Booleana

En este capítulo extendemos los conceptos de frecuencia, patrón frecuente, confianza, regla de asociación, así como la propiedad de Clausura Descendente del soporte, considerando funciones de semejanza Booleana no necesariamente simétricas, entre subdescripciones de objetos. Además, proponemos una estructura de datos y 3 algoritmos de minado de patrones similares frecuentes. Finalmente se adapta un algoritmo de minado de reglas de asociación Binarias al minado de reglas de asociación incorporando el concepto de semejanza Booleana entre descripciones y subdescripciones de objetos con datos mezclados.

3.1. Conceptos básicos

Sea $\Omega = \{O_1, \ldots, O_n\}$ una colección de descripciones de objetos en términos de un conjunto de atributos numéricos y no numéricos $R = \{r_1, \ldots, r_m\}$. Cada objeto O de Ω se representa por una tupla $I_R(O) = (v_{r_1}, \ldots, v_{r_m})$ donde $v_{r_i} \in D_i$ $(1 \leq i \leq m)$ y D_i es el dominio del atributo r_i, que puede ser numérico o no numérico. Es decir, Ω es una colección de datos Mezclados.

Definición 3.1. Dados dos objetos O,O' de Ω y dos conjuntos de atributos $\check{S}$, $\hat{S}$ subconjuntos de R, tales que $\check{S}$ es subconjunto de $\hat{S}$, $I_{\check{S}}(O)$ e $I_{\hat{S}}(O')$ exactamente iguales respecto a $\check{S}$, entonces decimos que $I_{\check{S}}(O)$ es una subdescripción de $I_{\hat{S}}(O')$ e $I_{\hat{S}}(O')$ es una superdescripción de $I_{\check{S}}(O)$.

Por otro lado, sea una subdescripción $P = I_{\hat{S}}(O)$ y $\check{S} \subseteq \hat{S}$, usaremos $I_{\check{S}}(P)$ como notación equivalente a $I_{\check{S}}(O)$.

Dado un subconjunto de atributos $S \subseteq R$, $S \neq \emptyset$, una función de semejanza [Martínez-Trinidad et al., 2000] Booleana, f_S, es una función que recibe como argumento dos sub-

descripciones $I_S(O)$ e $I_S(O')$, tal que $O, O' \in \Omega$ y que tiene como imagen el conjunto $\{0, 1\}$. Para no sobre cargar la notación, se usará O y O' en lugar de $I_S(O)$ e $I_S(O')$ como argumentos de f_S. Este convenio se usará en contextos donde esté explícito el subconjunto de atributos S. $f_S(O, O') = 0$ significa que O' no es semejante a O respecto al conjunto de atributos S; y $f_S(O, O') = 1$ significa que O' es semejante a O respecto al conjunto de atributos S. Algunos ejemplos de funciones de semejanza Booleana son:

$$f_S(O, O') = \begin{cases} 1 & \text{si } \forall r \in S, C_r(O[r], O'[r]) = 1 \\ 0 & \text{en otro caso} \end{cases} \tag{3.1}$$

$$f_S(O, O') = \begin{cases} & \dfrac{\sum_{r \in S} C_r(O[r], O'[r])}{|S|} \\ 1 & \text{si } \dfrac{\sum_{r \in S} C_r(O[r], O'[r])}{|S|} \geq \alpha \\ 0 & \text{en otro caso} \end{cases} \tag{3.2}$$

donde $C_r : D_r \times D_r \to \{0, 1\}$ es un criterio de comparación entre dos valores del atributo r.

Decimos que dos subdescripciones son estrictamente semejantes si y sólo si son semejantes pero no son identicamente iguales. Formalmente, $I_S(O')$ es estrictamente semejante a $I_S(O)$ si y sólo si $f_S(O, O') = 1$ y $\exists r \in S\ O'[r] \neq O[r]$.

Definición 3.2. Sea el subconjunto de atributos $S \subseteq R$, $S \neq \emptyset$, $O \in \Omega$, y f_S una función de semejanza Booleana; definimos la frecuencia de una subdescripción $I_S(O)$ en Ω para f_S como:

$$f_S freq(O) = \frac{|\{O' \in \Omega \mid f_S(O, O') = 1\}|}{|\Omega|}$$

Nótese que si f_S fuera la igualdad, entonces la frecuencia de $I_S(O)$ en Ω para f_S sería coincidente con el concepto tradicional de frecuencia, es decir, la fracción de objetos en Ω que contienen a la subdescripción $I_S(O)$, pero en este caso la frecuencia es la fracción de objetos O' cuyas subdescripciones $I_S(O')$ son semejantes a la subdescripción $I_S(O)$.

Definición 3.3. Decimos que una subdescripción $I_S(O)$ es una subdescripción f_S-frecuente (patrón similar frecuente[1]) en Ω si $f_S freq(O) \geq minFreq$, donde f_S es una función de semejanza Booleana y $minFreq$ es un umbral de mínima frecuencia.

Nosotros definimos regla de asociación como en [Dánger et al., 2004] (Definición 2.18), una expresión de la forma $X \to Y$, donde $X = I_{S_1}(O)$ y $Y = I_{S_2}(O)$, tal que, $O \in \Omega$, $S_1, S_2 \subset R$, $S_1 \neq \emptyset$, $S_2 \neq \emptyset$ y $S_1 \cap S_2 = \emptyset$. Sin embargo definimos Regla de asociación interesante como sigue:

Definición 3.4. Una regla de asociación $X \to Y$, donde $X = I_{S_1}(O)$ y $Y = I_{S_2}(O)$ es interesante si su confianza es mayor o igual que el umbral de confianza mínimo $minConf$ y X, Y y $Z = I_{S_1 \cup S_2}(O)$ son patrones similares frecuentes.

[1]Se utilizará este término en lugar de subdescripción f_S-frecuente en contextos generales o en los que no exista ambigüedad respecto al conjunto de atributos S y la función de semejanza f_S

A diferencia de la Definición 2.19, nosotros exigimos que tanto el antecedente como el consecuente de la regla de asociación sean patrones similares frecuentes. Esta condición no se exige en la Definición 2.19 debido a que es una consecuencia de que el soporte de Z sea mayor que el umbral de mínima frecuencia y de que las funciones que se permiten en [Dánger et al., 2004] tienen que satisfacer que, si dos objetos no son semejantes respecto a un conjunto de atributos S_1 entonces tampoco lo son respecto a cualquier conjunto S_2, tal que $S_1 \subset S_2$.

En nuestro caso las funciones de semejanza no tienen por qué satisfacer las condiciones planteadas y puede suceder que la frecuencia de una subdescripción sea menor que la frecuencia de una superdescripción de la misma. Un ejemplo de ello es la función de semejanza que considera que dos objetos (o subdescripciones de los mismos) son semejantes si ellos (o sus subdescripciones) son semejantes en al menos 90 % de los atributos.

Definición 3.5. Sea f_S una función de semejanza Booleana; definimos la confianza de una regla de asociación $X \to Y$, donde $X = I_{S_1}(O)$ y $Y = I_{S_2}(O)$, en Ω para f_S como:

$$f_S conf(I_{S_1}(O) \to I_{S_2}(O)) = \frac{f_{S_1 \cup S_2} freq(O)}{f_{S_1} freq(O)}$$

3.2. Propiedades de poda

La propiedad de Clausura Descendente del soporte (Propiedad 2.1), usada en la generación de conjuntos frecuentes de ítems para podar el espacio de búsqueda, asegura que todos los subconjuntos de un conjunto frecuente de ítems también son conjuntos frecuentes de ítems, y que todos los superconjuntos de un conjunto no frecuente de ítems son también conjuntos no frecuentes de ítems. Una propiedad análoga en nuestro contexto puede expresarse como: toda subdescripción de una subdescripción f_S-frecuente es una subdescripción f_S-frecuente, y toda superdescripción de una subdescripción no f_S-frecuente es una subdescripción no f_S-frecuente. Nosotros llamamos f_S-Clausura Descendente (Propiedad 3.2) a esta nueva propiedad.

A continuación se introducen además de la propiedad de f_S-Clausura Descendente, otras propiedades y proposiciones que permiten la poda del espacio de búsqueda de patrones similares frecuentes.

Definición 3.6. Una función de semejanza Booleana f_S es monótona no creciente, si y sólo si $\forall S, \hat{S} \subseteq R;\ O, O' \in \Omega;\ \emptyset \neq S \subset \hat{S}\ [f_S(O, O') = 0] \Rightarrow [f_{\hat{S}}(O, O') = 0]$.

La función de semejanza (3.1) es una función de semejanza Booleana monótona no creciente, y la función de semejanza (3.2) con $\alpha = 0{,}6$ no lo es.

Proposición 3.1. Dada una colección de objetos Ω y una función de semejanza Booleana monótona no creciente f_S si $I_S(O).\mathcal{S} = \{I_S(O') \mid O' \in \Omega \wedge I_S(O') \neq I_S(O) \wedge f_S(O', O) =$

Tabla 3.1: Colección de datos para ejemplificar una función de semejanza Booleana que no cumple la propiedad de f_S-Clausura Descendente.

Ω	r_1	r_2	r_3
O_1	0	0	0
O_2	0	0	1
O_3	0	1	1
O_4	1	1	1

1} es el conjunto de las subdescripciones a las cuales $I_S(O)$ es estrictamente semejante; entonces toda superdescripción $I_{\hat{S}}(O)$ de $I_S(O)$ sólo puede ser semejante a una superdescripción $I_{\hat{S}}(O')$ de $I_S(O')$, tal que $I_S(O') \in (I_S(O).\mathcal{S} \cup \{I_S(O)\})$

Demostración. La demostración resulta inmediata por reducción al absurdo; asumamos que existe una superdescripción $I_{\hat{S}}(O)$ de $I_S(O)$ semejante a una superdescripción $I_{\hat{S}}(O')$ de $I_S(O')$, tal que $I_S(O') \notin (I_S(O).\mathcal{S} \cup \{I_S(O)\})$. Entonces $f_S(O', O) = 0$ y $f_{\hat{S}}(O', O) = 1$. Sin embargo, esto es una contradicción puesto que, como f_S es monótona no creciente, $\forall S, \hat{S} \subseteq R;\ O, O' \in \Omega;\ \emptyset \neq S \subset \hat{S},\ [f_S(O', O) = 0] \Rightarrow [f_{\hat{S}}(O', O) = 0]$. Por tanto $I_S(O') \in (I_S(O).\mathcal{S} \cup \{I_S(O)\})$. □

Propiedad 3.1 (Monotonía de la frecuencia)**.** Dada una colección de objetos Ω y una función de semejanza Booleana f_S; decimos que f_S satisface la propiedad de monotonía de la frecuencia, si y sólo si $\forall S, \hat{S} \subseteq R;\ O \in \Omega\ [\emptyset \neq S \subset \hat{S}] \Rightarrow [f_S freq(O) \geq f_{\hat{S}} freq(O)]$.

Propiedad 3.2 ($f_S - Clausura$ Descendente)**.** Dada una colección de objetos Ω y una función de semejanza Booleana f_S; decimos que f_S satisface la propiedad de $f_S - Clausura$ Descendente, si y sólo si $\forall S, \hat{S} \subseteq R;\ O \in \Omega;\ \emptyset \neq S \subset \hat{S}\ [f_S freq(O) < minFreq] \Rightarrow [f_{\hat{S}} freq(O) < minFreq]$.

Sin embargo, esta propiedad (Propiedad 3.2) a diferencia de la propiedad de clausura descendente para minado de conjuntos frecuentes de ítems (Propiedad 2.1), no siempre es verdadera. Un ejemplo de función de semejanza Booleana que no cumple la propiedad de f_S-Clausura Descendente es (3.2) con $\alpha = 0{,}6$ usando la igualdad como criterios de comparación de todos los atributos. Dada la colección de objetos Ω mostrada en la tabla 3.1, si tomamos esta función, y tomando $minFreq$ como 0,5, y considerando $S = \{r_1, r_2\} \subset \hat{S} = \{r_1, r_2, r_3\}$, entonces para el objeto descrito por $(0, 1, 1)$, $f_S freq(O) = 0{,}25 < minFreq$, y $f_{\hat{S}} freq(O) = 0{,}75 > minFreq$.

El cumplimiento de la propiedad de f_S-Clausura Descendente depende de la monotonía de la frecuencia, la cual a su vez depende de la monotonía de la función de semejanza. Estas dependencias serán demostradas a continuación.

Proposición 3.2. Dada una colección de objetos Ω y una función de semejanza Booleana f_S, si f_S es monótona no creciente, entonces $\forall S, \hat{S} \subseteq R;\ O \in \Omega\ [\emptyset \neq S \subset \hat{S}] \Rightarrow [f_S freq(O) \geq f_{\hat{S}} freq(O)]$.

Demostración. Si f_S es una función de semejanza Booleana monótona no creciente entonces $\forall S, \hat{S} \subseteq R$; $O, O^{'} \in \Omega$; $\emptyset \neq S \subset \hat{S}$ $[f_S(O, O^{'}) = 0] \Rightarrow [f_{\hat{S}}(O, O^{'}) = 0]$. Como $\big[[f_S(O, O^{'}) = 0] \Rightarrow [f_{\hat{S}}(O, O^{'}) = 0] \equiv [f_{\hat{S}}(O, O^{'}) = 1] \Rightarrow [f_S(O, O^{'}) = 1]\big]$; $\forall O^{'} \in \Omega$ si $O^{'} \in \{O^{''} \mid f_{\hat{S}}(O, O^{''}) = 1\}$ entonces $O^{'} \in \{O^{''} \mid f_S(O, O^{''}) = 1\}$. Consecuentemente $\{O^{'} \mid f_{\hat{S}}(O, O^{'}) = 1\} \subseteq \{O^{'} \mid f_S(O, O^{'}) = 1\}$. Por lo tanto, $\forall S, \hat{S} \subseteq R; O \in \Omega$ $[\emptyset \neq S \subset \hat{S}]$

$$\Rightarrow |\{O^{'} \in \Omega \mid f_S(O, O^{'}) = 1\}| \geq |\{O^{'} \in \Omega \mid f_{\hat{S}}(O, O^{'}) = 1\}|$$

$$\Rightarrow \frac{|\{O^{'} \in \Omega \mid f_S(O, O^{'}) = 1\}|}{|\Omega|} \geq \frac{|\{O^{'} \in \Omega \mid f_{\hat{S}}(O, O^{'}) = 1\}|}{|\Omega|}$$

$$\Rightarrow f_S freq(O) \geq f_{\hat{S}} freq(O)$$

□

Proposición 3.3. Dada una colección de objetos Ω y una función de semejanza Booleana f_S, si f_S satisface la monotonía de la frecuencia, entonces f_S satisface la propiedad de f_S-Clausura Descendente.

La demostración de esta proposición es inmediata.

Proposición 3.4. Dada una colección de objetos Ω y una función de semejanza Booleana f_S, si f_S es monótona no creciente, entonces f_S satisface la propiedad de f_S-Clausura Descendente.

A partir de las Proposiciones 3.2 y 3.3, la demostración de esta proposición es inmediata.

Un nuevo concepto relacionado con la poda de espacio de búsqueda, es el de Patrón f_S-interesante, el cual se introduce a continuación. Consideramos que una subdescripción es un patrón f_S-interesante si es un patrón similar frecuente o contribuye a la frecuencia de al menos un patrón similar frecuente (Definición 3.7).

Definición 3.7 (Patrón f_S-interesante)**.** Una subdescripción $I_S(O)$ es un patrón f_S-interesante si $f_S freq(O) \geq minFreq$ o $\exists O^{'} \in \Omega$; $I_S(O^{'}) \neq I_S(O)$ $[f_S freq(O^{'}) \geq minFreq] \Rightarrow [f_S(O^{'}, O) = 1]$.

En contraposición, un patrón no f_S-interesante, ni es similar frecuente, ni contribuye a la frecuencia de ningún patrón similar frecuente (Definición 3.8).

Definición 3.8 (Patrón no f_S-interesante)**.** Una subdescripción $I_S(O)$ es un patrón no f_S-interesante si $f_S freq(O) < minFreq$ y $\forall O^{'} \in \Omega$; $I_S(O^{'}) \neq I_S(O)$ $[f_S freq(O^{'}) \geq minFreq] \Rightarrow [f_S(O^{'}, O) = 0]$.

Proposición 3.5. Dada una colección de objetos Ω y una función de semejanza Booleana monótona no creciente f_S, si una subdescripción $I_S(O)$ es un patrón no f_S-interesante, entonces toda superdescripción $I_{\hat{S}}(O)$ de la misma, es también un patrón no f_S-interesante.

Demostración. Si $I_S(O)$ es un patrón no f_S-interesante, entonces

$$f_S freq(O) < minFreq \tag{3.3}$$

y

$$\forall O^{'} \in \Omega;\, I_S(O^{'}) \neq I_S(O)\ [f_S freq(O^{'}) \geq minFreq] \Rightarrow [f_S(O^{'}, O) = 0] \tag{3.4}$$

Como f_S es una función de semejanza Booleana monótona no creciente, entonces f_S satisface la propiedad de f_S-Clausura Descendente. Luego a partir de (3.3) tenemos:

$$\forall \hat{S} \subseteq R;\, \emptyset \neq S \subset \hat{S}\ [f_S freq(O) < minFreq] \Rightarrow [f_{\hat{S}} freq(O) < minFreq] \tag{3.5}$$

Por otro lado a partir de (3.4) y de la monotonía de f_S tenemos:

$$\begin{gathered} \forall \hat{S} \subseteq R;\, O^{'} \in \Omega;\, \emptyset \neq S \subset \hat{S};\, I_S(O^{'}) \neq I_S(O) \\ [f_{\hat{S}} freq(O) \geq minFreq] \Rightarrow [f_S freq(O) \geq minFreq] \\ \Rightarrow [f_S(O^{'}, O) = 0] \Rightarrow [f_{\hat{S}}(O^{'}, O) = 0] \end{gathered} \tag{3.6}$$

Adicionalmente es evidente que:

$$\forall \hat{S} \subseteq R;\, O^{'} \in \Omega;\, \emptyset \neq S \subset \hat{S}\ [I_S(O^{'}) \neq I_S(O)] \Rightarrow [I_{\hat{S}}(O^{'}) \neq I_{\hat{S}}(O)] \tag{3.7}$$

Por tanto, a partir de (3.5), (3.6) y (3.7) obtenemos que $\forall \hat{S} \subseteq R;\, \emptyset \neq S \subset \hat{S}$:

$$f_{\hat{S}} freq(O) < minFreq$$

y

$$\forall O^{'} \in \Omega;\, I_S(O^{'}) \neq I_{\hat{S}}(O)\ [f_{\hat{S}} freq(O^{'}) \geq minFreq] \Rightarrow [f_{\hat{S}}(O^{'}, O) = 0].$$

Es decir, toda superdescripción de un patrón no f_S-interesante es un patrón no f_S-interesante. □

Como ya se dijo anteriormente, los patrones no f_S-interesantes, ni son patrones similares frecuentes, ni contribuyen a la frecuencia de los patrones similares frecuentes. Además, si la función de semejanza es monótona no creciente, entonces a partir de la Proposición 3.5 y de la definición de patrón no f_S-interesante, las superdescripciones de los patrones no f_S-interesantes no son patrones similares frecuentes. Adicionalmente, estas subdescripciones tampoco contribuyen a la frecuencia de los patrones similares frecuentes. Como consecuencia, todos los patrones no f_S-interesantes pueden ser podados sin perder patrones similares frecuentes.

3.3. Estructura de Datos $STree$

Uno de los problemas asociados al minado de patrones similares frecuentes es la necesidad de evaluar la semejanza de cada subdescripción $I_S(O)$ con el resto de las subdescripciones, para calcular su frecuencia. Esto, en el peor de los casos es $|\Omega|^2 - |\Omega|$ evaluaciones de la función de semejanza por cada $S \subseteq R$. Sin embargo, suele suceder que para varios objetos $O, O' \in \Omega$, $I_S(O) = I_S(O')$. En estos casos no es necesario evaluar la semejanza entre repeticiones de una subdescripción. Por ejemplo, dada una colección de descripciones de 10000 personas, en las cuales está involucrado el atributo $Edad$ y asumiendo que sólo existen en la colección 100 valores diferentes del mismo, entonces para calcular la frecuencia de las subdescripciones a partir de $S = \{Edad\}$ no es necesario realizar $10000^2 - 10000 = 99990000$ evaluaciones, basta con $100^2 - 100 = 9900$, o incluso menos si algún valor de la edad no se presenta.

Sea $\mathcal{I}_S$ el conjunto de subdescripciones (no idénticas) de objetos respecto al conjunto de atributos S (en el ejemplo anterior $\mathcal{I}_{\{Edad\}} = \{(Edad = 1), \ldots, (Edad = 100)\}$, suponiendo que todos los valores desde 1 hasta 100 se presentan en la colección) y si $P.\mathcal{O}$ es el conjunto de objetos en Ω que contienen a una subdescripción P, entonces la frecuencia de una subdescripción $I_S(O)$ puede calcularse mediante:

$$f_S freq(O) = \frac{|I_S(O).\mathcal{O}| + \sum_{P \in \mathcal{I}_S | P \neq I_S(O)} f_S(O, P) * |P.\mathcal{O}|}{|\Omega|}$$

En esta expresión $|I_S(O).\mathcal{O}|$ representa el número de repeticiones de la subdescripción $I_S(O)$ en Ω y dado que las semejanzas entre las mismas e $I_S(O)$ son 1, no es necesario calcularlas. Por su parte, $\sum_{P \in \mathcal{I}_S | P \neq I_S(O)} f_S(O, P) * |P.\mathcal{O}|$ representa el número de subdescripciones semejantes a $I_S(O)$ que no son idénticas a $I_S(O)$ y particularmente $|P.\mathcal{O}|$ representa el número de repeticiones de P en Ω. En este caso dado que las semejanzas entre las repeticiones de P e $I_S(O)$ son todas iguales a $f_S(O, P)$, sólo es necesario calcular la semejanza de O con P y no con todas sus repeticiones. De esta forma se puede reducir el número de evaluaciones de la función de semejanza.

Por otro lado, sea f_S monótona no creciente, si para una subdescripción $I_S(O)$ se tiene el conjunto de subdescripciones a las cuales es semejante ($I_S(O).\mathcal{S}$), entonces una superdescripción de $I_S(O)$ sólo puede ser semejante a las superdescripciones de los elementos de $I_S(O).\mathcal{S} \cup \{I_S(O)\}$ (véase Proposición 3.1). Debido a que al calcular la frecuencia de una superdescripción $I_{\hat{S}}(O)$ de $I_S(O)$ ya se tiene calculado el conjunto $I_S(O).\mathcal{S}$, puede reducirse aún más el número de evaluaciones de f_S, pues para obtener el número de descripciones semejantes a $I_{\hat{S}}(O)$ que no son idénticas a $I_{\hat{S}}(O)$ no es necesario calcular la semejanza entre $I_{\hat{S}}(O)$ y todas las superdescripciones $I_{\hat{S}}(O') \in \mathcal{I}_{\hat{S}}$. Sólo es necesario calcular la semejanza entre $I_{\hat{S}}(O)$ y las superdescripciones $I_{\hat{S}}(O')$ que sean una superdescripción de alguna subdescripción $I_S(O')$ tal que $I_S(O) \in (I_S(O').\mathcal{S} \cup \{I_S(O')\})$.

Sea f_S monótona no creciente. Dado $I_S(O).\mathcal{S}$, la frecuencia de $I_{\hat{S}}(O)$ una superdescripción de $I_S(O)$, puede calcularse mediante:

$$f_{\hat{S}}freq(O) = \frac{|I_{\hat{S}}(O).\mathcal{O}| + \sum_{\hat{P}\in\mathcal{I}_{\hat{S}}|\hat{P}\neq I_{\hat{S}}(O),\hat{P}\supset P,P\in(I_S(O).\mathcal{S}\cup\{I_S(O)\})} f_{\hat{S}}(O,\hat{P}) * |\hat{P}.\mathcal{O}|}{|\Omega|}$$

donde $\hat{P} \supset P$ significa que $\hat{P}$ es una superdescripción de P.

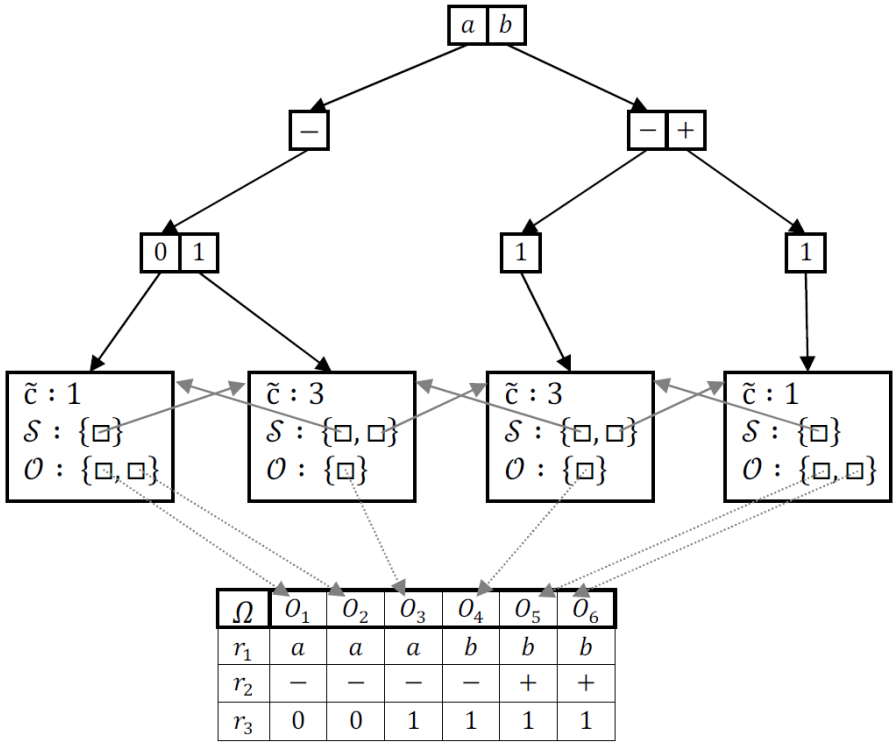
Basándonos en estas ideas, para facilitar el cálculo de la frecuencia y reducir el número de evaluaciones de la función de semejanza, proponemos una estructura de datos que denominamos $STree$. Un $STree_S$ es un árbol donde cada camino desde la raíz hasta una hoja representa una subdescripción P respecto al conjunto S. En cada hoja se almacena lo siguiente:

- $P.\tilde{c}$: Número de subdescripciones semejantes a P que no son idénticas a P.
- $P.\mathcal{O}$: Conjunto de objetos que contienen a la subdescripción P.
- $P.\mathcal{S}$: Conjunto de subdescripciones a las cuales P es semejante pero no idéntica.

En la figura 3.1 se muestra un ejemplo de estructura $STree_{\{r_1,r_2,r_3\}}$ para la colección $\Omega = \{O_1, O_2, O_3, O_4, O_5, O_6\}$, y la función de semejanza (3.2) con $\alpha = 0{,}6$ y usando la igualdad como criterio de comparación para cada atributo. De izquierda a derecha, los caminos desde la raíz hasta cada hoja representan respectivamente a las subdescripciones $(a,-,0)$, $(a,-,1)$, $(b,-,1)$ y $(b,+,1)$. Para la subdescripción $P = (a,-,0)$, $P.\mathcal{O} = \{O_1, O_2\}$ pues los objetos O_1 y O_2 contienen a P, $P.\mathcal{S} = \{(a,-,1)\}$ pues P sólo es semejante y no idéntica a $(a,-,1)$ y $P.\tilde{c} = 1$ pues la subdescripción $(a,-,1)$ es la única subdescripción que es semejante a P pero no es idéntica P.

Los algoritmos de minado de patrones similares frecuentes que serán propuestos en las siguientes secciones, construyen y utilizan convenientemente esta estructura. De forma general un $STree_S$ se construye en los siguientes cuatro pasos:

1. Construir el $STree_S$ vacío, el cual no contiene rama alguna.
2. Insertar las subdescripciones de los objetos que contendrá el $STree_S$. Al insertar la subdescripción de un objeto O, si el camino desde la raíz hasta una hoja que representa a la subdescripción $I_S(O)$ aun no existe, es decir $I_S(O)$ no está contenida en $STree_S$, entonces se crea esta rama. La inserción de la subdescripción del objeto O en el $STree_S$ concluye con la inserción de O en $I_S(O).\mathcal{O}$.
3. Calcular para cada subdescripción $I_S(O)$ contenida en el $STree_S$, el conjunto $I_S(O).\mathcal{S}$ a partir de las demás subdescripciones contenidas en dicha estructura.
4. Calcular para cada subdescripción $I_S(O)$ contenida en el $STree_S$, $I_S(O).\tilde{c}$ a partir de los conjuntos $\mathcal{S}$ de las demás subdescripciones contenidas en dicha estructura.



Ω	O_1	O_2	O_3	O_4	O_5	O_6
r_1	a	a	a	b	b	b
r_2	$-$	$-$	$-$	$-$	$+$	$+$
r_3	0	0	1	1	1	1

Figura 3.1: Ejemplo de estructura $STree_{\{r_1,r_2,r_3\}}$.

Una vez construida la estructura, la frecuencia de cada subdescripción $I_S(O)$ contenida en el $STree_S$ es calculada como:

$$f_S freq(O) = \frac{|I_S(O).\mathcal{O}| + I_S(O).\tilde{c}}{|\Omega|}$$

Después de este paso, dos de los algoritmos que serán propuestos en las siguientes secciones, eliminan las subdescripciones que son patrones no f_S-interesantes. Dado que las subdescripciones frecuentes son patrones f_S-interesantes, para realizar esta acción sólo se verifican si son o no f_S-interesantes las subdescripciones que no son similares frecuentes. Un patrón no similar frecuente P es no f_S-interesante si no existe un patrón similar frecuente $P^{'}$ en $P.\mathcal{S}$.

Eliminar del $STree_S$ una subdescripción P consiste en eliminar del árbol el camino que representa a P, en caso que no exista otra subdescripción en $STree_S$ que comparta parcialmente dicho camino. En otro caso, consiste en eliminar el camino parcial que P no comparte con ninguna otra subdescripción.

3.4. Algoritmos de minado de patrones similares frecuentes

El universo de funciones de semejanza Booleana puede ser dividido en dos conjuntos según las mismas sean o no monótonas no crecientes. En las siguientes secciones se proponen dos algoritmos de minado de patrones similares frecuentes para funciones de semejanza Booleana: *STreeDC-Miner* para el conjunto de funciones de semejanza Booleana monótonas no crecientes, lo cual implica que satisfacen la propiedad de f_S-Clausura Descendente y *STreeNDC-Miner* para el resto de las funciones. El primer algoritmo, poda el espacio de búsqueda de patrones similares frecuentes, mientras el segundo debido a que no exige el cumplimiento de alguna propiedad por parte de la función de semejanza Booleana, explora exhaustivamente dicho espacio.

Adicionalmente, proponemos otro algoritmo (*RP-Miner*) para funciones de semejanza Booleana, que no cumplen la propiedad de monotonía, basado en podas relajadas, y por tanto aunque puede no encontrar todos los patrones similares frecuentes es más eficiente que *STreeNDC-Miner*.

3.4.1. *STreeDC-Miner*

El algoritmo *STreeDC-Miner* está diseñado para funciones de semejanza Booleana monótonas no crecientes, y que por tanto satisfacen la propiedad de f_S-Clausura Descendente.

Las ideas sobre las cuales este algoritmo está basado son:

- Suponer que f_S es monótona no creciente. Como consecuencia:

- Los patrones no f_S-interesantes son podados. Para ello, el espacio de búsqueda es explorado a partir de los patrones f_S-interesantes descritos por un solo atributo y hacia los patrones f_S-interesantes con más atributos, por medio de sucesivas expansiones, en las cuales un atributo y un valor del mismo es agregado a los patrones f_S-interesantes. Para cada expansión de un patrón f_S-interesante se verifica si dicha expansión es un patrón similar frecuente.
- Sólo las superdescripciones de un patrón similar frecuente pueden ser patrones similares frecuentes. Por tanto, no es necesario calcular la frecuencia de las expansiones de patrones que son f_S-interesantes, pero que no son patrones similares frecuentes. Como consecuencia, tampoco es necesario calcular la semejanza entre las expansiones de los patrones que son f_S-interesantes, pero que no son patrones similares frecuentes.
- Si una subdescripción P no es semejante a otra subdescripción P' entonces no es necesario calcular la semejanza de una superdescripción $\hat{P}$ de P a otra superdescripción $\hat{P}'$ de P'.

- Considerar las subdescripciones que sean idénticas como una única subdescripción, lo cual reduce el número de evaluaciones de la función de semejanza.

Sea $\prec$ un orden lineal en R y f_S una función de semejanza monótona no creciente, entonces tenemos que:

- Un conjunto de atributos S es expandible, si y sólo si $S = \emptyset$ o existe al menos un patrón similar frecuente $I_S(O)$.
- Un conjunto de atributos $\hat{S}$ es una expansión directa de S, si y sólo si S es expandible, $\hat{S} = S \cup \{r\}$, $r \in R$ y $\forall r' \in S$, $r' \prec r$.
- Un conjunto de atributos $\hat{\hat{S}}$ es una expansión de S, si y sólo si $\hat{\hat{S}}$ es una expansión directa de S, o existe un conjunto de atributos $\hat{S}$, tal que $\hat{S}$ es una expansión directa de S y $\hat{\hat{S}}$ es una expansión de $\hat{S}$.

La cardinalidad del espacio de combinaciones de atributos $(2^{|R|} - 1)$ crece exponencialmente respecto al número de atributos $(|R|)$. Para podar este espacio, en la búsqueda de patrones similares frecuentes sólo son obtenidas las expansiones del conjunto vacío de atributos, y por cada expansión $\hat{\hat{S}}$ son buscados los patrones similares frecuentes. Nótese que, en este proceso, para expandir un conjunto de atributos S, éste debe ser expandible y para esto, es primero necesario que existan patrones similares frecuentes respecto a S.

Para facilitar la búsqueda de todos los patrones similares frecuentes respecto a cada expansión directa $\hat{S}$ de un conjunto de atributos S, nuestro algoritmo construye una estructura de datos $STree_{\hat{S}}$ para cada expansión directa $\hat{S}$.

Procedimiento `STreeDC-Miner`($STree_S$,$\hat{S}$,Ω, $f_{\hat{S}}$, $minFreq$)

Input: $STree_S$ - Estructura de Datos, $\hat{S}$ - Conjunto de atributos, Ω - Colección de datos, f_S - Función de semejanza Booleana, $minFreq$ - Umbral de mínimo soporte.
Output: F - Conjunto de patrones similares frecuentes.

```
1  if Ŝ ≠ ∅ then
2      STree_Ŝ ← emptySTree_Ŝ
3      if |Ŝ| = 1 then
4          foreach object O ∈ Ω do
5              if ¬STree_Ŝ.contain(I_Ŝ(O)) then
6                  STree_Ŝ.add(I_Ŝ(O))
7              STree_Ŝ.I_Ŝ(O).O ← STree_Ŝ.I_Ŝ(O).O ∪ {O}
8          foreach P, P' ∈ STree_Ŝ do
9              if f_Ŝ(P, P') = 1 then
10                 P'.S ← P'.S ∪ {P}
11     else
12         foreach P ∈ STree_S do
13             foreach object O ∈ P.O do
14                 if ¬STree_Ŝ.contain(I_Ŝ(O)) then
15                     STree_Ŝ.add(I_Ŝ(O))
16                 STree_Ŝ.I_Ŝ(O).O ← STree_Ŝ.I_Ŝ(O).O ∪ {O}
17         foreach P, P' ∈ STree_Ŝ such that I_S(P) ∈ F, I_S(P) ∈ I_S(P').S do
18             if f_Ŝ(P, P') = 1 then
19                 P'.S ← P'.S ∪ {P}
20     foreach P ∈ STree_Ŝ do
21         foreach P' ∈ P.S do
22             P'.c̃ ← P'.c̃ + |P.O|
23     F ← {P ∈ STree_Ŝ | P.c̃ + |P.O| ≥ minFreq}
24     STree_Ŝ.removeNonf_SInterestingPatterns()
25 if Ŝ = ∅ ∨ F ≠ ∅ then
26     foreach expansión directa Ŝ̂ of Ŝ do
27         F ← F∪STreeDC-Miner(STree_Ŝ,Ŝ̂,Ω, f_Ŝ, minFreq)
```

Algoritmo 3.1: Algoritmo *STreeDC-Miner*.

Al iniciar el algoritmo *STreeDC-Miner*(Algoritmo 3.1), el conjunto de patrones frecuentes F y el conjunto de atributos a expandir $\hat{S}$ son vacíos. Además, la estructura $STree_S$ es *null*.

En dependencia del conjunto de atributos a expandir, $STreeDC-Miner$ considera los siguientes casos:

- $\hat{S} = \emptyset$. El algoritmo se llama a sí mismo recursivamente para cada expansión directa de $\hat{S}$ (líneas 25-27).

- $|\hat{S}| = 1$. Todos los objetos de la colección son adicionados a $STree_{\hat{S}}$ (líneas 4-7). Después de esto, las semejanzas entre todas las subdescripciones contenidas en $STree_{\hat{S}}$ son calculadas, y para cada subdescripción $P^{'}$ en $STree_{\hat{S}}$, la lista de subdescripciones a las cuales $P^{'}$ es semejante es actualizada uniendo a ésta las subdescripciones P tales que $f_{\hat{S}}(P, P^{'}) = 1$ (líneas 8-10). Luego, para cada subdescripción P contenida en $STree_{\hat{S}}$, $P.\tilde{c}$ es calculado; los patrones similares frecuentes son obtenidos y el conjunto de patrones frecuentes es actualizado. Además, los patrones no f_S-interesantes son eliminados de $STree_{\hat{S}}$ (líneas $20-24$). Finalmente, si el conjunto de patrones similares frecuentes respecto a $\hat{S}$ no es vacío, el algoritmo se llama a sí mismo recursivamente para cada expansión directa de $\hat{S}$ (líneas 25-27).

- $|\hat{S}| > 1$. Para cada subdescripción P contenida en $STree_S$, los objetos contenidos en $P.\mathcal{O}$, son adicionados a $STree_{\hat{S}}$. Nótese que, llegado a este punto, los patrones no f_S-interesantes ya han sido eliminados de $STree_S$, y por tanto $STree_S$ sólo contiene patrones f_S-interesantes (líneas 12-16). Después de esto, sólo las semejanzas entre las subdescripciones P y $P^{'}$ contenidas en $STree_{\hat{S}}$, tales que $I_S(P)$ es un patrón similar frecuente y $f_S(P, P^{'}) = 1$, son calculadas. Además, la lista de subdescripciones a las cuales $P^{'}$ es semejante, es actualizada uniendo a ésta las subdescripciones P tales que $f_{\hat{S}}(P, P^{'}) = 1$ (líneas 17-19). Luego, para cada subdescripción P contenida en $STree_{\hat{S}}$, $P.\tilde{c}$ es calculado de igual forma que en caso anterior ($|\hat{S}| = 1$); los patrones similares frecuentes son obtenidos y el conjunto de patrones frecuentes es actualizado. Además, se eliminan de $STree_{\hat{S}}$ los patrones no f_S-interesantes (líneas 20-24). Finalmente, si el conjunto de patrones similares frecuentes respecto a $\hat{S}$ no es vacío, el algoritmo se llama a sí mismo recursivamente para cada expansión directa de $\hat{S}$ (líneas 25-27).

3.4.2. *STreeNDC-Miner*

El algoritmo *STreeNDC-Miner* está diseñado para funciones de semejanza Booleanas no monótonas no crecientes. Si la función de semejanza Booleana f_S no es monótona no creciente no se puede asegurar el cumplimiento de la propiedad de f_S-Clausura Descendente, ni que las superdescripciones de patrones no f_S-interesantes son patrones no f_S-interesantes. Como consecuencia, no es posible podar el espacio de búsqueda de patrones similares frecuentes sin que puedan perderse patrones similares frecuentes. Por

tanto, para garantizar que todos los patrones similares frecuentes puedan ser obtenidos es necesario buscarlos para todo $S \subseteq R$, $S \neq \emptyset$, lo cual implica una exploración exhaustiva del espacio de búsqueda.

Las ideas sobre las cuales el algoritmo *STreeNDC-Miner* está basado son:

- Considerar las subdescripciones que sean idénticas como una única subdescripción, lo cual reduce el número de evaluaciones de la función de semejanza.

- Usar una estrategia de recorrido ascendente. Así, el espacio de búsqueda es explorado desde las subdescripciones con $|R|$ atributos a las subdescripciones con un solo atributo, por medio de sucesivas reducciones de las subdescripciones, en las cuales un atributo y un valor del mismo es eliminado de ellas. Nótese que, dada una subdescripción P, el número de repeticiones de P es igual a la suma de las repeticiones de las superdescripciones de P que al eliminarles el mismo atributo se obtiene P. Por ejemplo, dada la colección de datos $\Omega = \{(a,-,0),(a,-,0),(a,-,1),(a,-,1),(a,+,0)\}$, el número de repeticiones de la subdescripción $P = (a,-)$ ($|P.\mathcal{O}| = 4$) es igual a la suma del número de repeticiones de la superdescripción $\hat{P'} = (a,-,0)$ de P ($|\hat{P'}.\mathcal{O}| = 2$), más el número de repeticiones de la superdescripción $\hat{P''} = (a,-,1)$ de P ($|\hat{P''}.\mathcal{O}| = 2$). Como consecuencia, en la estructura $STree$ usada para facilitar la búsqueda de todos los patrones similares frecuentes respecto a cada conjunto de atributos S, no es necesario almacenar para cada subdescripción P el conjunto de objetos $P.\mathcal{O}$, sino sólo $|P.\mathcal{O}|$. Por tanto, en la estructura $STree$ sustituimos $\mathcal{O}$ por $\bar{\bar{c}}$ ($\bar{\bar{c}} = |\mathcal{O}|$) para cada subdescripción.

Sea $\prec$ un orden lineal en R, entonces tenemos que:

- Un conjunto de atributos S es reducible, si y sólo si $|S| > 1$.

- Un conjunto de atributos $\check{S}$ es una reducción directa de S, si y sólo si S es reducible, $\check{S} = S - \{r\}$, $r \in S$ y $\forall r' \in (R - S)$, $r' \prec r$.

- Un conjunto de atributos $\check{\check{S}}$ es una reducción de S, si y sólo si $\check{\check{S}}$ es una reducción directa de S, o existe un conjunto de atributos $\check{S}$, tal que $\check{S}$ es una reducción directa de S y $\check{\check{S}}$ es una reducción de $\check{S}$.

Para descubrir los patrones similares frecuentes en una colección de datos Ω, $STreeNDC-Miner$ obtiene todas las reducciones de R, por medio de consecutivas reducciones directas. Para cada reducción $\check{S}$, se obtienen los patrones similares frecuentes.

Al iniciar el algoritmo, el conjunto de patrones frecuentes F es vacío y el conjunto de atributos a reducir $\check{S}$ es igual el conjunto que contiene a todos atributos ($\check{S} = R$). Además, la estructura $STree_S$ es *null*.

En dependencia del conjunto de atributos a reducir, *STreeNDC-Miner* considera los siguientes casos:

Procedimiento `STreeNDC-Miner`($STree_S$,$\check{S}$,Ω, $f_{\check{S}}$, $minFreq$)

Input: $STree_S$ - Estructura de Datos, $\check{S}$ - Conjunto de atributos, Ω - Colección de datos, f_S - Función de semejanza Booleana, $minFreq$ - Umbral de mínimo soporte.
Output: F - Conjunto de patrones similares frecuentes.

1 $STree_{\check{S}} \leftarrow emptySTree_{\check{S}}$
2 **if** $\check{S} = R$ **then**
3 **foreach** *object* $O \in \Omega$ **do**
4 **if** $\neg STree_{\check{S}}.contain(I_{\check{S}}(O))$ **then**
5 $STree_{\check{S}}.add(I_{\check{S}}(O))$
6 $STree_{\check{S}}.I_{\check{S}}(O).\bar{\bar{c}} \leftarrow STree_{\check{S}}.I_{\check{S}}(O).\bar{\bar{c}} + 1$
7 **else**
8 **foreach** $P \in STree_S$ **do**
9 **if** $\neg STree_{\check{S}}.contain(I_{\check{S}}(P))$ **then**
10 $STree_{\check{S}}.add(I_{\check{S}}(P))$
11 $STree_{\check{S}}.I_{\check{S}}(P).\bar{\bar{c}} \leftarrow STree_{\check{S}}.I_{\check{S}}(P).\bar{\bar{c}} + STree_S.I_S(P).\bar{\bar{c}}$
12 **foreach** $P, P^{'} \in STree_{\check{S}}$ *such that* $P \neq P^{'}$ **do**
13 **if** $f_{\check{S}}(P, P^{'}) = 1$ **then**
14 $P^{'}.\mathcal{S} \leftarrow P^{'}.\mathcal{S} \cup \{P\}$
15 **foreach** $P \in STree_{\check{S}}$ **do**
16 **foreach** $P^{'} \in P.\mathcal{S}$ **do**
17 $P^{'}.\tilde{c} \leftarrow P^{'}.\tilde{c} + P.\bar{\bar{c}}$
18 $F \leftarrow \{P \in STree_{\check{S}} \mid P.\tilde{c} + |P.\mathcal{O}| \geq minFreq\}$
19 **foreach** *reducción directa* $\check{\check{S}}$ *of* $\check{S}$ **do**
20 $F \leftarrow F \cup$ `STreeNDC-Miner`($STree_{\check{S}}$,$\check{\check{S}}$,Ω, $f_{\check{\check{S}}}$, $minFreq$)

Algoritmo 3.2: Algoritmo *STreeNDC-Miner*.

- $\check{S} = R$. Se adicionan todas las descripciones de los objetos de la colección a $STree_{\check{S}}$ (líneas 3-6).
- $\check{S} \neq R$. Se adicionan todas las subdescripciones respecto a $\check{S}$ de las subdescripciones contenidas en $STree_S$ ($\check{S}$ es una reducción directa de S) a $STree_{\check{S}}$ (líneas 8-11).

Después de esto, se calculan las semejanzas entre todas las subdescripciones contenidas en $STree_{\check{S}}$, y para cada subdescripción P' en $STree_{\check{S}}$, la lista de subdescripciones a las cuales P' es semejante se actualiza uniendo a ésta las subdescripciones P tales que $f_{\check{S}}(P, P') = 1$ (líneas 12-14). Luego, para cada subdescripción P contenida en $STree_{\check{S}}$, se calcula $P.\bar{\bar{c}}$; se obtienen los patrones similares frecuentes y se actualiza el conjunto de patrones similares frecuentes (líneas 15-18). Finalmente, el algoritmo se llama a sí mismo recursivamente para cada reducción directa de $\check{S}$ (líneas 19-20).

3.4.3. *RP-Miner*

Si la función de semejanza Booleana no es monótona no creciente, entonces al utilizar el algoritmo *STreeNDC-Miner*, se obtienen todos los patrones similares frecuentes, pero a costa de explorar exhaustivamente el espacio de búsqueda. Por otro lado, al utilizar el algoritmo *STreeDC-Miner*, el cual está diseñado para funciones de semejanza Booleana monótonas no crecientes, con funciones de semejanza Booleana que no son monótonas no crecientes, pueden perderse patrones similares frecuentes, debido a su mecanismo de poda.

El algoritmo *RP-Miner*, está diseñado para funciones de semejanza que no son monótonas no crecientes. En *RP-Miner* se relaja el mecanismo de poda de *STreeDC-Miner*. Como consecuencia, aunque pueden perderse patrones similares frecuentes, estos son generalmente muchos menos y nunca más que los que pierde *STreeDC-Miner*. Además, como resultado de la poda relajada, el tiempo empleado para explorar el espacio de búsqueda es menor que el empleado por *STreeNDC-Miner*.

En la figura 3.2 se muestra una colección de datos Ω, el espacio de búsqueda para dicha colección y la frecuencia de cada subdescripción usando la función de semejanza (3.2) con $\alpha = 0{,}5$ y como criterios de comparación la igualdad, la cual no es monótona no creciente. Considerando $minFreq = 0{,}8$, en la primera columna de cada nodo los patrones no similares frecuentes aparecen en celdas negras y los patrones similares frecuentes aparecen en celdas blancas. El color de la celda donde aparece la frecuencia de cada patrón similar frecuente $I_S(O)$ representa:

Blanco Toda subdescripción $I_{\check{S}}(O)$ de $I_S(O)$, es un patrón similar frecuente.

Gris claro No toda subdescripción $I_{\check{S}}(O)$ de $I_S(O)$, es un patrón similar frecuente, pero existe al menos una subdescripción $I_{\check{S}}(O)$ de $I_S(O)$, tal que $|\check{S}| = |S| - 1$, que si es un patrón similar frecuente.

Gris oscuro Ninguna subdescripción $I_{\check{S}}(O)$ de $I_S(O)$, es un patrón similar frecuente.

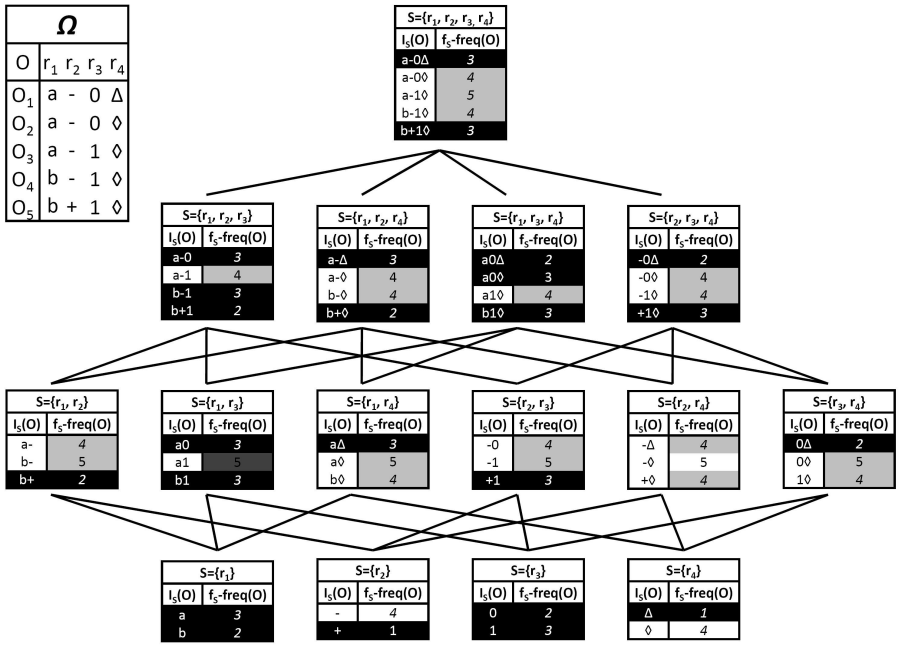


Figura 3.2: Espacio de búsqueda para la colección $\Omega = \{O_1, O_2, O_3, O_4, O_5\}$, y la función de semejanza (3.2) con $\alpha = 0{,}5$ y usado como criterios de comparación la igualdad.

En este ejemplo, existen 3 patrones similares frecuentes con celdas de frecuencia blancas, 19 patrones similares frecuentes con celdas de frecuencia grises claras, y sólo un patrón similar frecuente con celdas de frecuencia gris oscuro. El algoritmo *STreeDC-Miner* sólo descubriría los patrones similares frecuentes con celdas de frecuencia blancas (3), mientras el algoritmo *STreeNDC-Miner* encontraría todos los patrones similares frecuentes ($3 + 19 + 1 = 23$). *RP-Miner*, por su parte, descubriría los patrones similares frecuentes con celdas de frecuencia tanto blancas como grises claras ($3+19 = 22$). Nótese que, para cada patrón similar frecuente $I_S(O)$ con celda de frecuencia blanca o gris claro, existe al menos una secuencia de conjuntos de atributos $(S_1, S_2, \ldots, S_k)$, tal que $S_k = S$ y $\forall i < k$, $S_i \subset S_{i+1}$, $|S_{i+1}| = |S_i|+1$, $I_{S_i}(O)$ es un patrón similar frecuente con celda de frecuencia blanca o gris clara. Como consecuencia, cada patrón similar frecuente $I_S(O)$ con celda de frecuencia blanca o gris clara puede ser construido expandiendo sucesivamente algún patrón similar frecuente $I_S(O)$, por medio de la adición de un atributo y su valor. Por ejemplo, el patrón similar frecuente $(a, -, 0, \diamond)$ puede ser obtenido mediante la sucesiva adición de $r_1 = a$, $r_4 = \diamond$ y $r_3 = 0$ al patrón similar frecuente $(-)$. Puede apreciarse que los patrones intermedios $(a, -)$ y $(a, -, \diamond)$ son patrones similares frecuentes.

RP-Miner se basa en un proceso de expansión en el cual, a partir de los patrones similares frecuentes con un solo atributo, para cada S, $|S| = 1, 2, \ldots$, son obtenidos los patrones similares frecuentes respecto a cada $\hat{S}$, $\hat{S} = S \cup r$, $r \in R$. Obviamente, puede existir más de una secuencia de expansiones $(S_1, S_2, \ldots, S_k)$ para construir un patrón

$I_{S_k}(O)$. Por ejemplo, en la figura 3.2 el patrón $(a, -, 0, \diamond)$ puede ser obtenido por una expansión de los patrones $(a, 0, \diamond)$, $(-, 0, \diamond)$, $(a, -, \diamond)$ y $(a, -, 0)$. Sin embargo, en este proceso de expansión, si una subdescripción es generada por una secuencia y analizada (para verificar si es o no un patrón similar frecuente) y luego es generada nuevamente por otra secuencia, entonces la misma no es analizada nuevamente, ni expandida. Sólo los patrones expandidos que no han sido analizados previamente, son considerados como candidatos a patrones similares frecuentes. Adicionalmente, la semejanza entre dos patrones expandidos sólo es calculada si los patrones a partir de los cuales fueron generados son semejantes.

En la figura 3.3 se muestra la exploración del espacio de búsqueda mostrado en la figura 3.2, mediante el proceso de expansión de *RP-Miner*. Las flechas indican orden en el cual es explorado el espacio de búsqueda. Obsérvese que ningún patrón no similar frecuente es expandido. Sin embargo, cada expansión $I_{\hat{S}}(O)$ de un patrón no similar frecuente $I_S(O)$ puede ser obtenida por otra vía, si existe al menos un patrón similar frecuente $I_{S'}(O)$, tal que $S' \subset \hat{S}$, $|S'|+1 = |\hat{S}|$. Por ejemplo, el patrón no similar frecuente (a) no es expandido, pero su expansión $(a, -)$ es generada expandiendo el patrón similar frecuente $(-)$.

Siguiendo el proceso de expansión descrito anteriormente (a diferencia del algoritmo *STreeDC-Miner* que poda las expansiones de todos los patrones no similares frecuentes) sólo son podados los patrones no similares frecuentes $I_S(O)$ para los cuales no existe una secuencia de expansiones $(S_1, S_2, \ldots, S_k)$, tal que $S_k = S$ y $\forall i < k$, $S_i \subset S_{i+1}$, $|S_{i+1}| = |S_i| + 1$, $I_{S_i}(O)$ es un patrón similar frecuente. A esta poda la denominamos *Poda Relajada*.

Al iniciar el algoritmo *RP-Miner*, el conjunto de patrones analizados W (frecuentes o no, pero de tamaño mayor que 1), el conjunto de patrones frecuentes F y el conjunto de atributos a expandir $\hat{S}$ son vacíos. Además, la estructura $STree_S$ es *null*.

El comportamiento de *RP-Miner* al igual que *STreeDC-Miner* considera los siguientes casos: $\hat{S} = \emptyset$,$|\hat{S}| = 1$,$|\hat{S}| > 1$. En los tres casos, a diferencia de *STreeDC-Miner*, *RP-Miner* se llama a sí mismo recursivamente para cada expansión $\hat{S} \cup \{r\}$, $r \in (R - \hat{S})$ de $\hat{S}$ (líneas 26-28). Así, como se ha comentado anteriormente, una subdescripción respecto a k atributos puede ser obtenida mediante la expansión de cada una de sus subdescripciones respecto a $k - 1$ atributos.

Otra diferencia es que, en el caso en el que la cardinalidad del conjunto de atributos a expandir sea mayor que 1 ($|\hat{S}| > 1$), para cada subdescripción P contenida en $STree_S$, sólo los objetos contenidos en $P.\mathcal{O}$ cuya subdescripción respecto a S no ha sido analizada, son insertados en la estructura de datos $STree_{\hat{S}}$ y sus subdescripciones respecto a S son adicionadas al conjunto de patrones analizados W. De esta forma, una subdescripción sólo es analizada, o expandida en caso de ser f_S-frecuente, una vez.

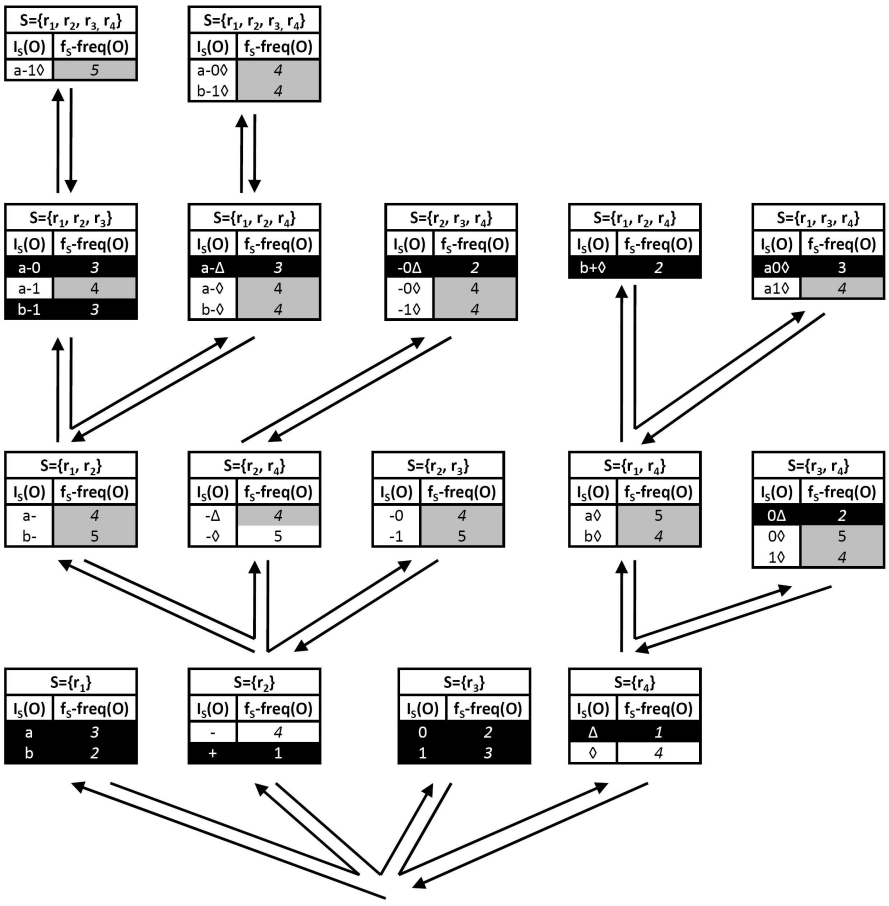


Figura 3.3: Exploración del espacio de búsqueda mostrado en la figura 3.2 mediante el proceso de expansión de *RP-Miner*.

Procedimiento `RP-Miner`($STree_S$,$\hat{S}$,Ω, $f_{\hat{S}}$, $minFreq$)

Input: $STree_S$ - Estructura de Datos, $\hat{S}$ - Conjunto de atributos, Ω - Colección de datos, f_S - Función de semejanza Booleana, $minFreq$ - Umbral de mínimo soporte.
Output: F - Conjunto de patrones similares frecuentes.

```
1  if Ŝ ≠ ∅ then
2      STree_Ŝ ← emptySTree_Ŝ
3      if |Ŝ| = 1 then
4          foreach object O ∈ Ω do
5              if ¬STree_Ŝ.contain(I_Ŝ(O)) then
6                  STree_Ŝ.add(I_Ŝ(O))
7              STree_Ŝ.I_Ŝ(O).O ← STree_Ŝ.I_Ŝ(O).O ∪ {O}
8          foreach P, P' ∈ STree_Ŝ do
9              if f_Ŝ(P, P') = 1 then
10                 P'.S ← P'.S ∪ {P}
11     else
12         foreach P ∈ STree_S do
13             foreach object O ∈ P.O do
14                 if I_Ŝ(O) ∉ W then
15                     if ¬STree_Ŝ.contain(I_Ŝ(O)) then
16                         STree_Ŝ.add(I_Ŝ(O))
17                     STree_Ŝ.I_Ŝ(O).O ← STree_Ŝ.I_Ŝ(O).O ∪ {O}
18                     W ← W ∪ I_Ŝ(O)
19         foreach P, P' ∈ STree_Ŝ such that I_S(P) ∈ F, I_S(P) ∈ I_S(P').S do
20             if f_Ŝ(P, P') = 1 then
21                 P'.S ← P'.S ∪ {P}
22     foreach P ∈ STree_Ŝ do
23         foreach P' ∈ P.S do
24             P'.c̃ ← P'.c̃ + |P.O|
25     F ← {P ∈ STree_Ŝ | P.c̃ + |P.O| ≥ minFreq}
26     STree_Ŝ.removeNonf_SInterestingPatterns()
27 if Ŝ = ∅ ∨ F ≠ ∅ then
28     foreach r ∈ (R − Ŝ) do
29         F ← F∪STreeRP-Miner(STree_Ŝ,Ŝ ∪ {r},Ω, f_Ŝ, minFreq)
```

Algoritmo 3.3: Algoritmo *RP-Miner*.

3.5. Algoritmo de Minado de Reglas de Asociación

Para generar reglas de asociación interesantes a partir de patrones similares frecuentes usando funciones de semejanza Booleana, proponemos la siguiente adaptación del algoritmo propuesto en [Agrawal and Srikant, 1994] para minar reglas de asociación en colecciones de datos binarios. Nuestra adaptación (Algoritmo 3.4) consiste en generar para cada patrón similar frecuente todas las reglas de asociación interesantes mediante la separación del mismo en dos patrones similares frecuentes (patrón antecedente y patrón consecuente) tales que los conjuntos que los describen sean disjuntos y que la confianza de la regla resultante sea mayor o igual que un umbral de mínima confianza ($minConf$).

Al iniciar el algoritmo *FSP-GenRules*, el conjunto F contiene los patrones similares frecuentes descubiertos por alguno de los algoritmos propuestos en las secciones anteriores y el conjunto de reglas de asociación generadas RA es vacío.

Procedimiento `FSP-GenRules(`F, f, $minConf$`)`

Input: F - Conjunto de patrones similares frecuentes, f - Función de semejanza Booleana, $minConf$ - Umbral de mínima confianza.
Output: RA - Conjunto de reglas de asociación interesantes.

```
1 foreach frequent similar pattern I_S(O) ∈ F do
2     foreach Š ⊂ S such that Š ≠ ∅, I_Š(O) ∈ F, I_{S−Š}(O) ∈ F do
3         if f_S conf(I_Š(O) → I_{S−Š}(O)) ≥ minConf then
4             RA ← RA ∪ {I_Š(O) → I_{S−Š}(O)}
```

Algoritmo 3.4: Algoritmo *FSP-GenRules*.

Si el conjunto F contuviera los patrones frecuentes descubiertos por los algoritmos del enfoque tradicional de minado de patrones frecuentes (los cuales usan la igualdad como función de semejanza), entonces pueden perderse reglas de asociación interesantes y más aun pueden generase reglas de asociación falsas, es decir, reglas que no son interesantes al usar una función de semejanza diferente de la igualdad. A continuación, se demuestra que al minar reglas de asociación usando como función de semejanza la igualdad, pueden perderse reglas de asociación interesantes y generarse falsas reglas de asociación interesantes.

Demostración. Sea f_S una función de semejanza diferente de la igualdad y $\bar{\bar{f}}_S$ la función de semejanza igualdad, entonces:

$\forall O, S;\ O \in \Omega;\ S \subseteq R$

$$\{O' \in \Omega | \{\bar{\bar{f}}_S(O, O') = 1\} \subseteq \{O' \in \Omega | f_S(O, O') = 1\} \tag{3.8}$$

Nótese, que al usar como función de semejanza la igualdad, cada subdescripción $I_S(O)$ sólo es semejante a las subdescripciones que son idénticamente iguales a ella; mientras que al usar una función de semejanza diferente de la igualdad, cada subdescripción $I_S(O)$

no sólo es semejante a las subdescripciones que son idénticamente iguales a ella, sino también a otras subdescripciones.

Por tanto $\forall O, S;\ O \in \Omega;\ S \subseteq R$

$$\bar{\bar{f}}_S freq(O) = \frac{|\{O' \in \Omega | \{\bar{\bar{f}}_S(O, O') = 1\}|}{|\Omega|} \leq \frac{|\{O' \in \Omega | f_S(O, O') = 1\}|}{|\Omega|} = f_S freq(O) \tag{3.9}$$

En este punto, para cada O, S, S', $O \in \Omega$, $S, S' \subseteq R$ pueden darse, entre otros, los siguientes casos:

1. $\bar{\bar{f}}_S freq(O) < minFreq \leq f_S freq(O)$ o $\bar{\bar{f}}_{S'} freq(O) < minFreq \leq f_{S'} freq(O)$. Para que una regla de asociación sea interesante tanto la subdescripción antecedente como la subdescripción consecuente deben ser patrones similares frecuentes. Como consecuencia, las reglas interesantes que contengan en el antecedente a la subdescripción $I_S(O)$ o en el consecuente a la subdescripción $I_{S'}(O)$ no serán generadas al emplear como función de semejanza la igualdad.

2. $minFreq \leq \bar{\bar{f}}_S freq(O) < f_S freq(O)$ y $minFreq \leq \bar{\bar{f}}_{S'} freq(O) < f_{S'} freq(O)$. En este caso ambas subdescripciones son similares frecuentes y por tanto la regla $I_S(O) \rightarrow I_{S'}(O)$ es candidata a ser una regla interesante y como consecuencia es el umbral de mínima confianza $minConf$ el que define si finalmente la regla es o no interesante.

   Si denominamos $\bar{\bar{conf}}$ a la confianza de la regla para $\bar{\bar{f}}_{S'}$, a partir de la definición de confianza se tiene,

$$\bar{\bar{conf}}(I_S(O) \rightarrow I_{S'}(O)) = \frac{\bar{\bar{f}}_{\{S \cup S'\}} freq(O)}{\bar{\bar{f}}_S freq(O)} \tag{3.10}$$

   y para f_S

$$conf(I_S(O) \rightarrow I_{S'}(O)) = \frac{f_{\{S \cup S'\}} freq(O)}{f_S freq(O)} \tag{3.11}$$

   .

   Debido a (3.9), los numeradores de $\bar{\bar{conf}}$ y $conf$ están relacionados por la desigualdad $\bar{\bar{f}}_{\{S \cup S'\}} freq(O) \leq f_{\{S \cup S'\}} freq(O)$ y los denominadores por la desigualdad $\bar{\bar{f}}_S freq(O) \leq f_S freq(O)$. Sin embargo no existe relación de orden entre $\bar{\bar{conf}}$ y $conf$ y por tanto pueden darse, entre otros, los siguientes casos:

   - $\bar{\bar{conf}}(I_S(O) \rightarrow I_{S'}(O)) < minConf \leq conf(I_S(O) \rightarrow I_{S'}(O))$. En este caso la regla interesante $I_S(O) \rightarrow I_{S'}(O)$ no es generada.
   - $conf(I_S(O) \rightarrow I_{S'}(O)) < minConf \leq \bar{\bar{conf}}(I_S(O) \rightarrow I_{S'}(O))$. En este caso es generada una falsa regla de asociación interesante $I_S(O) \rightarrow I_{S'}(O)$.

□

3.6. Síntesis y Conclusiones

En este capítulo se han extendido los conceptos de frecuencia, patrón frecuente, confianza y regla de asociación, considerando funciones de semejanza Booleana.

Además, fueron introducidas propiedades y proposiciones que permiten podar el espacio de búsqueda de patrones similares frecuentes cuando las funciones de semejanza Booleanas son monótonas no crecientes y fue propuesto el algoritmo de minado de patrones similares frecuentes *STreeDC-Miner* basado en las mismas.

Para el caso de funciones de semejanza que no cumplen con la propiedad de f_S-Clausura Descendente, fueron presentados dos algoritmos:

- *STreeNDC-Miner*. Realiza un exploración exhaustiva del espacio de búsqueda. No pierde patrones similares frecuentes cuando las funciones de semejanza Booleanas no son monótonas no crecientes. Es una solución factible para minar patrones frecuentes en colecciones de objetos descritos por un número pequeño de atributos.

- *RP-Miner*. Relaja el mecanismo de poda de *STreeDC-Miner*. Cuando las funciones de semejanza Booleanas no son monótonas no crecientes, aunque puede perder patrones similares frecuentes, estos son menos que los que pierde *STreeDC-Miner*.

Adicionalmente, se propuso una estructura de datos denominada $STree$, que es construida y utilizada por los 3 algoritmos, para facilitar la búsqueda de los patrones similares frecuentes.

Finalmente, fue adaptado el algoritmo de minado de reglas de asociación Binarias *GenRules*, para el minado de reglas de asociación incorporando el concepto de semejanza Booleana entre descripciones y subdescripciones de objetos con datos mezclados, obteniendo el algoritmo *FSP-GenRules*.

Con estos resultados se cumplen los objetivos particulares 1, 2, 3, 4 y 5 de esta investigación, para funciones de semejanza Booleana.

Capítulo 4

Minado de Patrones Frecuentes usando Funciones de Semejanza no Booleana

Existen problemas en los cuales los objetos de estudio son comparados usando funciones de semejanza no Booleana. En estos casos hay dos opciones para minar patrones similares frecuentes: transformar las funciones de semejanza no Booleana en funciones de semejanza Booleana; o desarrollar algoritmos para minar patrones similares frecuentes usando este tipo de funciones.

Podría pensarse que transformar las funciones de semejanza no Booleana en funciones de semejanza Booleana es una buena opción. Sin embargo, como explicaremos en la proxima sección tras mostrar algunos conceptos básicos, dicha transformación conlleva la pérdida de patrones similares frecuentes y la generación de falsos patrones similares frecuentes.

Por otro lado, al desarrollar algoritmos para minar patrones similares frecuentes que usen las funciones de semejanza no Booleanas, como también explicaremos en la próxima sección, puede suceder que una subdescripción se parezca poco al resto de las subdescripciones, pero aun así, acumule suficiente frecuencia para ser considerada un patrón similar frecuente; a este problema lo denominamos *problema de las bajas semejanzas y los muchos patrones semejantes*.

En este capítulo extendemos los conceptos de frecuencia, patrón frecuente, confianza, regla de asociación y la propiedad de Clausura Descendente del soporte, considerando funciones de semejanza no Booleana entre subdescripciones de objetos. Abordamos el *problema de las bajas semejanzas y los muchos patrones semejantes* mediante la introducción de un umbral de semejanza, y proponemos una estrategia para estimar este umbral. Además proponemos 3 algoritmos de minado de patrones similares frecuentes, considerando funciones de semejanza no Booleana. Finalmente, se muestra cómo adaptar el algoritmo de minado de reglas de asociación *FSP-GenRules* para permitir usar funciones de semejanza no Booleana.

4.1. Conceptos básicos

Sea $\Omega = \{O_1, \ldots, O_n\}$ una colección de descripciones de objetos en términos de un conjunto de atributos numéricos y no numéricos $R = \{r_1, \ldots, r_m\}$. Cada objeto O de Ω se representa por una tupla $I_R(O) = (v_{r_1}, \ldots, v_{r_m})$ donde $v_1 \in D_i$ $(1 \leq i \leq m)$ y D_i es el dominio del atributo r_i. Es decir Ω es una colección de datos Mezclados.

Cada subconjunto de atributos $S \subseteq R$ y $S \neq \emptyset$ tiene asociado una función de semejanza [Martínez-Trinidad et al., 2000] no Booleana f_S, entre subdescripciones de objetos de Ω, con imagen en $[0, 1]$. Algunos ejemplos de funciones de semejanza no Booleana son:

$$f_S(O, O') = \prod_{r \in S} C_r(O[r], O^{'}[r]) \tag{4.1}$$

$$f_S(O, O') = \frac{\sum_{r \in S} C_r(O[r], O^{'}[r])}{|S|} \tag{4.2}$$

donde $C_r : D_r \times D_r \to [0, 1]$ es un criterio de comparación entre valores del atributo r.

En el capítulo anterior se definió la frecuencia de una subdescripción $I_S(O)$ en Ω, $S \subseteq R$, para una función de semejanza Booleana f_S como:

$$f_S freq(O) \frac{|\{O' \in \Omega \mid f_S(O, O') = 1\}|}{|\Omega|}$$

Como consecuencia, para una función de semejanza Booleana cada subdescripción $I_S(O^{'})$ contribuye a la frecuencia de otra subdescripción $I_S(O)$, en dependencia de si $f_S(O, O^{'}) = 0$ ($I_S(O^{'})$ no es semejante a $I_S(O)$) o $f_S(O, O^{'}) = 1$ ($I_S(O^{'})$ es semejante a $I_S(O)$).

Siendo consecuentes, cuando la función de semejanza es no Booleana, cada subdescripción $I_S(O^{'})$ debería contribuir a la frecuencia de otra subdescripción $I_S(O)$, en dependencia del valor de $f_S(O, O^{'})$. Basándose en esto, la frecuencia se definiría como:

$$f_S freq(O) = \frac{\sum_{O' \in \Omega} f_S(O, O^{'})}{|\Omega|}$$

A partir de esta definición de frecuencia, a continuación se muestra que transformar las funciones de semejanza no Booleana en funciones de semejanza Booleana conlleva a la pérdida de patrones similares frecuentes y la generación de falsos patrones similares frecuentes.

Sea f_S una función de semejanza no Booleana, con valores en $[0, 1]$. La transformación de f_S en una función de semejanza Booleana $f_S^{'}$ requiere de un umbral α tal que si $f_S(P, P^{'}) \geq \alpha$ entonces $f_S^{'}(P, P^{'}) = 1$, en caso contrario $f_S^{'}(P, P^{'}) = 0$. Como consecuen-

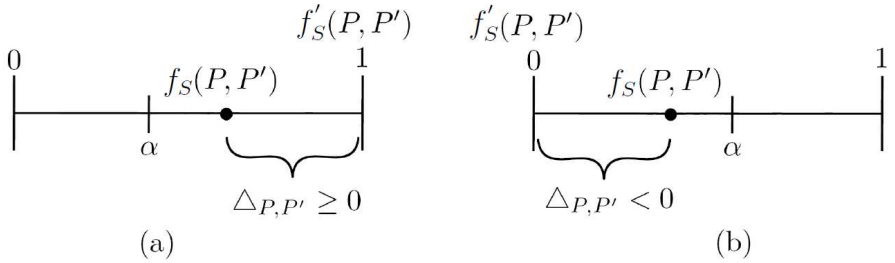


Figura 4.1: Ejemplo de transformación de una función de semejanza no Booleana f_S en una función de semejanza Booleana $f_S^{'}$ mediante un umbral α.

cia, sea $\Delta_{P,P'} = f'_S(P,P') - f_S(P,P')$, si $f_S(P,P') \geq \alpha$ entonces $\Delta_{P,P'} \geq 0$ (figura 4.1 (a)), en otro caso $\Delta_{P,P'} < 0$ (figura 4.1 (b)).

Por tanto, sea Ω una colección de objetos, P la descripción de un objeto respecto a un conjunto de atributos S, $\mathcal{P}$ el conjunto de las subdescripciones de todos los objetos en Ω respecto a S, $f_S freq$ la frecuencia de P usando f_S, $f_S freq^{'}$ la frecuencia de P usando $f_S^{'}$ y $minFreq$ el umbral de frecuencia mínima; entonces al calcular la frecuencia de la subdescripción P usando la función de semejanza Booleana $f_S^{'}$, la suma de las diferencias $\frac{\Delta_{P,P'}}{|\Omega|}$ para cada $P^{'} \in \mathcal{P}$ debidas a la Booleanización de la función de semejanza no Booleana f_S sería $\sum_{P' \in \mathcal{P}} \frac{\Delta_{P,P'}}{|\Omega|} = f_S freq^{'} - f_S freq$. Esta suma de diferencias puede ser positiva o negativa. En caso de ser positiva ($\sum_{P' \in \mathcal{P}} \frac{\Delta_{P,P'}}{|\Omega|} > 0$), entonces $f_S freq < f_S freq^{'}$. Adicionalmente, si $f_S freq < minFreq \leq f_S freq^{'}$ entonces P es un falso patrón similar frecuente, porque aunque la frecuencia ($f_S freq^{'}$) usando la función de semejanza Booleana ($f_S^{'}$) es mayor o igual que el umbral de frecuencia mínima $minFreq$, la frecuencia ($f_S freq$) usando la función de semejanza original no Booleana (f_S) no satisface dicho umbral. En caso de ser negativa la suma de diferencias ($\sum_{P' \in \mathcal{P}} \frac{\Delta_{P,P'}}{|\Omega|} < 0$), entonces $f_S freq^{'} < f_S freq$. Adicionalmente, si $f_S freq^{'} < minFreq \leq f_S freq$ entonces P es un patrón similar frecuente que está siendo considerado como un patrón no similar frecuente, porque aunque la frecuencia ($f_S freq$) usando la función de semejanza original no Booleana (f_S) es mayor o igual que el umbral de frecuencia mínima $minFreq$, la frecuencia ($f_S freq^{'}$) usando la función de semejanza Booleana ($f_S^{'}$) no satisface dicho umbral.

Por otro lado al desarrollar algoritmos para el minado de patrones similares frecuentes usando funciones de semejanza no Booleana debe considerarse el siguiente aspecto relativo a cálculo de la frecuencia de un patrón:

- Si el grado de semejanza entre P y las subdescripciones semejantes a P es muy bajo, pero el número de patrones semejantes a P es elevado, entonces P podría ser un patrón similar frecuente. Sin embargo, sería poco útil pues representaría a

Tabla 4.1: Colección de objetos para ejemplificar el *problema de las bajas semejanzas y los muchos patrones semejantes*.

Ω	r_1	r_2	r_3	r_4
O_1	A	A	A	A
O_2	B	B	B	A
O_3	C	C	A	C
O_4	D	A	D	D
O_5	D	A	D	D

muchos objetos, pero muy poco a cada uno. A esta situación la llamamos *problema de las bajas semejanzas y los muchos patrones semejantes*.

Por ejemplo, dada la colección de 5 objetos descritos por el conjunto de atributos $R = \{r_1, r_2, r_3, r_4\}$ mostrada en la tabla 4.1, una función de semejanza no Booleana que considera el grado de semejanza entre dos subdescripciones de objetos respecto a un conjunto de atributos S ($S \subseteq R$) como la fracción de los atributos en S para la cual ambas descripciones son iguales ($f_S(O, O') = \frac{|\{r \in S | O[r] = O'[r]\}|}{|S|}$) y fijando $minFreq = \frac{2}{5} = 0{,}4$, entonces la frecuencia de la descripción del objeto O_1 es $\frac{1+\frac{1}{4}+\frac{1}{4}+\frac{1}{4}+\frac{1}{4}}{5} = \frac{2}{5} = 0{,}4$ y como consecuencia ésta sería considerada erróneamente un patrón similar frecuente.

Este problema se puede atacar desde la etapa de modelación de la función de semejanza, definiendo la semejanza para los valores pequeños de la misma como cero. Para enfrentar este problema cuando es detectado posteriormente a la modelación de la semejanza, proponemos la siguiente definición de frecuencia.

Definición 4.1. Sea $S \subseteq R$, $S \neq \emptyset$, $O \in \Omega$, y f_S una función de semejanza no Booleana; definimos la frecuencia de una subdescripción $I_S(O)$ en Ω para f_S como:

$$f_S freq(O) = \frac{\sum_{O' \in \Omega,\ f_S(O,O') \geq \beta} f_S(O, O')}{|\Omega|}$$

donde β es un umbral de mínima semejanza para decidir cuándo el grado de semejanza, entre dos subdescripciones de objetos, pueden ser considerado suficientemente alto para ser tenido en cuenta en el cálculo de la frecuencia. El valor del parámetro β puede ser dado por los usuarios en dependencia del problema, la naturaleza de los datos y su experiencia; o puede ser estimado como mostraremos en la sección 4.3.

Para funciones de semejanza no Booleana, las definiciones de patrón similar frecuente, regla de asociación, confianza de una regla y regla de asociación interesante, son la extensión de las definiciones 3.3, 3.4 y 3.5 (para funciones de semejanza Booleana), pero sustituyendo la frecuencia de la Definición 3.2 por la de la Definición 4.1.

4.2. Propiedades de poda

Las propiedades y proposiciones definidas en el capítulo anterior y usadas para la poda del espacio de búsqueda de los patrones similares frecuentes, fueron propuestas para funciones de semejanza Booleana. A continuación extendemos estos resultados para que puedan ser usados en la poda del espacio de búsqueda cuando la función de semejanza es no Booleana.

Definición 4.2. Una función de semejanza no Booleana f_S es monótona no creciente, si y sólo si $\forall S, \hat{S} \subseteq R; O, O' \in \Omega; [\emptyset \neq S \subset \hat{S}] \Rightarrow [f_S(O, O') \geq f_{\hat{S}}(O, O')]$.

La función de semejanza no Booleana (4.1) es monótona no creciente, y la función de semejanza no Booleana (4.2) no lo es.

Proposición 4.1. Dada una colección de objetos Ω y una función de semejanza no Booleana monótona no creciente f_S, si $I_S(O).\mathcal{S} = \{(I_S(O'), f_S(O', O)) \mid O' \in \Omega \wedge I_S(O') \neq I_S(O) \wedge f_S(O', O) \geq \beta\}$ es el conjunto de pares $(I_S(O'), f_S(O', O))$ tal que $f_S(O', O) \geq \beta$; entonces $\forall \hat{S}$, $\hat{S} \supset S : f_{\hat{S}}(O', O) \geq \beta$ si y sólo si $(I_S(O'), f_S(O', O)) \in (I_S(O).\mathcal{S} \cup \{(I_S(O), 1)\})$

Demostración. La demostración resulta inmediata por reducción al absurdo; asumamos que existe $\emptyset \neq S \subseteq \hat{S}$, O, O' tal que $f_{\hat{S}}(O', O) \geq \beta$ y $(I_S(O'), f_S(O', O)) \notin (I_S(O).\mathcal{S} \cup \{(I_S(O), 1)\})$. Entonces $f_S(O', O) < \beta \leq f_{\hat{S}}(O', O)$. Sin embargo, esto es una contradicción puesto que, como f_S es monótona no creciente, $\forall S, \hat{S} \subseteq R; O, O' \in \Omega; [\emptyset \neq S \subset \hat{S}] \Rightarrow [f_S(O', O) \geq f_{\hat{S}}(O', O)]$. Por tanto $(I_S(O'), f_S(O', O)) \in (I_S(O).\mathcal{S} \cup \{(I_S(O), 1)\})$. □

Propiedad 4.1 (Monotonía de la frecuencia)**.** Dada una colección de objetos Ω y una función de semejanza no Booleana f_S; decimos que f_S satisface la propiedad de monotonía de la frecuencia, si y sólo si $\forall S, \hat{S} \subseteq R; O \in \Omega\ [\emptyset \neq S \subset \hat{S}] \Rightarrow [f_S freq(O) \geq f_{\hat{S}} freq(O)]$.

Propiedad 4.2 (f_S-Clausura Descendente)**.** Dada una colección de objetos Ω y una función de semejanza no Booleana f_S; decimos que f_S satisface la propiedad de f_S-Clausura Descendente, si y sólo si $\forall S, \hat{S} \subseteq R; O \in \Omega; \emptyset \neq S \subset \hat{S}\ [f_S freq(O) < minFreq] \Rightarrow [f_{\hat{S}} freq(O) < minFreq]$.

Al igual que para las funciones de semejanza Booleana, la propiedad de f_S-Clausura Descendente para funciones de semejanza no Booleana, no siempre es verdadera. Un ejemplo de función de semejanza no Booleana que no cumple dicha propiedad es la función de semejanza (4.2). Dada la colección de objetos Ω mostrada en la tabla 4.2, el umbral de mínima semejanza $\beta = 0{,}1$, la función de semejanza no Booleana (4.2) y utilizando para cada atributo como criterio de comparación la igualdad, si fijamos $minFreq$ en 0,4 podemos ver que $\exists O = (0, 0, 0) \in \Omega$, $S = \{r_3\} \subset \hat{S} = \{r_1, r_2, r_3\}$, tal que, $f_S freq(O) = \frac{1}{4} = 0{,}25 < minFreq$, y que $f_{\hat{S}} freq(O) = \frac{\frac{3}{3}+\frac{2}{3}+\frac{1}{3}}{4} = 0{,}5 > minFreq$.

Análogamente al caso de funciones de semejanza Booleana, para funciones de semejanza no Booleana, el cumplimiento de la propiedad de f_S-Clausura Descendente depende

Tabla 4.2: Colección de datos para ejemplificar una función de semejanza no Booleana que no cumple la propiedad de f_S-Clausura Descendente.

Ω	r_1	r_2	r_3
O_1	0	0	0
O_2	0	0	1
O_3	0	1	1
O_4	1	1	1

de la monotonía de la frecuencia, la cual a su vez depende de la monotonía de la función de semejanza. Estas dependencias serán demostradas a continuación.

Proposición 4.2. Dada una colección de objetos Ω y una función de semejanza no Booleana f_S, si f_S es monótona no creciente, entonces f_S satisface la monotonía de la frecuencia.

Demostración. Como f_S es una función de semejanza no Booleana monótona no creciente, se cumple que $\forall S, \hat{S} \subseteq R$; $O, O^{'} \in \Omega$;

$$[\emptyset \neq S \subset \hat{S}] \Rightarrow [f_S(O, O^{'}) \geq f_{\hat{S}}(O, O^{'})] \tag{4.3}$$

Por lo tanto $\forall S, \hat{S} \subseteq R$; $O, O^{'} \in \Omega$, $\emptyset \neq S \subset \hat{S}$

$$[f_S(O, O^{'}) < \beta] \Rightarrow [f_{\hat{S}}(O, O^{'}) < \beta] \tag{4.4}$$

A partir de (4.3) y (4.4) se tiene que $\forall S, \hat{S} \subseteq R$; $O \in \Omega$

$$\left[\emptyset \neq S \subset \hat{S}\right] \Rightarrow \left[\sum_{O^{'} \in \Omega | f_S(O,O^{'}) \geq \beta} f_S(O, O^{'}) \geq \sum_{O^{'} \in \Omega | f_{\hat{S}}(O,O^{'}) \geq \beta} f_{\hat{S}}(O, O^{'})\right] \tag{4.5}$$

Nótese, que en (4.5) si un valor de semejanza $f_S(O, O^{'})$ no es adicionado en la primera sumatoria, entonces el valor de semejanza $f_{\hat{S}}(O, O^{'})$ no es adicionado en la segunda sumatoria; pues $f_S(O, O^{'}) \geq f_{\hat{S}}(O, O^{'})$. Como consecuencia de (4.5), $\forall S, \hat{S} \subseteq R$; $O \in \Omega$

$$\left[\emptyset \neq S \subset \hat{S}\right] \Rightarrow \left[\frac{\sum_{O^{'} \in \Omega | f_S(O,O^{'}) \geq \beta} f_S(O, O^{'})}{|\Omega|} \geq \frac{\sum_{O^{'} \in \Omega | f_{\hat{S}}(O,O^{'}) \geq \beta} f_{\hat{S}}(O, O^{'})}{|\Omega|}\right] \tag{4.6}$$

Y por tanto $\forall S, \hat{S} \subseteq R$; $O \in \Omega$ $[\emptyset \neq S \subset \hat{S}] \Rightarrow [f_S freq(O) \geq f_{\hat{S}} freq(O)]$. □

Proposición 4.3. Dada una colección de objetos Ω y una función de semejanza no Booleana f_S, si f_S satisface la monotonía de la frecuencia, entonces f_S satisface la propiedad de f_S-Clausura Descendente.

La demostración de esta proposición es inmediata.$\forall S, \hat{S} \subseteq R;\ O \in \Omega;\ [[\emptyset \neq S \subset \hat{S}] \Rightarrow [f_S freq(O) \geq f_{\hat{S}} freq(O)]] \Rightarrow [[f_S freq(O) < minFreq] \Rightarrow [f_{\hat{S}} freq(O) < minFreq]]$.

Proposición 4.4. Dada una colección de objetos Ω y una función de semejanza no Booleana f_S, si f_S es monótona no creciente, entonces f_S satisface la propiedad de f_S-Clausura Descendente.

Basándose en la Proposición 4.2 y la Proposición 4.3, la demostración de esta proposición es inmediata.

El concepto de patrón f_S-interesante presentado en el capítulo anterior para funciones de semejanza Booleana también es útil para la poda del espacio de búsqueda de patrones similares frecuentes cuando la semejanza no es Booleana. A continuación se extiende este concepto para funciones de semejanza no Booleana.

Definición 4.3. Una subdescripción $I_S(O)$ es un patrón f_S-interesante si $f_S freq(O) \geq minFreq$ o $\exists O' \in \Omega;\ I_S(O') \neq I_S(O),\ [f_S freq(O') \geq minFreq] \Rightarrow [f_S(O', O) \geq \beta]$.

En contraposición:

Definición 4.4. Una subdescripción $I_S(O)$ es un patrón no f_S-interesante si $f_S freq(O) < minFreq$ y $\forall O' \in \Omega;\ I_S(O') \neq I_S(O),\ [f_S freq(O') \geq minFreq] \Rightarrow [f_S(O', O) < \beta]$.

Proposición 4.5. Dada una colección de objetos Ω y una función de semejanza no Booleana monótona no creciente f_S, si una subdescripción $I_S(O)$ es un patrón no f_S-interesante, entonces toda superdescripción $I_{\hat{S}}(O)$ de la misma es también un patrón no f_S-interesante.

Demostración. Si $I_S(O)$ es un patrón no f_S-interesante, entonces

$$f_S freq(O) < minFreq \tag{4.7}$$

y

$$\forall O' \in \Omega;\ I_S(O') \neq I_S(O),\ [f_S freq(O') \geq minFreq] \Rightarrow [f_S(O', O) < \beta] \tag{4.8}$$

Como f_S es una función de semejanza no Booleana monótona no creciente, entonces f_S satisface la propiedad de f_S-Clausura Descendente. Luego se tiene que

$$\forall \hat{S} \subseteq R; S \subset \hat{S},\ [f_S freq(O) < minFreq] \Rightarrow [f_{\hat{S}} freq(O) < minFreq] \tag{4.9}$$

Por otro lado, a partir de (4.8) y de la monotonía de f_S se tiene que

$$\begin{gathered}\forall \hat{S} \subseteq R; O' \in \Omega; S \subset \hat{S}; I_S(O') \neq I_S(O) \\ [f_{\hat{S}} freq(O) \geq minFreq] \Rightarrow [f_S freq(O) \geq minFreq] \Rightarrow [f_{\hat{S}}(O', O) \leq f_S(O', O) < \beta]\end{gathered} \tag{4.10}$$

Adicionalmente, es evidente que

$$\forall \hat{S} \subseteq R; O^{'} \in \Omega; S \subset \hat{S},\ [I_S(O^{'}) \neq I_S(O)] \Rightarrow [I_{\hat{S}}(O^{'}) \neq I_{\hat{S}}(O)] \tag{4.11}$$

Por tanto, a partir de (4.9) (4.10) y (4.11) obtenemos que $\forall \hat{S} \subseteq R; S \subset \hat{S}$

$$f_{\hat{S}} freq(O) < minFreq \tag{4.12}$$

y

$$\forall O^{'} \in \Omega;\ I_{\hat{S}}(O^{'}) \neq I_{\hat{S}}(O)\ [f_{\hat{S}} freq(O^{'}) \geq minFreq] \Rightarrow [f_{\hat{S}}(O^{'}, O) < \beta] \tag{4.13}$$

Es decir, toda superdescripción de un patrón no f_S-interesante es también un patrón no f_S-interesante. □

Como consecuencia de la definición de patrón no f_S-interesante y de la Proposición 4.5, si la función de semejanza no Booleana es monótona no creciente, entonces los patrones no f_S-interesantes pueden ser podados sin perder patrones similares frecuentes.

4.3. Estimación del Umbral de Mínima Semejanza β

Los atributos que describen a un objeto pueden ser de diferente naturaleza. Incluso, es de esperar que la semejanza entre dos objetos respecto a un mismo atributo en dos problemas o contextos distintos pueda ser diferente. Por consiguiente, para estimar el umbral de mínima semejanza tiene sentido considerar los valores de semejanza entre los objetos de una colección tanto respecto a cada atributo como respecto a cada subconjunto atributos. Nosotros proponemos estimar dicho umbral, a partir de umbrales parciales de semejanza calculados para cada subconjunto de atributos. El umbral parcial de semejanza para cada subconjunto de atributos S es denotado como β_S y estimado a partir del grado de semejanza entre las subdescripciones de los objetos como sigue:

$$\beta_S = \begin{cases} \min\limits_{O \in \Omega | M_S(O) \neq 0} M_S(O) & \text{si } \exists O \in \Omega \mid M_S(O) \neq 0 \\ 1 \quad \text{en otro caso} \end{cases}$$

donde

$$M_S(O) = \begin{cases} \max\limits_{\substack{O^{'} \in \Omega \\ f_S(O,O^{'}) \neq 1 \\ I_S(O) \neq I_S(O^{'})}} f_S(O, O^{'}) & \text{si } \exists O^{'} \in \Omega | f_S(O, O^{'}) \neq 1,\ I_S(O) \neq I_S(O^{'}) \\ 0 \quad \text{en otro caso} \end{cases}$$

Así, para un S subconjunto de R, una subdescripción contribuye a la frecuencia de otra subdescripción si el grado de semejanza entre ambas es mayor que el menor grado de semejanza entre cada subdescripción y la subdescripción más semejante a ésta.

Como consecuencia, los grados de semejanza más pequeños son excluidos del cálculo de la frecuencia. No obstante, si para cada subdescripción, todos los grados de semejanza entre ella y las demás subdescripciones no pertenecen a $\{0, 1\}$ (lo cual es de esperar si f_S es una función de semejanza no Booleana), entonces aun cuando los grados de semejanza entre una subdescripción y las demás subdescripciones sean muy pequeños, existe al menos una subdescripción que contribuye a su frecuencia.

En esta formulación, el máximo grado de semejanza 1 no es considerado debido a que es deseable que $\beta_S \neq 1$. Si β_S tomara valor 1, entonces el uso de funciones de semejanza no Booleana sería equivalente a transformar las funciones de semejanza no Booleana en funciones de semejanza Booleana mediante el umbral $\alpha = 1$. Además, el mínimo grado de semejanza 0 no es considerado. Esto se debe a que si $\beta_S = 0$ entonces todos los grados de semejanza serían considerados suficientes para ser tenidos en cuenta en el cálculo de la frecuencia.

La estimación del umbral de mínima semejanza β podría realizarse a partir de todos los umbrales parciales de semejanza. Esta alternativa parece aceptable debido a que no obvia la información que aporta cada β_S, sin embargo, no es viable pues el número de umbrales de semejanza es exponencial respecto al número de atributos R $(2^{|R|} - 1)$.

Una alternativa es considerar el umbral de mínima semejanza β como β_R. De esta forma el costo computacional no es comprometido. Sin embargo, como generalmente la semejanza entre objetos distintos es menor al considerar más atributos que al considerar menos atributos, el umbral de mínima semejanza β tiende a ser muy pequeño al considerar todos los atributos (tiende a ser el más pequeño de los β_S). Consecuentemente, muy pocas semejanzas tienden a ser menores que β y por tanto a no ser tenidas en cuenta en el cálculo de la frecuencia; por lo que el *problema de las bajas semejanzas y los muchos patrones semejantes* puede no ser resuelto.

Otra alternativa es estimar el umbral de mínima semejanza β a partir de los umbrales parciales de semejanza β_S, con $|S| = 1$. A diferencia del umbral de semejanza β_R, que tiende a ser el más pequeño de los umbrales de semejanza, los umbrales de semejanza β_S con $|S| = 1$, tienden a ser los umbrales de semejanza más grandes. En contraproposición a la alternativa anterior ($\beta = \beta_R$), al escoger un β_S con $|S| = 1$ como umbral de mínima semejanza β, muchas semejanzas tienden a ser menores que β y por tanto a no ser tenidas en cuenta en el cálculo de la frecuencia; con lo cual se ataca el *problema de las bajas semejanzas y los muchos patrones semejantes*. Sin embargo, no es una buena opción escoger un β_S con $|S| = 1$ muy grande, pues se obviarían del cálculo de la frecuencia, semejanzas que no son bajas. Por estas razones recomendamos estimar el umbral de mínima semejanza β como sigue:

$$\beta = \min_{r \in R} \beta_{\{r\}}$$

4.4. Estructura de Datos $STree^*$

Para facilitar el cálculo de la frecuencia y reducir el número de evaluaciones de la función de semejanza no Booleana, proponemos extender la estructura de datos $STree$ a una estructura más adecuada para funciones de semejanza no Booleana que denominamos $STree^*$. Al igual que un $STree_S$, un $STree^*_S$ es un árbol donde cada camino desde la raíz hasta una hoja representa una subdescripción P de un objeto O respecto al conjunto S ($P = I_S(O)$). En cada hoja se almacena:

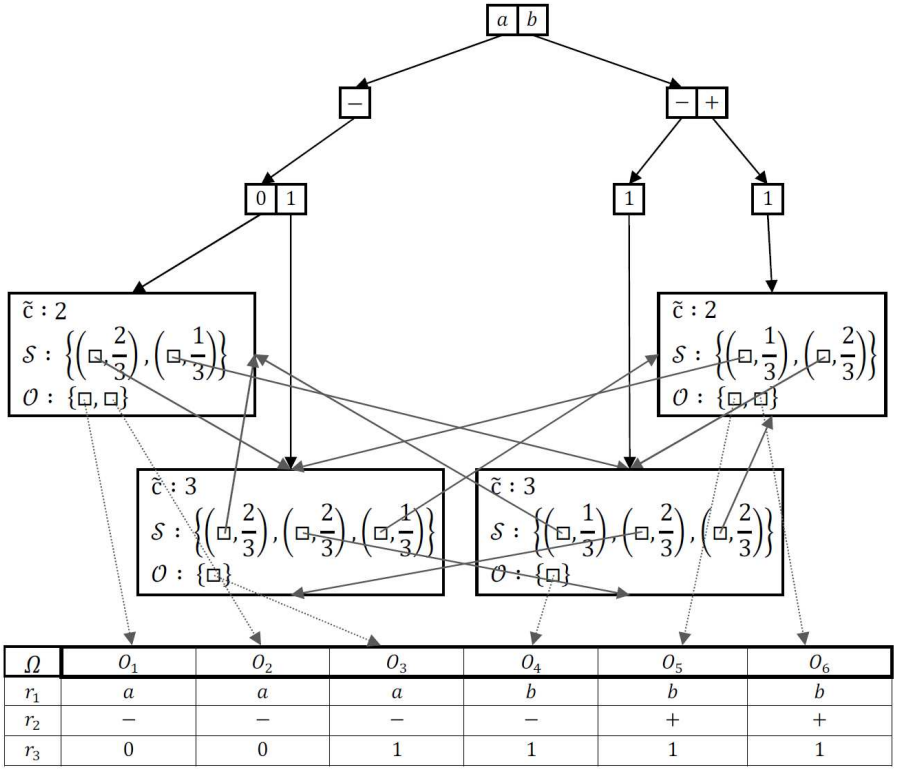
- $P.\tilde{c}$: Número de repeticiones de subdescripciones semejantes a P pero no idénticas, con grado de semejanza mayor o igual que β.
- $P.\mathcal{O}$: Conjunto de objetos que contienen a la subdescripción P.
- $P.\mathcal{S}$: Conjunto de pares $(subdescripción, sim)$ tales que el grado de semejanza sim de P a $subdescripción$ es al menos igual a β, pero P y $subdescripción$ no son idénticas. Formalmente, sea $\mathcal{I}_S$ el conjunto de subdescripciones (no idénticas) de objetos respecto al conjunto de atributos S, $P.\mathcal{S} = \{(I_S(O^{'}), f_S(O^{'}, O)) \mid I_S(O^{'}) \in \mathcal{I}_S \wedge I_S(O^{'}) \neq P \wedge f_S(I_S(O^{'}), P) \geq \beta\}$.

En la figura 4.2 se muestra la estructura $STree^*_{\{r_1,r_2,r_3\}}$ para una colección $\Omega = \{O_1, O_2, O_3, O_4, O_5, O_6\}$, la función de semejanza (4.2) tomando como criterio de comparación la igualdad y $\beta = 0{,}25$. De izquierda a derecha, los caminos desde la raíz hasta cada hoja representan respectivamente a las subdescripciones $(a, -, 0)$, $(a, -, 1)$, $(b, -, 1)$ y $(b, +, 1)$. Para la subdescripción $P = (a, -, 0)$, $P.\mathcal{O} = \{O_1, O_2\}$ pues los objetos O_1 y O_2 contienen a P, $P.\mathcal{S} = \{((a, -, 1), \frac{2}{3}), ((b, -, 1), \frac{1}{3})\}$ pues la subdescripción P es semejante con grado mayor o igual que β pero no idéntica a las subdescripciones $(a, -, 1)$ y $(b, -, 1)$; y $P.\tilde{c} = 2$ pues las subdescripciones $(a, -, 1)$ y $(b, -, 1)$ son semejantes a P con grado mayor o igual que β pero no son idénticas a P.

Mediante un $STree^*_S$ la frecuencia de una subdescripción $I_S(O)$ puede calcularse mediante:

$$f_S freq(O) = |I_S(O).\mathcal{O}| + \sum_{P \in \mathcal{I}_S | (I_S(O), sim) \in P.\mathcal{S}} sim * |P.\mathcal{O}|$$

En esta expresión $|I_S(O).\mathcal{O}|$ representa el número de repeticiones de la subdescripción $I_S(O)$ en Ω y dado que las semejanzas entre las mismas e $I_S(O)$ son 1, no es necesario calcularlas. Por su parte, $\sum_{P \in \mathcal{I}_S | (I_S(O), sim) \in P.\mathcal{S}} sim * |P.\mathcal{O}|$ representa la suma de las semejanzas con valor mayor o igual a β entre $I_S(O)$ y el resto de las subdescripciones de Ω que no son idénticas a $I_S(O)$. Particularmente $|P.\mathcal{O}|$ representa el número de repeticiones de P en Ω y por cada $P \in \mathcal{I}_S$, sim representa la semejanza entre $I_S(O)$ y P. Nótese que por cada P, como la semejanza entre sus repeticiones e $I_S(O)$ son iguales a sim, la suma de las mismas es igual a $sim * |P.\mathcal{O}|$. De esta forma se puede reducir el número de evaluaciones de la función de semejanza.



Ω	O_1	O_2	O_3	O_4	O_5	O_6
r_1	a	a	a	b	b	b
r_2	−	−	−	−	+	+
r_3	0	0	1	1	1	1

Figura 4.2: Ejemplo de estructura $STree^*_{\{r_1,r_2,r_3\}}$.

Por otro lado, debido a la Proposición 4.1, sea f_S una función de semejanza no Booleana f_S monótona no creciente, si al calcular la frecuencia de una superdescripción $I_{\hat{S}}(O)$ de $I_S(O)$ se tiene para todo $I_S(O^{'}) \in \mathcal{I}_S$ el conjunto $I_S(O^{'}).\mathcal{S}$, entonces puede reducirse aún más el número de evaluaciones de f_S, pues no es necesario calcular la semejanza entre $I_{\hat{S}}(O)$ y todas las subdescripciones $I_{\hat{S}}(O^{'}) \in \mathcal{I}_{\hat{S}}$. Sólo es necesario calcular la semejanza entre $I_{\hat{S}}(O)$ y las superdescripciones $I_{\hat{S}}(O^{'})$ que sean una superdescripción de alguna subdescripción $I_S(O^{'})$ tal que $I_S(O) \in (I_S(O^{'}).\mathcal{S} \cup \{I_S(O^{'})\})$.

Como consecuencia, la frecuencia de una subdescripción $I_{\hat{S}}(O)$ puede calcularse mediante:

$$f_{\hat{S}}freq(O) = |I_{\hat{S}}(O).\mathcal{O}| + \sum_{I_{\hat{S}}(O') \in \mathcal{I}_{\hat{S}} | \hat{S} \supset S \wedge (I_S(O), sim) \in I_S(O').\mathcal{S}} f_{\hat{S}}(O, O^{'}) * |I_{\hat{S}}(O^{'}).\mathcal{O}|$$

En esta expresión $\sum_{I_{\hat{S}}(O') \in \mathcal{I}_{\hat{S}} | \hat{S} \supset S \wedge (I_S(O), sim) \in I_S(O').\mathcal{S}} f_{\hat{S}}(O, O^{'}) * |I_{\hat{S}}(O^{'}).\mathcal{O}|$ representa la suma de las semejanzas con valor mayor o igual a β entre $I_{\hat{S}}(O)$ y el resto de las subdescripciones de Ω que no son idénticas a $I_{\hat{S}}(O)$.

Análogamente a la estructura $STree$ presentada en el capítulo anterior, la estructura $STree^*$ es construida y usada por los algoritmos de minado de patrones similares frecuentes que proponemos a continuación.

4.5. Algoritmos de minado de patrones similares frecuentes

En las siguientes secciones proponemos dos algoritmos de minado de patrones similares frecuentes para funciones de semejanza no Booleana: *STree*DC-Miner* para funciones de semejanza monótonas no crecientes, lo cual implica que dichas funciones satisfacen la propiedad f_S-Clausura Descendente y *STree*NDC-Miner* para funciones que no satisfacen esta propiedad. Análogamente al capítulo anterior, el primer algoritmo poda el espacio de búsqueda de patrones similares frecuentes, mientras el segundo, explora exhaustivamente dicho espacio.

Adicionalmente, proponemos otro algoritmo (*RP*-Miner*) para funciones de semejanza que no cumplen la propiedad de monotonía. *RP*-Miner* está basado en podas relajadas, razón por la cual, aunque es más eficiente que *STree*NDC-Miner*, puede no encontrar todos los patrones similares frecuentes.

4.5.1. *STree*DC-Miner*

El algoritmo *STree*DC-Miner* está diseñado para funciones de semejanza no Booleana monótonas no crecientes, lo cual implica que dichas funciones satisfacen la propiedad de f_S-Clausura Descendente. *STree*DC-Miner* es una extensión del algoritmo *STreeDC-Miner* en la cual:

- Se sustituye la estructura de datos $STree$ por la estructura de datos $STree^*$ para permitir manipular las relaciones de semejanza no Booleana entre las subdescripciones.

- Se incluye el concepto de *umbral de mínima semejanza* en el cálculo de la frecuencia, para atacar el *problema de las bajas semejanzas y los muchos patrones semejantes.*

- Se utiliza el concepto de *patrón no f_S-interesante*, basado en el concepto de umbral de semejanza, para la poda del espacio de búsqueda.

Sea $\prec$ un orden lineal en R y f_S una función de semejanza no Booleana monótona no creciente, entonces tenemos que:

- Un conjunto de atributos S es expandible, si y sólo si $S = \emptyset$ o existe al menos un patrón similar frecuente $I_S(O)$.

- Un conjunto de atributos $\hat{S}$ es una expansión directa de S, si y sólo si S es expandible, $\hat{S} = S \cup \{r\}$, $r \in R$ y $\forall r^{'} \in S$, $r^{'} \prec r$.

- Un conjunto de atributos $\hat{\hat{S}}$ es una expansión de S, si y sólo si $\hat{\hat{S}}$ es una expansión directa de S, o existe un conjunto de atributos $\hat{S}$, tal que $\hat{S}$ es una expansión directa de S y $\hat{\hat{S}}$ es una expansión de $\hat{S}$.

Al igual que en el algoritmo *STreeDC-Miner*, en el algoritmo *STree*DC-Miner* obtenemos las expansiones del conjunto vacío de atributos, y por cada expansión $\hat{\hat{S}}$ buscamos los patrones similares frecuentes. Análogamente, para expandir un conjunto de atributos S, S debe ser expandible y para esto, es necesario obtener primero los patrones similares frecuentes respecto a S.

Al iniciar el algoritmo, el conjunto de patrones frecuentes F y el conjunto de atributos a expandir $\hat{S}$ son vacíos. Además, la estructura $STree^*_S$ es *null*.

En dependencia del conjunto de atributos a expandir, *STree*DC-Miner* considera los siguientes casos:

- $\hat{S} = \emptyset$. El algoritmo se llama a sí mismo recursivamente para cada expansión directa de $\hat{S}$ (líneas 28-30).

- $|\hat{S}| = 1$. Todos los objetos de la colección son adicionados a $STree^*_{\hat{S}}$ (líneas 4-7). Después de esto, las semejanzas entre todas las subdescripciones contenidas en $STree^*_{\hat{S}}$ son calculadas, y para cada subdescripción P' en $STree^*_{\hat{S}}$, la lista de subdescripciones a las cuales P' es semejante es actualizada uniendo a ésta las subdescripciones P tales que $f_{\hat{S}}(P, P') \geq \beta$ (líneas 8-11). Luego, para cada subdescripción P contenida en $STree^*_{\hat{S}}$, $P.\tilde{c}$ es calculado; los patrones similares frecuentes son obtenidos y el conjunto de patrones frecuentes es actualizado. Además, los patrones no f_S-interesantes son eliminados de $STree^*_{\hat{S}}$ (líneas 22-26). Finalmente, si

Procedimiento `STree*DC-Miner`($STree^*_S$,$\hat{S}$,Ω, $f_{\hat{S}}$, β, $minFreq$)

Input: $STree^*_S$ - Estructura de Datos, $\hat{S}$ - Conjunto de atributos, Ω - Colección de datos, f_S - Función de semejanza no Booleana, β - Umbral de mínima semejanza, $minFreq$ - Umbral de mínimo soporte.

Output: F - Conjunto de patrones similares frecuentes.

```
1  if Ŝ ≠ ∅ then
2      STree*_Ŝ ← emptySTree*_Ŝ
3      if |Ŝ| = 1 then
4          foreach object O ∈ Ω do
5              if ¬STree*_Ŝ.contain(I_Ŝ(O)) then
6                  STree*_Ŝ.add(I_Ŝ(O))
7              STree*_Ŝ.I_Ŝ(O).O ← STree*_Ŝ.I_Ŝ(O).O ∪ {O}
8          foreach P, P' ∈ STree*_Ŝ do
9              sim ← f_Ŝ(P, P')
10             if sim ≥ β then
11                 P'.S ← P'.S ∪ {(P, sim)}
12     else
13         foreach P ∈ STree*_S do
14             foreach object O ∈ P.O do
15                 if ¬STree*_Ŝ.contain(I_Ŝ(O)) then
16                     STree*_Ŝ.add(I_Ŝ(O))
17                 STree*_Ŝ.I_Ŝ(O).O ← STree*_Ŝ.I_Ŝ(O).O ∪ {O}
18         foreach P, P' ∈ STree*_Ŝ such that I_S(P) ∈ F, (I_S(P),) ∈ I_S(P').S do
19             sim ← f_Ŝ(P, P')
20             if sim ≥ β then
21                 P'.S ← P'.S ∪ {(P, sim)}
22     foreach P ∈ STree*_Ŝ do
23         foreach (P', sim) ∈ P.S do
24             P'.c̃ ← P'.c̃ + |P.O| * sim
25     F ← {P ∈ STree*_Ŝ | P.c̃ + |P.O| ≥ minFreq}
26     STree_Ŝ.removeNonf_SInterestingPatterns()
27 if Ŝ = ∅ ∨ F ≠ ∅ then
28     foreach expansión directa Ŝ̂ of Ŝ do
29         F ← F∪STree*DC-Miner(STree*_Ŝ,Ŝ̂,Ω, f_Ŝ̂, B, minFreq)
```

Algoritmo 4.1: Algoritmo *STree*DC-Miner*.

el conjunto de patrones similares frecuentes respecto a $\hat{S}$ no es vacío, el algoritmo se llama a sí mismo recursivamente para cada expansión directa de $\hat{S}$ (líneas 27-29).

- $|\hat{S}| > 1$. Para cada subdescripción P contenida en $STree^*_S$, los objetos contenidos en $P.\mathcal{O}$, son adicionados a $STree^*_{\hat{S}}$. Nótese que, llegando a este punto, los patrones no f_S-interesantes ya han sido eliminados de $STree^*_S$, y por tanto, $STree^*_S$ sólo contiene patrones f_S-interesantes (líneas 13-17). Después de esto, sólo las semejanzas entre las subdescripciones P y $P^{'}$ contenidas en $STree^*_{\hat{S}}$, tales que $f_S(P, P^{'}) \geq \beta$ o lo que lo mismo $(I_S(P), f_S(P, P^{'})) \in I_S(P^{'}).\mathcal{S}$, son calculadas. Además, la lista de subdescripciones a las cuales P' es semejante, es actualizada uniendo a ésta las subdescripciones P tales que $f_{\hat{S}}(P, P^{'}) \geq \beta$ (líneas 18-21). Luego, para cada subdescripción P contenida en $STree^*_{\hat{S}}$, $P.\tilde{c}$ es calculado; los patrones similares frecuentes son obtenidos y el conjunto de patrones frecuentes es actualizado. Además, los patrones no f_S-interesantes son eliminados de $STree^*_{\hat{S}}$ (líneas 22-26). Finalmente, si el conjunto de patrones similares frecuentes respecto a $\hat{S}$ no es vacío, el algoritmo se llama a sí mismo recursivamente para cada expansión directa de $\hat{S}$ (líneas 27-29).

4.5.2. *STree*NDC-Miner*

El algoritmo *STree*NDC-Miner* es una extensión del algoritmo *STreeNDC-Miner* diseñada para funciones de semejanza no Booleana para las que no se puede asegurar el cumplimiento de la propiedad de f_S-Clausura Descendente. Como consecuencia, no es posible usar esta propiedad para podar el espacio de búsqueda de patrones similares frecuentes sin que puedan perderse patrones similares frecuentes. Y por tanto, al igual que en el algoritmo *STreeNDC-Miner* para garantizar que todos los patrones similares frecuentes puedan ser obtenidos, se buscarán para todo $S \subseteq R$, $S \neq \emptyset$, lo cual implica una exploración exhaustiva del espacio de búsqueda.

El algoritmo *STree*NDC-Miner* es similar a *STreeNDC-Miner* pero:

- Se sustituye la estructura de datos $STree$ por la estructura de datos $STree^*$ para permitir manipular las relaciones de semejanza no Booleana entre las subdescripciones.

- Se sustituye en la estructura de datos $STree^*$ el atributo $\mathcal{O}$ asociado a cada subdescripción por $\bar{\bar{c}}$ ($\bar{\bar{c}} = |\mathcal{O}|$).

- Se incluye el concepto de *umbral de mínima semejanza* en el cálculo de la frecuencia, para atacar el *problema de las bajas semejanzas y los muchos patrones semejantes.*

Sea $\prec$ un orden lineal en R, entonces tenemos que:

- Un conjunto de atributos S es reducible, si y sólo si $|S| > 1$.

- Un conjunto de atributos $\check{S}$ es una reducción directa de S, si y sólo si S es reducible, $\check{S} = S - r$, $r \in S$ y $\forall r^{'} \in (R - S)$, $r^{'} \prec r$.

- Un conjunto de atributos $\check{\check{S}}$ es una reducción de S, si y sólo si $\check{\check{S}}$ es una reducción directa de S, o existe un conjunto de atributos $\check{S}$, tal que $\check{S}$ es una reducción directa de S y $\check{\check{S}}$ es una reducción de $\check{S}$.

Para descubrir los patrones similares frecuentes en una colección de objetos Ω, *STree*NDC-Miner* obtiene todas las reducciones de R, por medio de consecutivas reducciones directas. Para cada reducción $\check{S}$, los patrones similares frecuentes son obtenidos.

Al iniciar el algoritmo, el conjunto de patrones frecuentes F es vacío y el conjunto de atributos a reducir $\check{S} = R$. Además, la estructura $STree^*_S$ es *null*.

Procedimiento `STree*NDC-Miner(`$STree^*_S$,$\check{S}$,Ω, $f_{\check{S}}$, β, $minFreq$`)`

Input: $STree^*_S$ - Estructura de Datos, $\check{S}$ - Conjunto de atributos, Ω - Colección de datos, f_S - Función de semejanza no Booleana, β - Umbral de mínima semejanza, $minFreq$ - Umbral de mínimo soporte.
Output: F - Conjunto de patrones similares frecuentes.

1 $STree^*_{\check{S}} \leftarrow emptySTree^*_{\check{S}}$
2 **if** $\check{S} = R$ **then**
3 **foreach** *object* $O \in \Omega$ **do**
4 **if** $\neg STree^*_{\check{S}}.contain(I_{\check{S}}(O))$ **then**
5 $STree^*_{\check{S}}.add(I_{\check{S}}(O))$
6 $STree^*_{\check{S}}.I_{\check{S}}(O).\bar{\bar{c}} \leftarrow STree^*_{\check{S}}.I_{\check{S}}(O).\bar{\bar{c}} + 1$
7 **else**
8 **foreach** $P \in STree^*_S$ **do**
9 **if** $\neg STree^*_{\check{S}}.contain(I_{\check{S}}(P))$ **then**
10 $STree^*_{\check{S}}.add(I_{\check{S}}(P))$
11 $STree^*_{\check{S}}.I_{\check{S}}(P).\bar{\bar{c}} \leftarrow STree^*_{\check{S}}.I_{\check{S}}(P).\bar{\bar{c}} + STree^*_S.I_S(P).\bar{\bar{c}}$
12 **foreach** $P, P^{'} \in STree^*_{\check{S}}$ *such that* $P \neq P^{'}$ **do**
13 $sim \leftarrow f_{\check{S}}(P, P^{'})$
14 **if** $sim \geq \beta$ **then**
15 $P^{'}.\mathcal{S} \leftarrow P^{'}.\mathcal{S} \cup \{(P, sim)\}$
16 **foreach** $P \in STree^*_{\check{S}}$ **do**
17 **foreach** $(P^{'}, sim) \in P.\mathcal{S}$ **do**
18 $P^{'}.\tilde{c} \leftarrow P^{'}.\tilde{c} + P.\bar{\bar{c}} * sim$
19 $F \leftarrow \{P \in STree^*_{\check{S}} \mid P.\tilde{c} + |P.\mathcal{O}| \geq minFreq\}$
20 **foreach** *reducción directa* $\check{\check{S}}$ *of* $\check{S}$ **do**
21 $F \leftarrow F \cup$ `STree*NDC-Miner(`$STree^*_{\check{S}}$,$\check{\check{S}}$,Ω, $f_{\check{\check{S}}}$, $\mathcal{B}$, $minFreq$`)`

Algoritmo 4.2: Algoritmo *STree*NDC-Miner*.

En dependencia del conjunto de atributos a reducir, *STree*NDC-Miner* considera los siguientes casos:

- $\check{S} = R$. Se adicionan todas las subdescripciones respecto a R de los objetos de la colección a $STree^*_{\check{S}}$ (líneas 3-6).
- $\check{S} \subset R$. Se adicionan todas las subdescripciones respecto a $\check{S}$ de las subdescripciones contenidas en $STree^*_S$ a $STree^*_{\check{S}}$ (líneas 8-11).

Después de esto, se calculan las semejanzas entre todas las subdescripciones contenidas en $STree^*_{\check{S}}$, y para cada subdescripción P' en $STree^*_{\check{S}}$, la lista de subdescripciones a las cuales P' es semejante es actualizada uniendo a ésta las subdescripciones P tales que $f_{\check{S}}(P, P') \geq \beta$ (líneas 12-15). Luego, para cada subdescripción P contenida en $STree^*_{\check{S}}$, $P.\tilde{c}$ es calculado; los patrones similares frecuentes son obtenidos y el conjunto de patrones frecuentes es actualizado (líneas 16-19). Finalmente, el algoritmo se llama a sí mismo recursivamente para cada reducción directa de $\check{S}$. (líneas 20-21).

4.5.3. *RP*-Miner*

Si las funciones de semejanza no Booleana empleadas para calcular la frecuencia de subdescripciones de objetos no son monótonas no crecientes, entonces pueden existir superdescripciones de subdescripciones no frecuentes que sean frecuentes. Por tanto, si son podados todos los patrones no f_S-interesantes (*STree*DC-Miner*), entonces pueden perderse patrones similares frecuentes. Por otro lado, si el espacio de búsqueda de los patrones similares frecuentes es explorado exhaustivamente (*STree*NDC-Miner*), entonces todos los patrones similares frecuentes son encontrados, pero este proceso es muy costoso.

Para reducir el número de patrones similares frecuentes que pueden perderse si se emplea el algoritmo *STree*DC-Miner* y a la vez no explorar exhaustivamente el espacio de búsqueda, proponemos el algoritmo *RP*-Miner*. El mismo es una extensión del algoritmo *RP-Miner*, en la cual:

- Se sustituye la estructura de datos $STree$ por la estructura de datos $STree^*$ para permitir manipular las relaciones de semejanza no Booleana entre las subdescripciones.
- Se incluye el concepto de *umbral de mínima semejanza* en el cálculo de la frecuencia, para atacar el *problema de las bajas semejanzas y los muchos patrones semejantes.*
- Se utiliza el concepto de *patrón no f_S-interesante*, basado en el concepto de *umbral de semejanza*, para la poda del espacio de búsqueda.

La idea de este algoritmo es la siguiente: Primero son obtenidos los patrones similares frecuentes de tamaño 1. Luego los patrones similares frecuente son expandidos sucesivamente por medio de la adición de un atributo. En este proceso de expansión, si un

patrón que ya fue obtenido es obtenido nuevamente, el mismo no es analizado como posible patrón frecuente, ni expandido.

Al iniciar el algoritmo, el conjunto de patrones analizados W (frecuentes o no, pero de tamaño mayor que 1) y el conjunto de patrones frecuentes F son vacíos. Además el conjunto de atributos a expandir $\hat{S}$ también es vacío y la estructura $STree^*_{\hat{S}}$ es *null*.

En dependencia del conjunto de atributos a expandir *RP*-Miner* considera los siguientes casos:

- $\hat{S} = \emptyset$. El algoritmo se llama a sí mismo recursivamente para cada expansión directa de S (líneas 30-32)

- $|\hat{S}| = 1$. Todos los objetos de la colección son adicionados a $STree^*_{\hat{S}}$ (líneas 4-7). Después de esto, las semejanzas entre todas las subdescripciones contenidas en $STree^*_{\hat{S}}$ son calculadas, y para cada subdescripción P' en $STree^*_{\hat{S}}$, la lista de subdescripciones a las cuales P' es semejante es actualizada uniendo a ésta las subdescripciones P tales que $f_{\hat{S}}(P, P') \geq \beta$ (líneas 8-11). Luego, para cada subdescripción P contenida en $STree^*_{\hat{S}}$, $P.\tilde{c}$ es calculado; los patrones similares frecuentes son obtenidos y el conjunto de patrones frecuentes es actualizado. Además, los patrones no f_S-interesantes son eliminados de $STree^*_{\hat{S}}$ (líneas 24-28). Finalmente, si el conjunto de patrones similares frecuentes respecto a $\hat{S}$ no es vacío, el algoritmo se llama a sí mismo recursivamente para cada expansión $\hat{S} \cup \{r\}$, $r \in (R - \hat{S})$ de $\hat{S}$ (líneas 29-31).

- $|\hat{S}| > 1$. Para cada subdescripción P contenida en $STree^*_S$, cada objeto O contenido en $P.\mathcal{O}$, tal que $I_{\hat{S}}(O)$ no ha sido analizado, es adicionado a $STree^*_{\hat{S}}$ e $I_{\hat{S}}(O)$ es adicionado al conjunto de patrones analizados W. Nótese que llegando a este punto los patrones no f_S-interesantes ya han sido eliminados de $STree^*_S$, y por tanto, $STree^*_S$ sólo contiene patrones f_S-interesantes (líneas 13-19). Después de esto, sólo las semejanzas entre las subdescripciones P y P' contenidas en $STree^*_{\hat{S}}$, tales $f_S(P, P') \geq \beta$ o lo que lo mismo $(I_S(P), f_S(P, P')) \in I_S(P').\mathcal{S}$, son calculadas. Además, la lista de subdescripciones a las cuales P' es semejante es actualizada uniendo a ésta las subdescripciones P tales que $f_{\hat{S}}(P, P') \geq \beta$ (líneas 20-23). Luego, para cada subdescripción P contenida en $STree^*_{\hat{S}}$, $P.\tilde{c}$ es calculado; los patrones similares frecuentes son obtenidos y el conjunto de patrones frecuentes es actualizado. Además, los patrones no f_S-interesantes son eliminados de $STree^*_{\hat{S}}$ (líneas 24-28). Finalmente, si el conjunto de patrones similares frecuentes respecto a $\hat{S}$ no es vacío, el algoritmo se llama a sí mismo recursivamente para cada expansión $\hat{S} \cup \{r\}$, $r \in (R - \hat{S})$ de $\hat{S}$ (líneas 29-31).

El algoritmo *RP*-Miner* a diferencia *STree* DC-Miner*, no se llama a sí mismo recursivamente para cada expansión directa de conjunto $\hat{S}$, sino que lo hace para cada expansión $\hat{S} \cup \{r\}$, $r \in (R - \hat{S})$ de $\hat{S}$. De esta forma una subdescripción respecto a k atributos puede ser obtenida mediante la expansión de cada una de sus subdescripciones

Procedimiento RP*-Miner($STree^*_S$,$\hat{S}$,Ω, $f_{\hat{S}}$, β, $minFreq$)

Input: $STree^*_S$ - Estructura de Datos, $\hat{S}$ - Conjunto de atributos, Ω - Colección de datos, f_S - Función de semejanza no Booleana, β - Umbral de mínima semejanza, $minFreq$ - Umbral de mínimo soporte.

Output: F - Conjunto de patrones similares frecuentes.

```
1  if Ŝ ≠ ∅ then
2      STree*_Ŝ ← emptySTree*_Ŝ
3      if |Ŝ| = 1 then
4          foreach object O ∈ Ω do
5              if ¬STree*_Ŝ.contain(I_Ŝ(O)) then
6                  STree*_Ŝ.add(I_Ŝ(O))
7              STree*_Ŝ.I_Ŝ(O).𝒪 ← STree*_Ŝ.I_Ŝ(O).𝒪 ∪ {O}
8          foreach P, P' ∈ STree*_Ŝ do
9              sim ← f_Ŝ(P, P')
10             if sim ≥ β then
11                 P'.𝒮 ← P'.𝒮 ∪ {(P, sim)}
12     else
13         foreach P ∈ STree*_Ŝ do
14             foreach object O ∈ P.𝒪 do
15                 if I_Ŝ(O) ∉ W then
16                     if ¬STree*_Ŝ.contain(I_Ŝ(O)) then
17                         STree*_Ŝ.add(I_Ŝ(O))
18                     STree*_Ŝ.I_Ŝ(O).𝒪 ← STree*_Ŝ.I_Ŝ(O).𝒪 ∪ {O}
19                     W ← W ∪ I_Ŝ(O)
20         foreach P, P' ∈ STree*_Ŝ such that I_S(P) ∈ F, I_S(P) ∈ I_S(P').𝒮 do
21             sim ← f_Ŝ(P, P')
22             if sim ≥ β then
23                 P'.𝒮 ← P'.𝒮 ∪ {(P, sim)}
24     foreach P ∈ STree*_Ŝ do
25         foreach (P', sim) ∈ P.𝒮 do
26             P'.c̃ ← P'.c̃ + |P.𝒪| * sim
27     F ← {P ∈ STree*_Ŝ | P.c̃ + |P.𝒪| ≥ minFreq}
28     STree_Ŝ.removeNonfSInterestingPatterns()
29 if Ŝ = ∅ ∨ F ≠ ∅ then
30     foreach r ∈ (R − Ŝ) do
31         F ← F∪RP*-Miner(STree*_Ŝ,Ŝ ∪ {r},Ω, f_Ŝ, minFreq)
```

Algoritmo 4.3: Algoritmo *RP*-Miner*.

respecto a $k-1$ atributos. Otra diferencia es que cuando $|\hat{S}| > 1$, para cada subdescripción P contenida en $STree^*_S$, sólo los objetos contenidos en $P.\mathcal{O}$ cuya subdescripción respecto a S no ha sido analizada, son insertados en la estructura de datos $STree^*_{\hat{S}}$ y sus subdescripciones respecto S son adicionadas al conjunto de patrones analizados W. De esta forma, una subdescripción sólo es analizada, o expandida en caso de ser f_S-frecuente, una vez.

4.6. Algoritmo de Minado de Reglas de Asociación

Para generar reglas de asociación interesantes a partir de patrones similares frecuentes usando funciones de semejanza no Booleana proponemos emplear el algoritmo *FSP-GenRules* mostrado en el capítulo anterior.

En este caso, al iniciar el algoritmo *FSP-GenRules*, el conjunto F contiene los patrones similares frecuentes, considerando como frecuencia la Definición 4.1, descubiertos por alguno de los algoritmos propuestos en las secciones anteriores (*STree*DC-Miner*, *STree*NDC-Miner*, *RP*-Miner*) y el conjunto de reglas de asociación generadas RA es vacío. Además, al verificar que la confianza de la regla resultante sea mayor o igual que el umbral de mínima confianza ($minConf$), la confianza es calculada utilizando como frecuencia la Definición 4.1.

Si el conjunto F contuviera los patrones frecuentes descubiertos usando una Booleanización de una la función de semejanza no Booleana, entonces pueden perderse reglas de asociación interesantes y también pueden generase reglas de asociación falsas. A continuación, se demuestra que al minar reglas de asociación usando una Booleanización de una función de semejanza no Booleana pueden perderse reglas de asociación interesantes y generarse falsas reglas de asociación interesantes.

Demostración. Sea f_S una función de semejanza no Booleana y $\bar{\bar{f}}_S$ una Booleanización de f_S, entonces como se demostró en la sección 4.1, para cada $O \in \Omega$ y $S \subseteq R$, puede suceder que $\bar{\bar{f}}_S freq(O) < f_S freq(O)$ o $\bar{\bar{f}}_S freq(O) > f_S freq(O)$.

Enfocándonos sólo en el caso $\bar{\bar{f}}_S freq(O) < f_S freq(O)$, para cada O, S, S', $O \in \Omega$, $S, S' \subseteq R$ pueden darse, entre otras, las siguientes situaciones:

1. $\bar{\bar{f}}_S freq(O) < minFreq \leq f_S freq(O)$ o $\bar{\bar{f}}_{S'} freq(O) < minFreq \leq f_{S'} freq(O)$. Para que una regla de asociación sea interesante tanto la subdescripción antecedente como la subdescripción consecuente deben ser patrones similares frecuentes. Como consecuencia, las reglas interesantes que contengan en el antecedente a la subdescripción $I_S(O)$ o en el consecuente a la subdescripción $I_{S'}(O)$ no serán generadas al emplear como función de semejanza la igualdad.

2. $minFreq \leq \bar{\bar{f}}_S freq(O) < f_S freq(O)$ y $minFreq \leq \bar{\bar{f}}_{S'} freq(O) < f_{S'} freq(O)$. En este caso ambas subdescripciones son similares frecuentes y por tanto la regla $I_S(O) \rightarrow I_{S'}(O)$ es candidata a ser una regla interesante y como consecuencia es el

umbral de mínima confianza $minConf$ el que define si finalmente la regla es o no interesante.

Si denominamos $\bar{\bar{conf}}$ a la confianza de la regla para $\bar{\bar{f}}_{S'}$, a partir de la definición de confianza se tiene,

$$\bar{\bar{conf}}(I_S(O) \rightarrow I_{S'}(O)) = \frac{\bar{\bar{f}}_{\{S \cup S'\}} freq(O)}{\bar{\bar{f}}_S freq(O)} \tag{4.14}$$

y para f_S

$$conf(I_S(O) \rightarrow I_{S'}(O)) = \frac{f_{\{S \cup S'\}} freq(O)}{f_S freq(O)} \tag{4.15}$$

.

Como no existe relación de orden entre $\bar{\bar{conf}}$ y $conf$, entonces pueden darse, entre otros, los siguientes casos:

- $\bar{\bar{conf}}(I_S(O) \rightarrow I_{S'}(O)) < minConf \leq conf(I_S(O) \rightarrow I_{S'}(O))$. En este caso la regla interesante $I_S(O) \rightarrow I_{S'}(O)$ no es generada.
- $conf(I_S(O) \rightarrow I_{S'}(O)) < minConf \leq \bar{\bar{conf}}(I_S(O) \rightarrow I_{S'}(O))$. En este caso es generada una falsa regla de asociación interesante $I_S(O) \rightarrow I_{S'}(O)$.

□

4.7. Síntesis y Conclusiones

En este capítulo fue abordado el problema de encontrar patrones similares frecuentes cuando la función de semejanza es no Booleana. Usar directamente funciones de semejanza no Booleana es importante puesto que transformar las funciones de semejanza no Booleana en funciones de semejanza Booleanas conlleva a la pérdida de patrones similares frecuentes y la generación de falsos patrones similares frecuentes. Además, fue abordado el *problema de las bajas semejanzas y los muchos patrones semejantes* que puede producirse al usar funciones de semejanza no Booleana.

En este capítulo, fueron extendidos los resultados obtenidos en el capítulo anterior, para el uso de funciones de semejanza no Booleana. Para esto, fueron extendidos los conceptos de frecuencia, confianza, patrón similar frecuente, reglas de asociación interesantes, así como propiedades y proposiciones que permiten la poda del espacio de búsqueda de patrones similares frecuentes. También fue extendida la estructura de datos *STree* para permitir el uso de funciones de semejanza no Booleana. Además fueron propuestos tres nuevos algoritmos de minado de patrones similares frecuentes para este tipo de funciones:

- *STree*DC-Miner*. Extensión del algoritmo *STreeDC-Miner*. Está basado en propiedades y proposiciones que permiten la poda del espacio de búsqueda de patrones

similares frecuentes cuando las funciones de semejanza no Booleana son monótonas no crecientes. Para este tipo de funciones *STree*DC-Miner* obtiene todos los patrones similares frecuentes, en otros casos puede perder muchos patrones similares frecuentes debido al mecanismo de poda.

- *STree*NDC-Miner*. Extensión del algoritmo *STreeNDC-Miner*. Realiza un exploración exhaustiva del espacio de búsqueda. No pierde patrones patrones similares frecuentes cuando las funciones de semejanza no Booleana no son monótonas no crecientes.

- *RP*-Miner*. Extensión del algoritmo *RP-Miner*. Relaja el mecanismo de poda de *STree*DC-Miner*. Cuando las funciones de semejanza no Booleana no son monótonas no crecientes, aunque puede perder patrones similares frecuentes, estos son generalmente menos que los que pierde *STree*DC-Miner*.

Finalmente, fue descrito cómo usar el algoritmo de minado de reglas de asociación *FSP-GenRules* para minado de reglas de asociación usando funciones de semejanza no Booleana.

Con estos resultados se cumplen los objetivos particulares 1, 2, 3, 4 y 5 de esta investigación, para funciones de semejanza no Booleana.

Capítulo 5

Resultados Experimentales

En este capítulo se muestran los resultados experimentales obtenidos al evaluar el desempeño de los algoritmos de minado de patrones similares frecuentes propuestos (*STreeDC-Miner*, *STreeNDC-Miner*, *RP-Miner*, *STree* DC-Miner*, *STree*NDC-Miner* y *RP*-Miner*) y una comparación experimental contra otros algoritmos reportados en la literatura. Además son comparadas las reglas de asociación obtenidas a partir de patrones similares frecuentes con las reglas de asociación obtenidas por el enfoque tradicional.

5.1. Descripción general de los experimentos

En la experimentación se comparan, en términos de su eficacia, su eficiencia y la calidad del conjunto de patrones encontrados, los algoritmos de minado de patrones similares frecuentes propuestos en esta tesis (*STreeDC-Miner*,*STreeNDC-Miner*,*RP-Miner*,*STree*DC-Miner*, *STree*NDC-Miner*, y *RP*-Miner*), el único algoritmo para minar patrones frecuentes que usa el concepto de semejanza para comparar las subdescripciones de los objetos (*ObjectMiner*) y el enfoque tradicional de minado de patrones frecuentes en el cual la igualdad es usada para comparar subdescripciones de objetos. Además, se comparan las reglas obtenidas a partir de patrones similares frecuentes con las reglas obtenidas por el enfoque tradicional, en términos de las reglas de asociación interesantes que pueden perderse y de las reglas de asociación falsas que pueden generase.

El problema de minado de patrones similares frecuentes consiste en encontrar todos los patrones similares frecuentes. Como consecuencia la eficacia de cada algoritmo es medida a través del número de patrones similares frecuentes encontrados.

La eficiencia es medida como el tiempo requerido para encontrar los patrones similares frecuentes. Otras medidas como el número de evaluaciones de la función de semejanza, el número de candidatos, así como la proporción entre el número de patrones similares frecuentes y el tiempo de ejecución son también usadas.

Por otro lado, obtener un conjunto grande de patrones frecuentes no necesariamente implica que este conjunto sea mejor. Por esta razón es necesario medir la calidad de los

conjuntos de patrones similares frecuentes obtenidos por cada algoritmo. En esta tesis, la calidad de los patrones similares frecuentes encontrados se mide como la precisión que alcanza un clasificador supervisado basado en este conjunto de patrones cuando nuevos objetos son clasificados. La precisión de un clasificador es el porciento de objetos correctamente clasificados del total de objetos clasificados.

Los algoritmos propuestos en esta tesis fueron implementados en el lenguaje de programación *Java*. Por otro lado los autores de *ObjectMiner* realizaron su implementación en el lenguaje de programación *Phyton*. Para poder hacer comparaciones respecto al tiempo de ejecución, en condiciones similares, el algoritmo *ObjectMiner* fue implementado por nosotros en *Java*. Es importante apuntar que dicha implementación fue realizada cuidadosamente y que los tiempos de ejecución de la misma fueron menores que los de la implementación de los autores en *Phyton*.

Todos los experimentos fueron realizados en una PC con procesador Core 2 Quad de 2,6GHz con 4GB de RAM usando Linux-Debian a 64-bits. Se utilizó la máquina virtual de *Java* de IBM versión 6. El espacio de memoria disponible fue fijado en 4GB de RAM para eliminar la influencia de las operaciones de paginado.

La experimentación se dividió en 2 partes. Las secciones 5.2 y 5.3 están dedicadas a comparar los algoritmos de minado de patrones similares frecuentes para funciones de semejanza Booleana y no Booleana respectivamente. La sección 5.4 está dedicada a comparar las reglas obtenidas a partir de los patrones similares frecuentes, con las reglas obtenidas por el enfoque tradicional.

5.2. Experimentos de minado de patrones similares frecuentes con función de semejanza Booleana

En la sección 5.2.1 se presentan los experimentos realizados con el algoritmo propuesto para el minado de patrones similares frecuentes que usa una función de semejanza Booleana que cumple la propiedad de f_S-Clausura Descendente. Se compara el algoritmo propuesto contra *ObjectMiner* y el enfoque tradicional de minado de patrones similares frecuentes. En la sección 5.2.2 se presenta una experimentación análoga a la presentada en la sección 5.2.1 pero con los algoritmos propuestos para el minado de patrones similares frecuentes que usan una función de semejanza Booleana que no cumple la propiedad de f_S-Clausura Descendente.

En los experimentos se utilizaron las colecciones de datos mostradas en la tabla 5.1. Las mismas han sido utilizadas anteriormente[1] para tareas de minería de datos como son clasificación, agrupamiento y minería de patrones frecuentes y reglas de asociación.

[1]UCI Machine learning Repository. `http://archive.ics.uci.edu/ml/`

Tabla 5.1: Descripción de las colecciones de datos usadas en los experimentos con funciones de semejanza Booleana.

Colección de Datos	Objetos	Atributos Numéricos	Atributos No Numéricos	Clases
Car Evaluation	1728	2	5	4
Contraceptive Method Choice	1473	2	8	3
Census	32561	6	9	2
Poker Hand	1000000	0	11	10

5.2.1. Experimentos con los algoritmos propuestos para funciones de semejanza Booleana que cumplen la propiedad de f_S-Clausura Descendente

Como función de semejanza Booleana que cumple la propiedad de f_S-Clausura Descendente usamos la función (5.1) con los criterios de comparación (5.2) y (5.3). El criterio de comparación (5.2) fue usado para los atributos numéricos *Age* con $\varepsilon = 5$, *Doors* con $\varepsilon = 2$ y *Persons* con $\varepsilon = 2$ de la colección *Car Evaluation*; para el atributo numérico *Age* con $\varepsilon = 5$ para la colección *Contraceptive Method Choice*; y para los atributos numéricos *Age* con $\varepsilon = 5$, *Capital gain* con $\varepsilon = 1000$ y *Capital loss* con $\varepsilon = 1000$ para la colección *Census*. Estos valores de ε fueron tomados a priori. El criterio de comparación (5.3) fue usado para los restantes atributos numéricos y para los atributos no numéricos.

$$f_S(O, O') = \begin{cases} 1 & \text{si} \forall r \in S \; C_r(O[r], O'[r]) = 1 \\ 0 & \text{en otro caso} \end{cases} \tag{5.1}$$

$$C_r(x, y) = \begin{cases} 1 & \text{si } |x - y| \leq \varepsilon \\ 0 & \text{en otro caso} \end{cases} \tag{5.2}$$

$$C_r(x, y) = \begin{cases} 1 & \text{si } x = y \\ 0 & \text{en otro caso} \end{cases} \tag{5.3}$$

Eficiencia de los algoritmos

La figura 5.1 muestra los tiempos de ejecución de los algoritmos *STreeDC-Miner* y *ObjectMiner*, mientras la figura 5.2 muestra el número de evaluaciones de la función de semejanza realizadas por los mismos. En ambas figuras, los umbrales de mínima frecuencia son variados entre 0,02 y 0,16 para cada colección de datos.

En la figura 5.1 puede apreciarse que el algoritmo *STreeDC-Miner* en todos los casos logra mejores tiempos que *ObjectMiner*. Estos resultados se deben a la poda que realiza *STreeDC-Miner*, gracias a la cual, el número de evaluaciones de la función de semejanza realizadas es mucho menor que en *ObjectMiner* (véase figura 5.2).

Las mayores diferencias en cuanto al tiempo de ejecución de *STreeDC-Miner* y *ObjectMiner* se observan para los valores más pequeños de *minFreq*. Para estos valores,

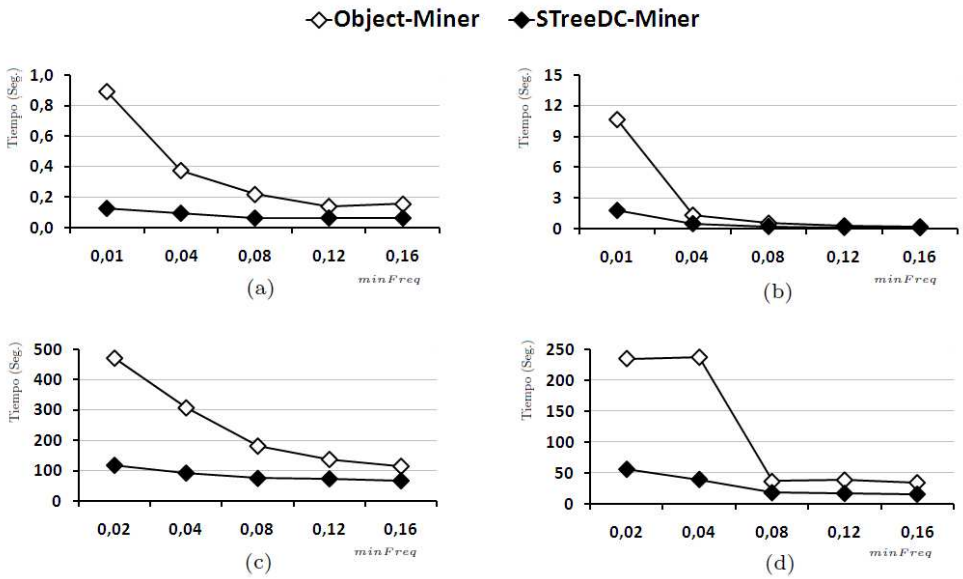


Figura 5.1: Tiempos de ejecución de *STreeDC-Miner* y *ObjectMiner* para la función de semejanza Booleana (5.1) que cumple la propiedad de f_S-Clausura Descendente en las colecciones de datos (a) *Car Evaluation*, (b) *Contraceptive Method Choice*, (c) *Census* y (d) *Poker Hand*.

STreeDC-Miner logra tiempos de ejecución de hasta 7,1, 5,9, 3,9 y 4,2 veces mejores que los tiempos de ejecución de *ObjectMiner* para las respectivas colecciones de datos.

Calidad de los patrones minados

En contextos donde los objetos son comparados mediante una función de semejanza Booleana diferente de la igualdad, si al minar patrones frecuentes, se usa la igualdad, como en el enfoque tradicional de minado de patrones frecuentes, pueden perderse patrones.

Para evaluar cuán útiles pueden ser los patrones perdidos cuando se usa la igualdad como función de semejanza para comparar subdescripciones de objetos, en nuestros experimentos obtendremos el conjunto de patrones frecuentes usando tanto una función de semejanza diferente de la igualdad (5.1) como usando la igualdad y mediremos la calidad de dichos conjuntos.

Para medir la calidad, se usó un clasificador, el cual en la fase de entrenamiento, para cada clase son obtenidos los patrones similares frecuentes y eliminados aquellos patrones que aparecen en otra clase con una mayor frecuencia. En la fase de clasificación, cada objeto de la colección de prueba es clasificado en la clase donde existen más patrones similares a sus subdescripciones. Por cada colección y umbral de mínima frecuencia, se repite el experimento 10 veces, seleccionando aleatoriamente de cada clase, el 50 % del

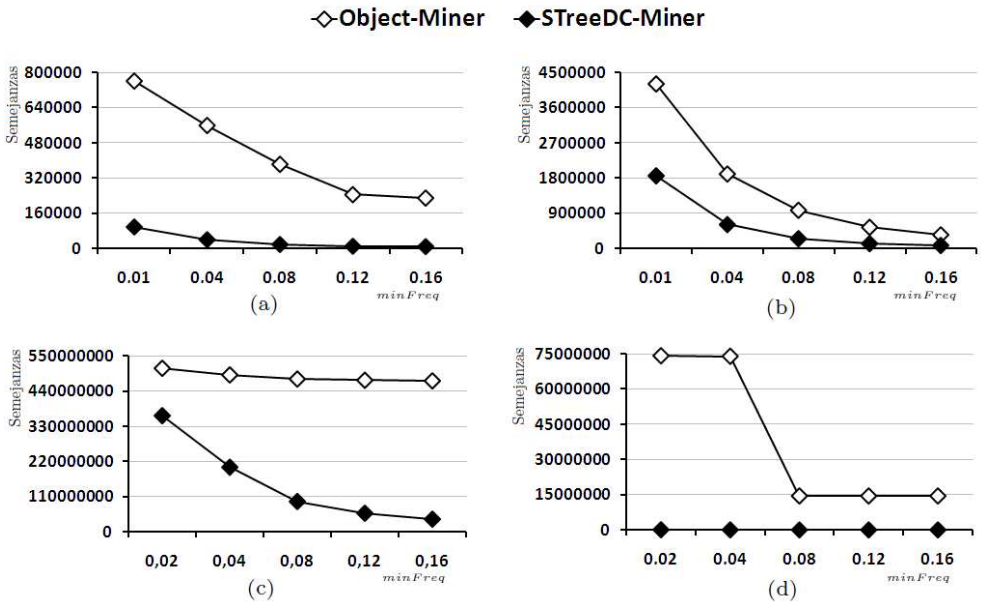


Figura 5.2: Número de evaluaciones de la función de semejanza realizadas por *STreeDC-Miner* y *ObjectMiner* para la función de semejanza Booleana (5.1) que cumple la propiedad de f_S-Clausura Descendente en las colecciones de datos (a) *Car Evaluation*, (b) *Contraceptive Method Choice*, (c) *Census* y (d) *Poker Hand*.

total de objetos de la clase con menos objetos, para el entrenamiento; y usando el resto de los objetos para clasificar. De esta forma, en la colección de entrenamiento el número de objetos por clase es igual y se maximiza el tamaño de la misma, garantizando que el número de objetos por clase usados en la fase de clasificación siempre es mayor o igual que el número de objetos por clase usados en el entrenamiento.

En la tabla 5.2 son mostradas las calidades logradas en las colecciones de datos *Car Evaluation*, *Contraceptive Method Choice* y *Census*. Dado que la función de semejanza (5.1) satisface la propiedad de f_S-Clausura Descendente los algoritmos *STreeDC-Miner* y *ObjectMiner* obtienen el mismo conjunto de patrones similares frecuentes (todos los patrones similares frecuentes). La columna *STreeDC-Miner* de la tabla 5.2 contiene la calidad de los conjuntos de patrones frecuentes obtenidos usando la función de semejanza diferente de la igualdad y la columna *Enfoque Tradicional* contiene la calidad de los conjuntos de patrones frecuentes obtenidos usando como función de semejanza la igualdad. La colección de datos *Poker Hand* no fue incluida en tabla 5.2, puesto que al ser todos sus atributos no numéricos, entonces la función de semejanza Booleana (5.1) resulta la igualdad, y por tanto los resultados para *STreeDC-Miner* y para el *Enfoque Tradicional* son idénticos.

Como puede apreciarse en la tabla 5.2, para la mayoría de los umbrales $minFreq$, la calidad de los patrones similares frecuentes obtenidos por *STreeDC-Miner* usando

Tabla 5.2: Calidad de los conjuntos de patrones similares frecuentes encontrados por *STreeDC-Miner* y *Enfoque Tradicional* en la colecciones de datos *Car Evaluation*, *Contraceptive Method Choice* y *Census*.

Colección de Datos	*minFreq*	*STreeDC-Miner*	*Enfoque Tradicional*
Car Evaluation	0,01	**80.49**	80,44
	0,04	76,29	75,60
	0,08	69,75	70,28
	0,12	65,50	65,43
	0,16	56,49	55,65
Precisión Promedio		69,70	69,48
Contraceptive Method Choice	0,01	**45.69**	41,75
	0,04	40,20	40,65
	0,08	37,46	37,66
	0,12	36,17	33,76
	0,16	35,64	29,65
Precisión Promedio		39,03	36,69
Census	0,02	74,66	73,06
	0,04	**76.00**	72,39
	0,08	**76.00**	71,26
	0,12	72,66	70,80
	0,16	73,33	70,93
Precisión Promedio		74,53	71,68

la función de semejanza diferente de la igualdad es mayor o igual a la calidad de los patrones frecuentes usando la igualdad (*Enfoque Tradicional*). Esto evidencia la utilidad de los patrones que se pierden al emplear el enfoque tradicional de minado de patrones frecuentes.

5.2.2. Experimentos con los algoritmos propuestos para funciones de semejanza Booleana que no cumplen la propiedad de f_S-Clausura Descendente

Cuando la función de semejanza no satisface la propiedad de f_S-Clausura Descendente tanto el algoritmo *ObjectMiner* como el algoritmo *STreeDC-Miner*, cuyos mecanismos de poda suponen el cumplimiento de esta propiedad, pierden patrones similares frecuentes. Por otro lado el algoritmo *STreeNDC-Miner*, propuesto para este tipo de funciones, no pierde patrones similares frecuentes, pero recorre exhaustivamente el espacio de búsqueda lo cual afecta su eficiencia, mientras el algoritmo *RP-Miner*, también propuesto para este tipo de funciones, con su poda relajada pierde menos patrones similares frecuentes que *ObjectMiner* y *STreeDC-Miner* y es más eficiente que *STreeNDC-Miner*.

A continuación, los algoritmos *STreeNDC-Miner*, *RP-Miner* son comparados en cuanto a su eficiencia y eficacia contra los *ObjectMiner*, *STreeDC-Miner*, cuando se usa una función de semejanza que no cumple la propiedad de f_S-Clausura Descendente. Además, la calidad del conjunto de patrones similares frecuentes obtenidos por cada uno, también es comparada.

Eficiencia y eficacia de los algoritmos

Como función de semejanza Booleana que no cumple la propiedad de f_S-Clausura Descendente usamos la función (5.4) con $\gamma = 0{,}7$ y los criterios de comparación (5.2) y

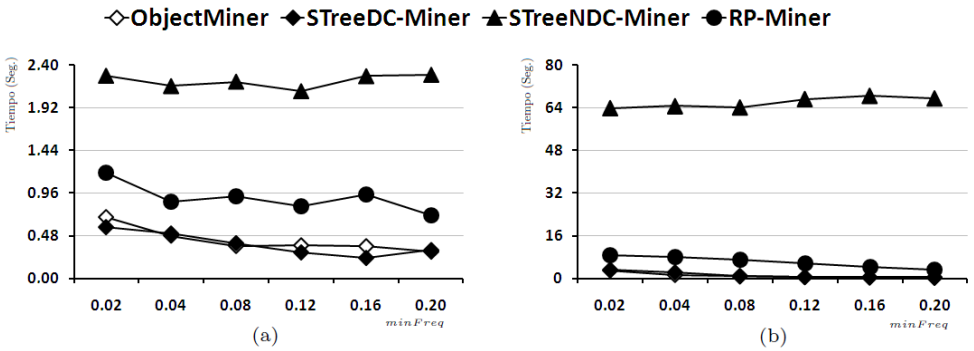


Figura 5.3: Tiempos de ejecución de *STreeDC-Miner*, *ObjectMiner*, *RP-Miner* y *STreeNDC-Miner* para la función de semejanza Booleana (5.4) que no cumple la propiedad de f_S-Clausura Descendente en las colecciones de datos (a) *Car Evaluation* y (b) *Contraceptive Method Choice*.

(5.3) igual que en la sección anterior.

$$f_S(O, O') = \begin{cases} 1 & \text{si } \dfrac{\sum_{r \in S} C_r(O[r], O'[r])}{|S|} \geq \gamma \\ 0 & \text{otro caso} \end{cases} \tag{5.4}$$

En la figura 5.3 son mostrados los tiempos de ejecución de los algoritmos para las colecciones *Car Evaluation* y *Contraceptive Method Choice*.

En ambas colecciones el tiempo de ejecución de *RP-Miner* fue mayor que el tiempo de ejecución de *STreeDC-Miner* y *ObjectMiner*; y el tiempo de ejecución de *STreeNDC-Miner* fue el mayor de todos. Esto es una consecuencia del número de patrones similares frecuentes encontrados (véase figura 5.5) y del número de evaluaciones de la función de semejanza (véase figura 5.4).

El número de patrones similares frecuentes encontrados para *Car Evaluation* y *Contraceptive Method Choice* para cada $minfreq$ es mostrado en la figura 5.5. Es importante subrayar que *STreeNDC-Miner* encuentra todos los patrones similares frecuentes, mientras *ObjectMiner* y *STreeDC-Miner*, los cuales suponen que la función de semejanza cumple la propiedad de f_S-Clausura Descendente, pueden no encontrar todos los patrones similares frecuentes. Por su parte, *RP-Miner* también puede no encontrar todos los patrones similares frecuentes, pero el uso de la poda relajada le permite encontrar patrones similares frecuentes que *ObjectMiner* y *STreeDC-Miner* pierden.

Nótese que en *Car Evaluation* para $minFreq = 0{,}02$, respecto al número de patrones similares frecuentes obtenidos por *STreeNDC-Miner*, *ObjectMiner* pierde hasta 14 168 (70,64 %) patrones similares frecuentes y *STreeDC-Miner* pierde hasta 3 805 (18,97 %) patrones similares frecuentes, mientras *RP-Miner* pierde menos patrones similares frecuentes (3 279, que presentan el 16,35 %) (véase Figura 5.5).

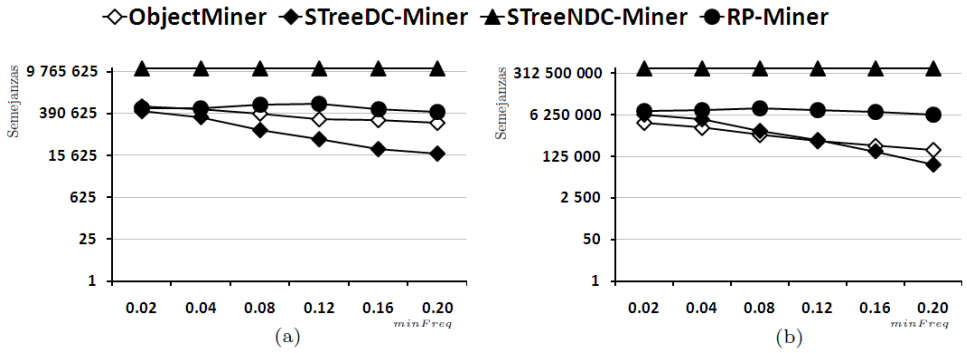


Figura 5.4: Número de evaluaciones de la función de semejanza realizadas por *STreeDC-Miner*, *ObjectMiner*, *RP-Miner* y *STreeNDC-Miner* para función de semejanza Booleana (5.4) que no cumple la propiedad de f_S-Clausura Descendente en las colecciones de datos (a) *Car Evaluation* y (b) *Contraceptive Method Choice*.

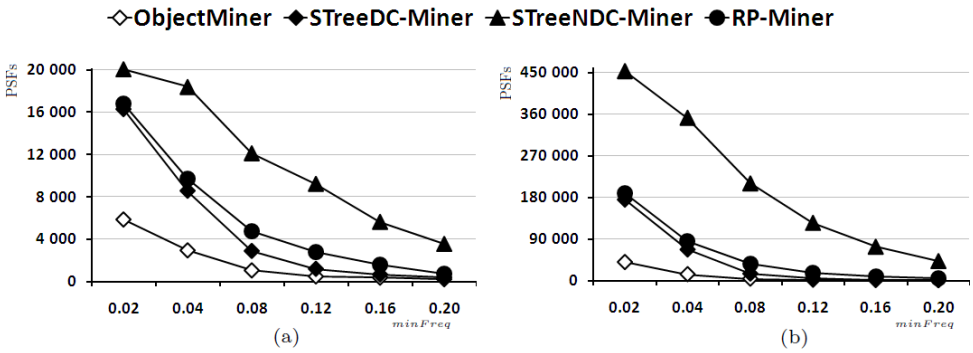


Figura 5.5: Número de patrones similares frecuentes encontrados por *STreeDC-Miner*, *ObjectMiner*, *RP-Miner* y *STreeNDC-Miner* para la función de semejanza Booleana (5.4) que no cumple la propiedad de f_S-Clausura Descendente en las colecciones de datos (a) *Car Evaluation* y (b) *Contraceptive Method Choice*.

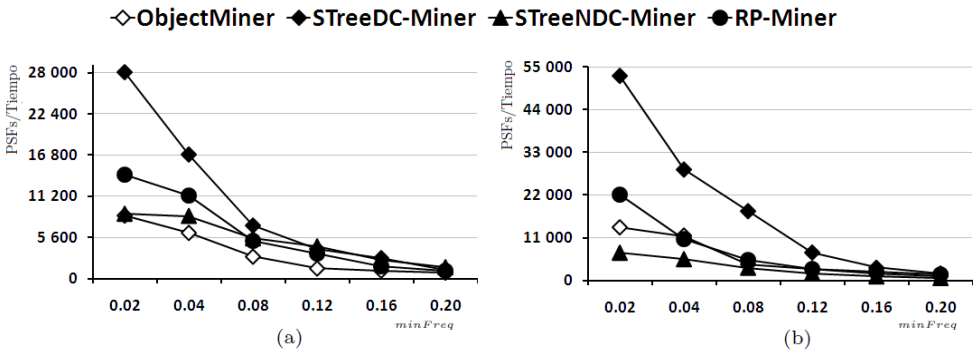


Figura 5.6: Proporción entre el número de patrones similares frecuentes encontrados y el tiempo de ejecución de *STreeDC-Miner*, *ObjectMiner*, *RP-Miner* y *STreeNDC-Miner* para la función de semejanza Booleana (5.4) que no cumple la propiedad de f_S-Clausura Descendente en las colecciones de datos (a) *Car Evaluation* y (b) *Contraceptive Method Choice*.

Análogamente, en *Contraceptive Method Choice* para $minFreq = 0{,}02$, *ObjectMiner* pierde hasta 412 977 (91,27 %) patrones similares frecuentes y *STreeDC-Miner* pierde hasta 278 031 (61,45 %) patrones similares frecuentes, mientras *RP-Miner* pierde menos patrones similares frecuentes (263 197, que presentan el 58,17 %) (véase figura 5.5).

Para los otros valores de $minFreq$, tanto en *Car Evaluation* como en *Contraceptive Method Choice*, el número de patrones similares frecuentes perdidos por los algoritmos es menor, no obstante, *RP-Miner* siempre pierde menos patrones similares frecuentes que *ObjectMiner* y *STreeDC-Miner*.

Respecto a la proporción entre el número de patrones similares frecuentes y el tiempo de ejecución, en ambas colecciones, los mejores resultados fueron obtenidos por *STreeDC-Miner* (hasta en 3,1 y 7,4 veces superior a *STreeNDC-Miner* respectivamente para $minFreq = 0{,}02$), seguido por *RP-Miner* (hasta en 1,6 y 3,1 veces superior a *STreeNDC-Miner* respectivamente para $minFreq = 0{,}02$) (véase figura 5.6). Un elemento relevante es que *RP-Miner*, con su poda relajada, obtiene en la mayoría de los casos más patrones similares frecuentes por unidad de tiempo que *ObjectMiner* con su poda estricta.

El número de patrones similares frecuentes encontrados para las colecciones de datos *Poker Hand* y *Census* son mostradas en las figura 5.7. Estas colecciones de datos contienen muchos objetos más que *Car Evaluation* y *Contraceptive Method Choice*. Aunque *STreeNDC-Miner* es el algoritmo más efectivo (encuentra todos los patrones frecuentes), también es muy lento debido a que realiza una búsqueda exhaustiva de los patrones similares frecuentes. Es importante señalar que para las colecciones *Poker Hand* y *Census* usando $minFreq = 0{,}02$, *STreeNDC-Miner* no pudo terminar de correr después de 10 días. Por esta razón, en este experimento, los resultados de *STreeNDC-Miner* no son comparados con los resultados de los otros algoritmos.

Como en las anteriores colecciones, en *Poker Hand* y *Census*, el tiempo de ejecución de

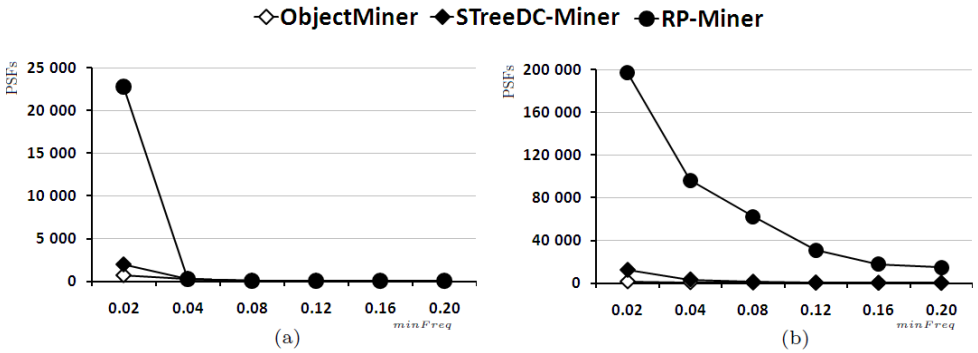


Figura 5.7: Número de patrones similares frecuentes encontrados por *STreeDC-Miner*, *ObjectMiner* y *RP-Miner* para la función de semejanza Booleana (5.4) que no cumple la propiedad de f_S-Clausura Descendente en las colecciones de datos (a) *Poker Hand* y (b) *Census*.

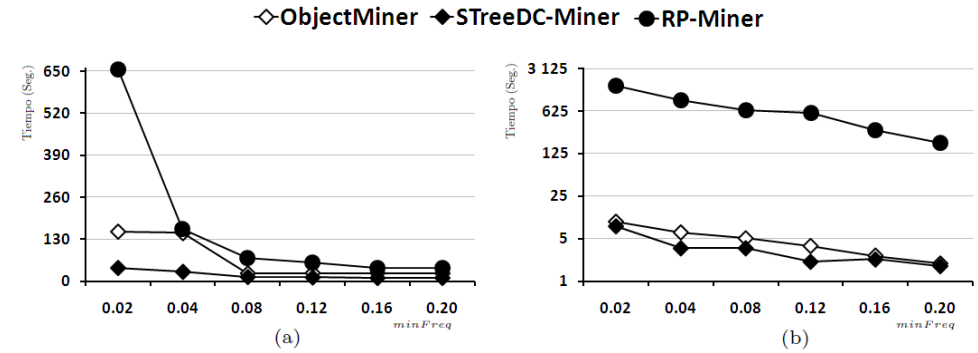


Figura 5.8: Tiempo de ejecución de *STreeDC-Miner*, *ObjectMiner* y *RP-Miner* para la función de semejanza Booleana (5.4) que no cumple la propiedad de f_S-Clausura Descendente en las colecciones de datos (a) *Poker Hand* y (b) *Census*.

RP-Miner fue mayor que el tiempo de ejecución de *STreeDC-Miner* y *ObjectMiner* (véase figura 5.8). Esto se debe a que el número de patrones similares frecuentes encontrados por *RP-Miner* es mayor que el número de patrones similares frecuentes encontrados por *STreeDC-Miner* y *ObjectMiner* (véase figura 5.7); y a que en varios casos, el número de evaluaciones de la función de semejanza realizadas por *RP-Miner* es mayor que el número de evaluaciones de la función de semejanza realizadas por *STreeDC-Miner* y *ObjectMiner* (véase figura 5.9).

En estas colecciones el número de patrones similares frecuentes (véase figura 5.7) obtenidos por *RP-Miner* y perdidos por *ObjectMiner* y *STreeDC-Miner* es mucho mayor que en las colecciones anteriores.

Puede notarse que *ObjectMiner* y *STreeDC-Miner* pierden más patrones similares frecuentes comparados con los obtenidos por *RP-Miner* tanto en *Poker Hand* como en

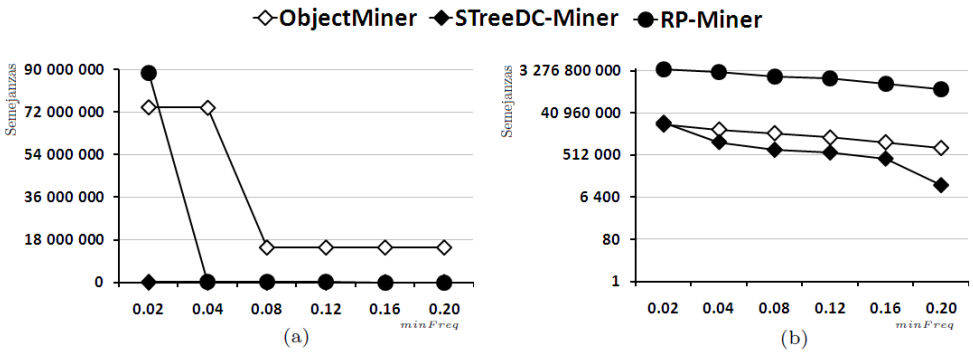


Figura 5.9: Número de evaluaciones de la función de semejanza realizadas por *STreeDC-Miner*, *ObjectMiner* y *RP-Miner* para función de semejanza Booleana (5.4) que no cumple la propiedad de f_S-Clausura Descendente en las colecciones de datos (a) *Poker Hand* y (b) *Census*.

Census para $minFreq = 0{,}02$; y que *ObjectMiner* es el que más patrones similares frecuentes pierde.

Para los demás valores de $minFreq$, en ambas colecciones, el número de patrones similares frecuentes encontrados por los algoritmos fue similar. Sin embargo, *RP-Miner* siempre pierde menos o igual número de patrones similares frecuentes que *ObjectMiner* y *STreeDC-Miner*.

Adicionalmente, respecto a la proporción entre el número de patrones similares frecuentes y el tiempo de ejecución, para *Poker Hand* los mejores resultados fueron obtenidos por *STreeDC-Miner*, seguido por *RP-Miner*; mientras que para *Census* los mejores resultados también fueron obtenidos por *STreeDC-Miner*, aunque seguido por *ObjectMiner* (véase figura 5.10). No obstante, para *Census* apenas se diferencian los resultados obtenidos por *ObjectMiner* y *RP-Miner*.

Por otro lado, se puede observar que para todas las colecciones, cuando $minFreq$ crece, el número de patrones similares frecuentes obtenidos por todos los algoritmos decrece, o se mantiene igual (véase figuras 5.5 y 5.7).

Además, pequeños valores de $minFreq$ favorecen a *RP-Miner* en cuanto a la eficacia comparado contra *ObjectMiner* y *STreeDC-Miner*; y favorecen a *STreeDC-Miner* en cuanto a la eficiencia comparado contra *ObjectMiner* y *RP-Miner*.

Adicionalmente, cuando el valor de $minFreq$ crece, para *ObjectMiner*, *STreeDC-Miner* y *RP-Miner*, debido a la poda, el número de evaluaciones de la función de semejanza tiende a decrecer (véase Figuras 5.4 y 5.9); mientras para *STreeNDC-Miner*, debido a la búsqueda exhaustiva, el número de evaluaciones de la función de semejanza tiende a mantenerse similar (véase figura 5.4).

Basado en estos experimentos, cuando la función de semejanza no cumple la propiedad de f_S-Clausura Descendente se puede afirmar que:

- En problemas donde sea necesario encontrar todos los patrones similares frecuentes

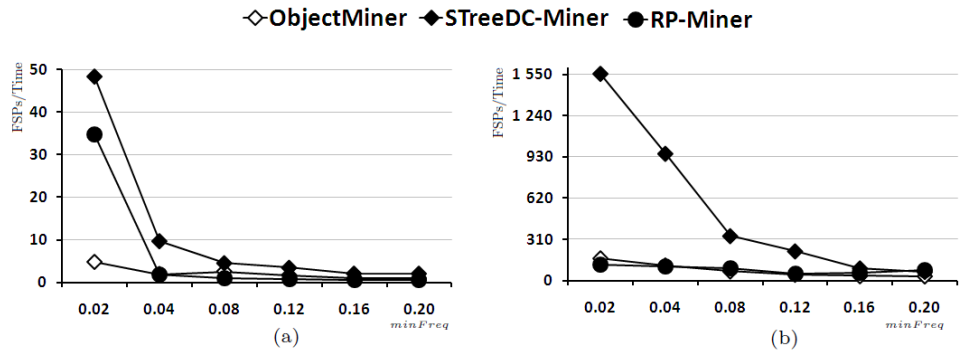


Figura 5.10: Proporción entre el número de patrones similares frecuentes y el tiempo de ejecución de *STreeDC-Miner*, *ObjectMiner* y *RP-Miner* para la función de semejanza Booleana (5.4) que no cumple la propiedad de f_S-Clausura Descendente en las colecciones de datos (a) *Poker Hand* y (b) *Census*.

y las colecciones contengan hasta 2000 objetos y 10 atributos, el algoritmo *STreeNDC-Miner* es una buena alternativa.

- En problemas donde sea necesario encontrar la mayor cantidad de patrones similares frecuentes posibles (idealmente todos), las colecciones contengan mas de 2000 objetos y 10 atributos y el tiempo de ejecución del algoritmo *STreeNDC-Miner* no sea aceptable, el algoritmo *RP-Miner* es una alternativa entre *STreeNDC-Miner* y *STreeDC-Miner* pues es más eficiente que *STreeNDC-Miner* y más eficaz que *STreeDC-Miner*.

Calidad de los patrones minados

Obtener un conjunto grande de patrones similares frecuentes, no necesariamente implica que este conjunto sea mejor. Por esta razón, la calidad del conjunto de patrones similares frecuentes obtenidos por cada algoritmo es comparada usando un clasificador supervisado basado en patrones, tal y como se describió en la sección 5.2.1.

Dado que *STreeNDC-Miner* require de más de 10 días para encontrar los patrones similares frecuente en las colecciones *Poker Hand* y *Census*, la calidad de los patrones similares frecuentes obtenidos por *STreeNDC-Miner* no es evaluada para estas colecciones.

Como función de semejanza Booleana que no cumple la propiedad de f_S-Clausura Descendente usamos la función (5.4) con $\gamma = 0{,}7$ y los criterios de comparación (5.2) para los atributos numéricos y (5.3) para los atributos no numéricos. Para cada atributo numérico r, el parámetro ε de su criterio de comparación es fijado en $\frac{Max_r - Min_r}{10}$, donde $Max_r = \max_{O\in\Omega} I_{\{r\}}(O)$, $Min_r = \min_{O\in\Omega} I_{\{r\}}(O)$.

Por cada colección y algoritmo, la clasificación es realizada para diferentes valores del umbral $minFreq$ a partir de $minFreq = 0{,}10$ y hasta $minFreq = 0{,}20$ con incrementos de 0,01. En la tabla 5.3, son mostradas las calidades obtenidas.

Tabla 5.3: Calidad de los conjuntos de patrones similares frecuentes encontrados por *ObjectMiner*, *STreeDC-Miner*, *STreeNDC-Miner*, *RP-Miner* y *Enfoque Tradicional* en la colecciones de datos *Car Evaluation*, *Contraceptive Method Choice*, *Poker Hand* y *Census*.

Colección de Datos	*minFreq*	*ObjectMiner*	*STreeDC-Miner*	*STreeNDC-Miner*	*RP-Miner*	*Enfoque Tradicional*
Car Evaluation	0,10	64,86	65,96	51,01	**66.27**	65,43
	0,11	64,86	65,96	51,01	**66.27**	65,43
	0,12	64,86	65,96	51,01	**66.27**	65,43
	0,13	60,52	61,01	38,91	60,94	60,14
	0,14	60,52	61,01	38,91	60,94	60,14
	0,15	60,52	61,01	38,91	60,94	60,14
	0,16	55,59	56,08	31,31	56,23	55,65
	0,17	55,59	56,08	31,31	56,23	55,65
	0,18	55,59	56,08	31,31	56,23	55,65
	0,19	55,39	55,77	28,61	55,71	55,39
	0,20	55,39	55,77	28,61	55,71	55,39
Precisión Promedio		59, 43	60, 06	38, 26	60, 16	59, 49
Contraceptive Method Choice	0,10	41,12	40,93	38,41	41,01	35,15
	0,11	41,36	41,29	37,19	40,84	34,79
	0,12	41,05	40,99	37,27	**41.49**	33,76
	0,13	41,02	40,69	35,70	39,98	33,23
	0,14	40,96	40,73	35,03	39,94	30,96
	0,15	40,67	40,67	34,57	40,37	30,32
	0,16	40,76	40,34	34,08	40,25	29,65
	0,17	39,74	39,14	33,33	39,92	28,34
	0,18	40,36	39,58	33,02	40,27	28,63
	0,19	40,48	39,04	32,14	40,49	30,28
	0,20	41,03	39,41	32,14	40,37	30,23
Precisión Promedio		40, 78	40, 26	34, 81	40, 45	31, 39
Poker Hand	0,10	9,40	9,40	–	**19.30**	9,40
	0,11	8,18	8,18	–	18,43	8,18
	0,12	7,35	7,35	–	15,76	7,35
	0,13	8,91	8,91	–	14,97	8,91
	0,14	9,97	9,97	–	14,60	9,97
	0,15	12,79	12,79	–	14,33	12,79
	0,16	13,66	13,66	–	14,31	13,66
	0,17	14,16	14,16	–	14,28	14,16
	0,18	13,96	13,96	–	13,96	13,96
	0,19	13,96	13,96	–	13,96	13,96
	0,20	12,18	12,18	–	12,18	12,18
Precisión Promedio		11, 32	11, 32	–	15, 10	11, 32
Census	0,10	70, 20	72, 13	–	**73,47**	70, 80
	0,11	70, 07	71, 33	–	73, 07	71, 40
	0,12	70, 07	71, 33	–	73, 07	71, 40
	0,13	69, 60	70, 80	–	72, 67	70, 87
	0,14	68, 47	70, 07	–	72, 60	70, 53
	0,15	66, 87	69, 73	–	72, 00	69, 80
	0,16	66, 87	69, 73	–	72, 00	69, 80
	0,17	66, 67	68, 07	–	71, 67	69, 60
	0,18	65, 00	67, 07	–	70, 40	69, 47
	0,19	63, 60	66, 40	–	70, 13	70, 87
	0,20	63, 60	66, 40	–	70, 13	70, 87
Precisión Promedio		67, 37	69, 37	–	71, 93	70, 49

La última columna en la tabla 5.3 (*Enfoque Tradicional*) contiene los resultados del algoritmo *STreeDC-Miner* usando la igualdad como función de semejanza. A partir de esta columna, se puede observar que usando la función igualdad, como en el enfoque tradicional de minado de patrones frecuentes, la precisión de la clasificación obtenida por el conjunto de patrones frecuentes es la mayoría de las veces menor que la precisión de la clasificación obtenida por los conjuntos de patrones similares frecuentes obtenidos utilizando una función de semejanza diferente de la igualdad.

En estos experimentos, en general, la precisión de la clasificación alcanzada mediante los patrones similares frecuentes encontrados por *RP-Miner* es mejor que la precisión de la clasificación alcanzada por *ObjectMiner*, *STreeDC-Miner* y el enfoque tradicional. De hecho, las mejores precisiones (celdas grises en la tabla 5.3) para cada colección, fue alcanzada por *RP-Miner*.

Nótese que el conjunto de patrones similares frecuentes encontrados por *RP-Miner* es un superconjunto del conjunto de los patrones similares frecuentes encontrados por *ObjectMiner* y *STreeDC-Miner*. Por esta razón se puede afirmar que los patrones similares frecuentes perdidos por *ObjectMiner* y *STreeDC-Miner* afectan la clasificación, mientras que aquellos otros patrones adicionales encontrados por *RP-Miner* contribuyen a alcanzar una mejor precisión.

Además es importante señalar que a pesar de ser el conjunto de patrones similares frecuentes encontrados por *STreeNDC-Miner* un superconjunto del conjunto de los patrones similares frecuentes encontrados por *RP-Miner*, la calidad de los patrones similares frecuentes obtenidos por *STreeNDC-Miner* es inferior a la calidad de los patrones similares frecuentes obtenidos por *RP-Miner*. Esto evidencia que los patrones adicionales encontrados por *STreeNDC-Miner* afectan la clasificación. Los patrones similares frecuentes son obtenidos por *RP-Miner* mediante un proceso de expansión de los patrones con un único atributo, en el cual la poda rebajada es usada como mecanismo de contención, mientras que *STreeNDC-Miner* obtiene los patrones frecuentes explorando exhaustivamente el espacio de búsqueda. Por tanto, los patrones adicionales encontrados por *STreeNDC-Miner* son más largos que los encontrados por *RP-Miner*. Los patrones largos son más específicos que patrones más cortos y cubren menos objetos que estos últimos. Como consecuencia, al usar los patrones adicionales encontrados por *STreeNDC-Miner*, el clasificador es sobreentrenado y pierde capacidad de generalización.

Basándose en estos experimentos, cuando la función de semejanza no cumple la propiedad de f_S-Clausura Descendente se puede afirmar que:

- Encontrar más patrones similares frecuentes, incluso todos, no es siempre la mejor opción.
- Mediante el algoritmo *RP-Miner* pueden encontrarse conjuntos de patrones con mayor calidad, al menos para la tarea de clasificación, que con los algoritmos *ObjectMiner*, *STreeDC-Miner* y *STreeNDC-Miner*.
- La calidad de los patrones similares frecuentes es generalmente mejor que la calidad

Tabla 5.4: Descripción de las colecciones de datos usadas en los experimentos con funciones de semejanza no Booleana.

Colección de Datos	Objetos	Atributos Numéricos	Atributos No Numéricos	Clases
Iris	150	4	1	3
Diabetes	768	8	1	2
Liver Disorders	345	6	1	2
Page Blocks	5473	10	1	2

de los patrones frecuentes del enfoque tradicional.

5.3. Experimentos de minado de patrones similares frecuentes con función de semejanza no Booleana

En la sección 5.3.1 se presentan los experimentos realizados con el algoritmo propuesto para el minado de patrones similares frecuentes que usa una función de semejanza no Booleana que cumple la propiedad de f_S-Clausura Descendente. Se compara el algoritmo propuesto contra *STreeDC-Miner* que usa una función de semejanza Booleana) y contra el enfoque tradicional de minado de patrones similares frecuentes. En la sección 5.3.2 se presenta una experimentación análoga a la presentada en la sección 5.3.1 pero con los algoritmos propuestos para el minado de patrones similares frecuentes que usan una función de semejanza no Booleana que no cumple la propiedad de f_S-Clausura Descendente. La sección 5.3.3 está dedicada a abordar el problema de las bajas semejanzas y los muchos patrones frecuentes.

En los experimentos se utilizaron las colecciones de datos mostradas en la tabla 5.4. Las mismas, al igual que las colecciones usadas en la sección 5.2, han sido utilizadas anteriormente [2] para tareas de minería de datos como son clasificación, agrupamiento y minería de patrones frecuentes y reglas de asociación, pero a diferencia de ellas contienen un mayor número de atributos numéricos. Con ello se facilita el uso de criterios de comparación no Booleanos y funciones de semejanza no Booleana.

5.3.1. Experimentos con los algoritmos propuestos para funciones de semejanza no Booleana que cumplen la propiedad de f_S-Clausura Descendente

Como función de semejanza no Booleana que cumple la propiedad de f_S-Clausura Descendente usamos la función (5.5) con los criterios de comparación (5.3) para los atributos no numéricos y (5.6) para los atributos numéricos.

[2]UCI Machine learning Repository. `http://archive.ics.uci.edu/ml/`

$$f_S(O, O') = \prod_{r \in S} C_r(O[r], O'[r]) \tag{5.5}$$

$$C_r(x, y) = 1 - \frac{|x - y|}{Max_r - Min_r} \tag{5.6}$$

Como función de semejanza Booleana tomamos la misma función (5.5) pero con el criterio de comparación (5.7) con $\alpha = 0{,}9$. Nótese que el criterio de comparación (5.7) usado en la función de semejanza Booleana es una Booleanización de (5.6). Además, como umbral de mínima semejanza tomamos $\beta = 0$, es decir, no atacamos el problema de los muchos patrones y las bajas frecuencias.

$$C_r(x, y) = \begin{cases} 1 & \text{if } 1 - \frac{|x-y|}{Max_r - Min_r} \geq \alpha \\ 0 & \text{otro caso} \end{cases} \tag{5.7}$$

donde $Max_r = \text{máx}_{O \in \Omega}\, I_{\{r\}}(O)$, $Min_r = \text{mín}_{O \in \Omega}\, I_{\{r\}}(O)$.

Para minar patrones similares frecuentes con la función de semejanza Booleana es usado el algoritmo *STreeDC-Miner* pues para esta función los algoritmos *ObjectMiner*, *STreeDC-Miner*, *STreeNDC-Miner*, *RP-Miner* obtienen el mismo conjunto de patrones similares frecuentes y *STreeDC-Miner* es el más rápido de ellos.

Eficiencia y eficacia de los algoritmos

La figura 5.11 muestra los tiempos de ejecución de los algoritmos *STree*DC-Miner* y *STreeDC-Miner* variando los umbrales de mínima frecuencia en las colecciones de datos *Diabetes*, *Liver Disorders* e *Iris*.

En los experimentos realizados el algoritmo *STreeDC-Miner* logra mejores tiempos que *STree*DC-Miner*. Estos resultados se deben a que:

- *STreeDC-Miner* realiza sólo operaciones aritméticas con enteros (los valores de semejanza toman valor 0 o 1) las cuales consumen menos tiempo que las operaciones aritméticas con punto flotante realizadas por *STree*DC-Miner* (los valores de semejanza toman valor en $[0, 1]$).

- El número de evaluaciones de la función de semejanza realizadas por *STreeDC-Miner* es mucho menor que el número de evaluaciones de la función de semejanza realizadas *STree*DC-Miner* (véase Figuras 5.12). Tanto *STreeDC-Miner* como *STree*DC-Miner* sólo evalúan la función de semejanza entre dos subdescripciones, si estas son expansiones de subdescripciones semejantes. La diferencia se debe a que cuando la función de semejanza es Booleana (como en *STreeDC-Miner*) se considera que dos subdescripciones son semejantes si el resultado de la evaluación es 1, mientras que cuando la función de semejanza es no Booleana (como en *STree*DC-Miner*) todas las subdescripciones con valor de semejanza entre ellas mayor o igual que $\beta > 0$, son consideradas semejantes.

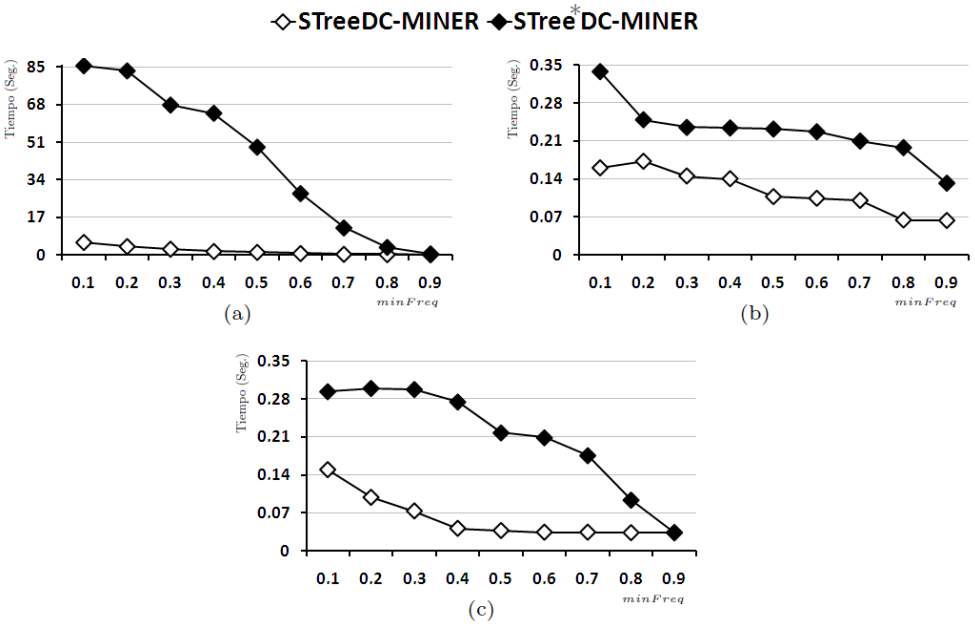


Figura 5.11: Tiempos de ejecución de *STree*DC-Miner* para la función de semejanza no Booleana (5.5) que cumple la propiedad de f_S-Clausura Descendente y de *STreeDC-Miner* para una Booleanización de dicha función, en las colecciones de datos (a) *Diabetes*, (b) *Liver Disorders* y (c) *Iris*.

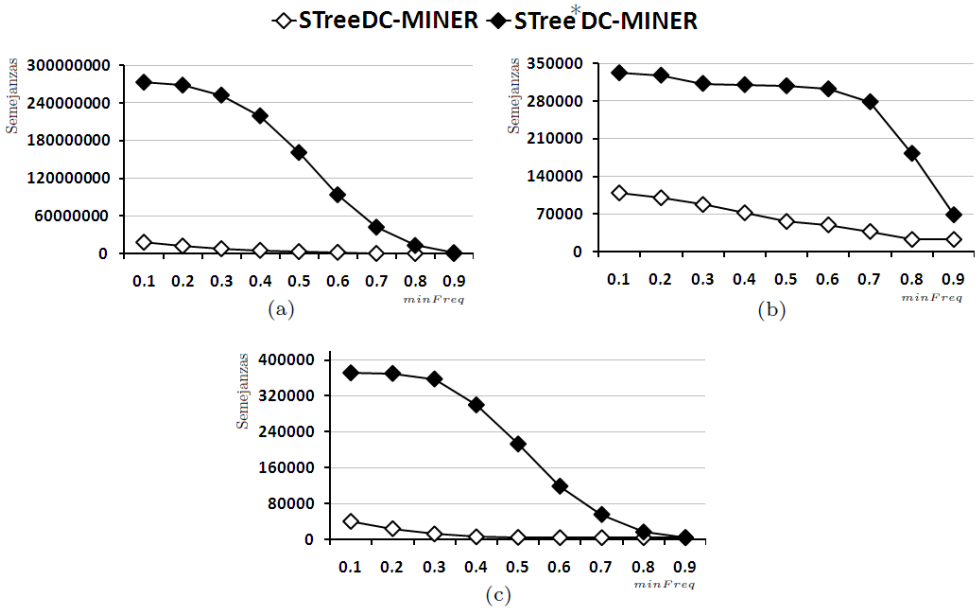


Figura 5.12: Número de evaluaciones de la función de semejanza realizadas por *STree*DC-Miner* para la función de semejanza no Booleana (5.5) que cumple la propiedad de f_S-Clausura Descendente y por *STreeDC-Miner* para una Booleanización de dicha función, en las colecciones de datos (a) *Diabetes*, (b) *Liver Disorders* y (c) *Iris*.

- El número de patrones similares frecuentes encontrados por *STreeDC-Miner* es mucho menor que el número de patrones similares frecuentes encontrados por *STree*DC-Miner* (véase figuras 5.13). Esto es una consecuencia del criterio de comparación (5.6) usado en la función de semejanza no Booleana y del criterio de comparación (5.7) con $\alpha = 0{,}9$ usado en la función de semejanza Booleana. Nótese, que para cada subdescripción P, el conjunto de subdescripciones semejantes a P usando (5.7) es un subconjunto del conjunto de subdescripciones semejantes a P usando (5.6). Por tanto, menos subdescripciones contribuyen a la frecuencia de P al usar (5.7), que al usar (5.6).

Sin embargo, es importante señalar que la diferencia entre el número de patrones similares frecuentes encontrados usando la función de semejanza no Booleana y su Booleanización, indica una cota inferior de la suma de patrones que, o son falsos patrones similares frecuentes obtenidos por *STreeDC-Miner*, o son patrones similares frecuentes que no pudieron ser encontrados por *STreeDC-Miner*.

Como puede observarse en la figura 5.13, al Booleanizar la función de semejanza y por tanto transformar el problema original, el número de falsos patrones similares frecuentes más el número de patrones que no pudieron ser encontrados está acotado inferiormente por 307 254 patrones en *Diabetes*, 623 patrones en *Liver Disorders* y 1 864 patrones en *Iris* para el umbral de mínima frecuencia 0,1.

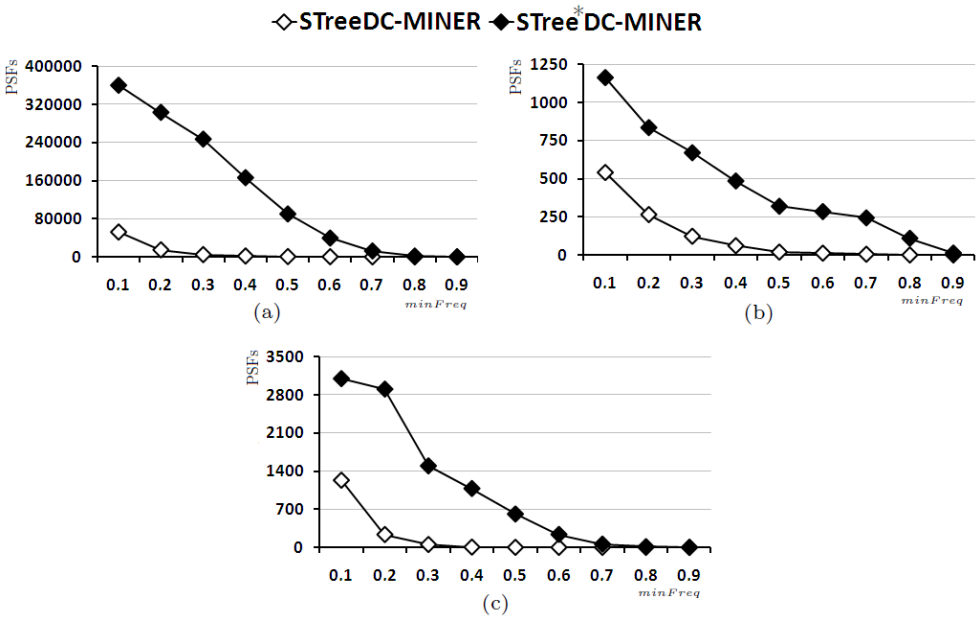


Figura 5.13: Número de patrones similares frecuentes encontrados por *STree*DC-Miner* para función de semejanza no Booleana (5.5) que cumple la propiedad de f_S-Clausura Descendente y por *STreeDC-Miner* para una Booleanización de dicha función, en las colecciones de datos (a) *Diabetes*, (b) *Liver Disorders* y (c) *Iris*.

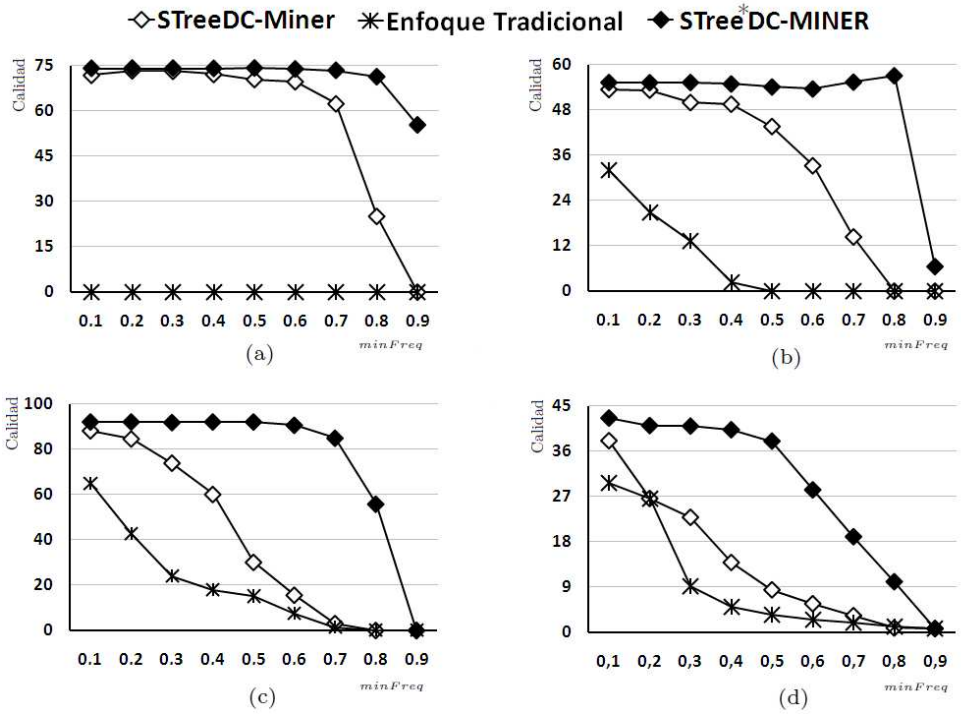


Figura 5.14: Calidad de los conjuntos de patrones similares frecuentes encontrados por *STreeDC-Miner*, *STree*DC-Miner* y *Enfoque Tradicional* en las colecciones de datos (a) *Diabetes*, (b) *Liver Disorders*, (c) *Iris* y (d) *Page Blocks*.

Calidad de los patrones minados

La calidad del conjunto de patrones similares frecuentes obtenidos usando la función de semejanza no Booleana y la Booleanización de la misma es medida tal y como se describió en la sección 5.2.1. Además incluimos los resultados usando la igualdad como función de semejanza (*Enfoque Tradicional*).

La clasificación es realizada para diferentes valores del umbral $minFreq$ a partir de $minFreq = 0{,}1$ y hasta $minFreq = 0{,}9$ con incrementos de 0,1. En la figura 5.14 y la tabla 5.5 son mostradas las calidades logradas por cada algoritmo en las colecciones de datos *Diabetes*, *Liver Disorders*, *Iris* y *Page Blocks*.

En la figura 5.14, se puede observar que la precisión de la clasificación alcanzada mediante los patrones similares frecuentes obtenidos por el algoritmo *STree*DC-Miner* usando la función de semejanza no Booleana fue mejor que la precisión de la clasificación alcanzada mediante los patrones similares frecuentes obtenidos por el algoritmo *STreeDC-Miner* usando la Booleanización de la misma. Estos resultados confirman el efecto negativo que puede acarrear transformar una función no Booleana en Booleana, es decir, el efecto que pueden provocar los falsos patrones similares frecuentes que son generados y los patrones similares frecuentes que se pierden como resultado de

Tabla 5.5: Calidad de los conjuntos de patrones similares frecuentes encontrados por *STreeDC-Miner*, *Enfoque Tradicional* y *STree*DC-Miner* en la colecciones de datos *Diabetes*, *Liver Disorders*, *Iris* y *Page Blocks*.

Colección de Datos	*minFreq*	*STreeDC-Miner*	*Enfoque Tradicional*	*STree* DC-Miner*
Diabetes	0,10	71,9	0,0	74,0
	0,20	73,4	0,0	**74.1**
	0,30	73,2	0,0	**74.1**
	0,40	72,2	0,0	74,0
	0,50	70,5	0,0	**74.1**
	0,60	69,7	0,0	73,8
	0,70	62,4	0,0	73,4
	0,80	25,1	0,0	71,3
	0,90	0,0	0,0	55,5
Precisión Promedio		57,6	0,0	71,6
Liver Disorders	0,10	53,4	32,0	55,3
	0,20	53,2	20,8	55,2
	0,30	50,0	13,3	55,3
	0,40	49,5	2,3	54,9
	0,50	43,6	0,0	54,1
	0,60	33,2	0,0	53,5
	0,70	14,3	0,0	55,4
	0,80	0,0	0,0	**57.1**
	0,90	0,0	0,0	6,3
Precisión Promedio		33,0	7,6	49,7
Iris	0,10	88,3	64,9	**92.3**
	0,20	84,7	42,7	**92.3**
	0,30	74,0	23,9	92,0
	0,40	60,1	17,9	**92.3**
	0,50	30,1	15,1	**92.3**
	0,60	15,6	7,5	90,8
	0,70	3,1	1,3	85,1
	0,80	0,0	0,0	55,9
	0,90	0,0	0,0	0,0
Precisión Promedio		39,5	19,3	77,0
Page Blocks	0,10	38,1	29,7	**42.6**
	0,20	26,6	26,6	41,1
	0,30	22,9	9,2	41,0
	0,40	13,9	4,9	40,3
	0,50	8,4	3,5	38,0
	0,60	5,6	2,5	28,3
	0,70	3,3	1,8	19,0
	0,80	0,8	1,1	10,0
	0,90	0,7	0,7	0,7
Precisión Promedio		13,4	8,9	29,0

la Booleanización.

También se puede apreciar una vez más que usando la igualdad como función de semejanza (como en el enfoque tradicional de minado de patrones frecuentes) la precisión de la clasificación alcanzada mediante el conjunto de patrones frecuentes es menor que la precisión de la clasificación alcanzada mediante el conjunto de patrones similares frecuentes obtenidos usando tanto funciones de semejanza Booleanas como no Booleanas (diferentes de la igualdad).

Adicionalmente, en la tabla 5.5, se puede observar que cuando el umbral de mínima frecuencia $minFreq$ crece, la cantidad de patrones similares frecuentes decrece. Sin embargo, el uso de la función de semejanza no Booleana permitió al algoritmo *STree*DC-Miner* encontrar más patrones similares frecuentes, incluso para valores altos de $minFreq$, y como consecuencia la precisión del clasificador cuando usa estos patrones similares frecuentes disminuye más lentamente que al usar los patrones similares frecuentes obtenidos usando la función de semejanza Booleana. Más aun, cuando la igualdad es usada como función de semejanza, el número de patrones frecuentes encontrados para la colección *Diabetes* es muy bajo incluso para valores pequeños de $minFreq$ y por tanto el clasificador no fue capaz de clasificar correctamente algún objeto.

5.3.2. Experimentos con los algoritmos propuestos para funciones de semejanza no Booleana que no cumplen la propiedad de f_S-Clausura Descendente

Análogamente a la situación que se da para las funciones de semejanza Booleanas, cuando las funciones de semejanza no Booleana no cumplen la propiedad de f_S-Clausura Descendente, el algoritmo *STree*DC-Miner* cuyo mecanismo de poda supone que el cumplimiento de dicha propiedad, también pierde patrones similares frecuentes, mientras el algoritmo *STree*NDC-Miner* no pierde patrones similares frecuentes pero recorre exhaustivamente el espacio de búsqueda lo cual afecta su eficiencia y el algoritmo *RP*-Miner* debido a su poda relajada pierde menos patrones similares frecuentes que *STree*DC-Miner* y es más eficiente que *STree*NDC-Miner*.

A continuación los algoritmos *STree*DC-Miner*, *STree*NDC-Miner*, *RP*-Miner* son comparados en cuanto a su eficiencia y eficacia. Además, la calidad del conjunto de patrones similares frecuentes obtenidos por cada uno también es comparada.

Como función de semejanza que no cumple la propiedad de f_S-Clausura Descendente usamos la función (5.8) con los criterios de comparación (5.3) para los atributos no numéricos y (5.6) para los atributos numéricos.

$$f_S(O, O') = \frac{\sum_{r \in S} C_r(O[r], O'[r])}{|S|} \tag{5.8}$$

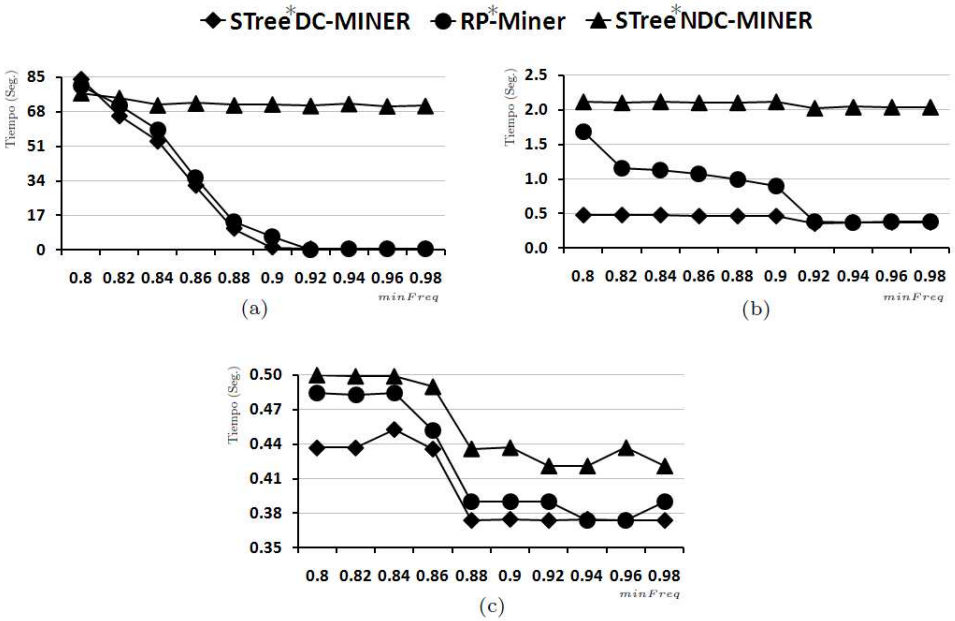


Figura 5.15: Tiempos de ejecución de *STree*DC-Miner*, *RP*-Miner* y *STree*NDC-Miner* para la función de semejanza no Booleana (5.8) que no cumple la propiedad de f_S-Clausura Descendente en las colecciones de datos (a) *Diabetes*, (b) *Liver Disorders* y (c) *Iris*.

Eficiencia y eficacia de los algoritmos

La figura 5.15 muestra los tiempos de ejecución de los algoritmos *STree*DC-Miner*, *RP*-Miner* y *STree*NDC-Miner* variando los umbrales de mínima frecuencia en las colecciones de datos *Diabetes*, *Liver Disorders* e *Iris*.

Para las tres colecciones el tiempo de ejecución de *RP*-Miner* fue mayor que el tiempo de ejecución de *STree*DC-Miner* y el tiempo de ejecución de *STree*NDC-Miner* fue generalmente el mayor de todos. Análogamente a los algoritmos de minado de patrones similares frecuentes que usan funciones de semejanza Booleana, esto es una consecuencia del número de patrones similares frecuentes encontrados (véase tabla 5.6) y del número de evaluaciones de la función de semejanza (véase figura 5.16) por cada algoritmo.

Es importante subrayar que *STree*NDC-Miner* encuentra todos los patrones similares frecuentes, mientras *STree*DC-Miner*, que supone que la función de semejanza f_S cumple la propiedad de f_S-Clausura Descendente, puede no encontrar todos los patrones similares frecuentes. Por su parte, *RP*-Miner* también puede no encontrar todos los patrones similares frecuentes, pero el uso de la poda relajada le permite encontrar patrones similares frecuentes que *STree*DC-Miner* pierde.

Nótese que para $minFreq = 0{,}8$, respecto al número de patrones similares frecuentes obtenidos por *STree*NDC-Miner*, en *Diabetes*, tanto *STree*DC-Miner* como *RP*-Miner*

Tabla 5.6: Número de patrones similares frecuentes encontrados por *STree*DC-Miner*, *RP*-Miner* y *STree*NDC-Miner* para la función de semejanza no Booleana (5.8) que no cumple la propiedad de f_S-Clausura Descendente en las colecciones de datos *Diabetes*, *Liver Disorders* e *Iris*.

		minFreq									
		0,80	0,82	0,84	0,86	0,88	0,90	0,92	0,94	0,96	0,98
Colección de Datos	Algoritmo	Patrones Similares Frecuentes									
Diabetes	*STree* DC-Miner*	211592	154652	93999	42754	7998	448	0	0	0	0
	RP-Miner*	211592	154652	93999	42899	8163	809	0	0	0	0
	STree NDC-Miner*	226225	168696	105711	50994	10355	873	0	0	0	0
Liver Disorders	*STree* DC-Miner*	256	232	207	161	120	61	0	0	0	0
	RP-Miner*	256	232	207	161	120	61	0	0	0	0
	STree NDC-Miner*	267	239	219	185	138	67	0	0	0	0
Iris	*STree* DC-Miner*	29	6	4	1	0	0	0	0	0	0
	RP-Miner*	45	21	4	1	0	0	0	0	0	0
	STree NDC-Miner*	65	21	4	1	0	0	0	0	0	0

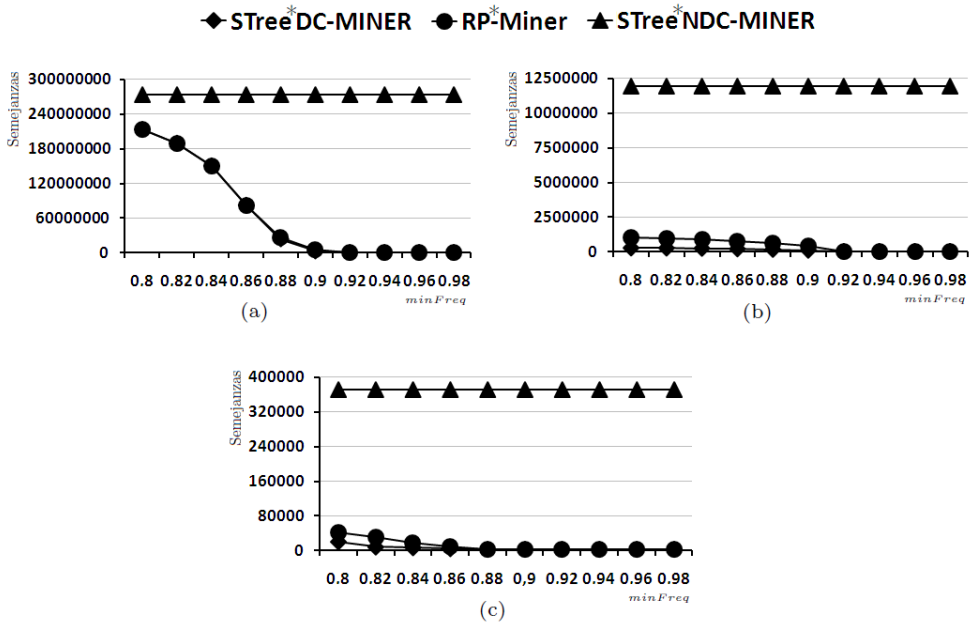


Figura 5.16: Número de evaluaciones de la función de semejanza realizadas por *STree*DC-Miner*, *RP*-Miner* y *STree*NDC-Miner* para la función de semejanza no Booleana (5.8) que no cumple la propiedad de f_S-Clausura Descendente en las colecciones de datos *Diabetes*, (b) *Liver Disorders* y (c) *Iris*.

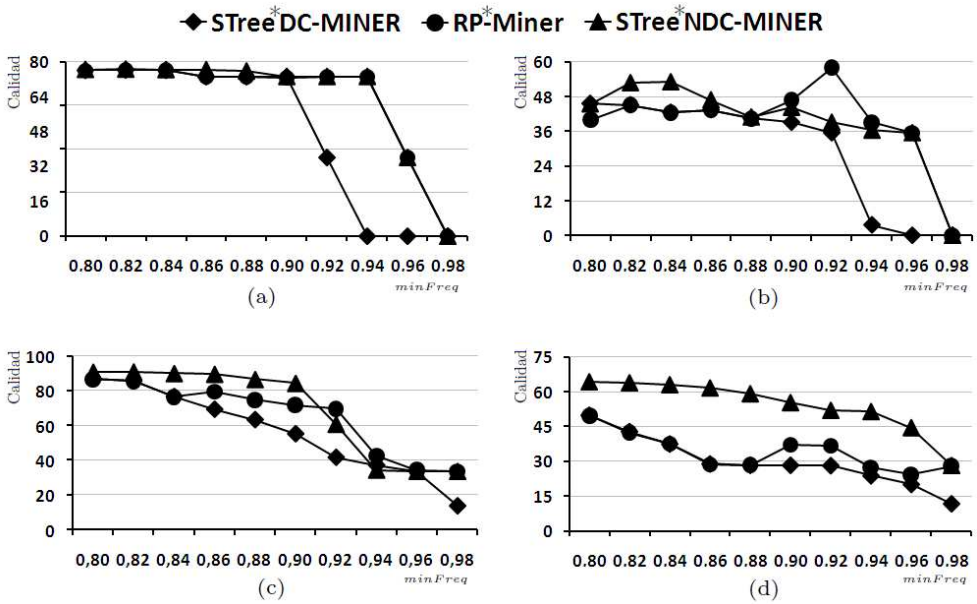


Figura 5.17: Calidad de los conjuntos de patrones similares frecuentes encontrados por *STree*DC-Miner*, *RP*-Miner* y *STree*NDC-Miner* en las colecciones de datos (a) *Diabetes*, (b) *Liver Disorders*, (c) *Iris* y (d) *Page Blocks*.

pierden 14 633 (6,47 %) patrones similares frecuentes. Análogamente, en *Liver Disorders*, *STree*DC-Miner* y *RP*-Miner* pierden 11 (4,12 %) patrones similares frecuentes. En el caso de la colección *Iris*, *STree*DC-Miner* pierde 36 (55,38 %) patrones similares frecuentes, mientras *RP*-Miner* pierde menos patrones similares frecuentes (20, que presentan el 30,70 %).

Para los otros valores de $minFreq$, en las tres colecciones el número de patrones similares frecuentes perdidos por los algoritmos es menor, no obstante, *RP*-Miner* siempre pierde menos o los mismos patrones similares frecuentes que *STree*DC-Miner*.

Calidad de los patrones minados

En esta sección se evalúa la calidad del conjunto de patrones similares frecuentes obtenidos por *STree*DC-Miner*, *STree*NDC-Miner* y *RP*-Miner* algoritmo tal y como se describió en la sección 5.2.1.

La clasificación es realizada para diferentes valores del umbral $minFreq$ a partir de $minFreq = 0{,}80$ y hasta $minFreq = 0{,}98$ con incrementos de 0,02. En la figura 5.17 y la tabla 5.7 son mostradas las calidades logradas por cada algoritmo en las colecciones de datos *Diabetes*, *Liver Disorders*, *Iris* y *Page Blocks*.

En la mayoría de los casos la calidad del conjunto de patrones similares frecuentes obtenidos con *STree*NDC-Miner* fue mejor que la calidad del conjunto de patrones similares frecuentes obtenidos mediante *RP*-Miner*, la cual a su vez en la mayoría de los

Tabla 5.7: Calidad de los conjuntos de patrones similares frecuentes encontrados por *STree*DC-Miner*, *RP*-Miner* y *STree*NDC-Miner* en la colecciones de datos *Diabetes*, *Liver Disorders*, *Iris* y *Page Blocks*.

Colección de Datos	$minFreq$	*STree*DC-Miner*	*RP*-Miner*	*STree*NDC-Miner*
Diabetes	0,80	76,10	76,10	76,46
	0,82	76,26	76,24	**76.58**
	0,84	76,12	76,30	76,36
	0,86	73,20	73,20	76,30
	0,88	73,20	73,20	75,70
	0,90	73,20	72,76	73,20
	0,92	36,10	73,20	73,20
	0,94	0,00	73,20	73,20
	0,96	0,00	36,10	36,10
	0,98	0,00	0,00	0,00
Precisión Promedio		40,81	55,42	56,06
Liver Disorders	0,80	45,52	40,00	45,42
	0,82	45,07	45,07	52,69
	0,84	42,49	42,49	52,99
	0,86	43,38	43,38	46,77
	0,88	40,40	40,40	40,85
	0,90	39,05	46,77	44,23
	0,92	35,42	**58.01**	39,05
	0,94	3,63	39,05	36,32
	0,96	0,00	35,42	35,42
	0,98	0,00	0,00	0,00
Precisión Promedio		24,94	35,06	34,83
Iris	0,80	86,53	86,53	90,80
	0,82	85,33	85,33	**90.93**
	0,84	76,40	76,40	90,00
	0,86	69,07	79,33	89,60
	0,88	63,07	74,67	86,53
	0,90	54,93	71,60	84,27
	0,92	41,33	69,47	60,67
	0,94	36,80	42,40	34,13
	0,96	33,33	34,13	33,47
	0,98	13,33	33,33	33,33
Precisión Promedio		47,36	56,67	60,29
Page Blocks	0,80	49,71	49,77	**64.32**
	0,82	42,72	42,42	63,91
	0,84	37,41	37,51	63,08
	0,86	28,87	28,59	61,67
	0,88	28,32	28,31	59,20
	0,90	28,31	37,18	55,32
	0,92	28,15	36,68	51,99
	0,94	23,82	27,44	51,46
	0,96	20,10	24,25	44,53
	0,98	27,91	27,91	28,17
Precisión Promedio		26,56	29,03	47,93

casos fue mejor que la calidad del conjunto de patrones similares frecuentes obtenidos mediante *STree*DC-Miner*. De hecho la mejor precisión lograda por el clasificador (celdas grises en la tabla 5.7) para 3 colecciones fue alcanzada por *STree*NDC-Miner* y en la restante colección fue alcanzada por *RP*-Miner*.

Dado que el conjunto de patrones similares frecuentes encontrados por *STree*DC-Miner* es un subconjunto de los patrones similares frecuentes encontrados por *RP*-Miner* y estos son un subconjunto de los patrones similares frecuentes encontrados por *STree*NDC-Miner*; se puede afirmar que en general tanto los patrones similares frecuentes perdidos por *STree*DC-Miner* y encontrados por *RP*-Miner*, como los patrones similares frecuentes perdidos por *RP*-Miner* y encontrados por *STree*NDC-Miner* afectan la clasificación. No obstante, no siempre encontrar más patrones o todos los patrones similares frecuentes es la mejor opción.

5.3.3. Experimentos tratando el problema de los bajas semejanzas y los muchos patrones frecuentes

El problema de las bajas semejanzas y los muchos patrones frecuentes, como se explica en el capítulo 4, puede darse cuando el grado de semejanza entre una subdescripción P y las subdescripciones semejantes a ella, es muy bajo, pero el número de patrones semejantes a ella es elevado y como consecuencia P podría ser considerado erróneamente un patrón similar frecuente. Cuando esta situación se presenta, la misma se puede atacar desde la etapa de modelación de la función de semejanza, definiendo la semejanza para los valores pequeños de la misma como cero. En el capítulo 4 se presenta una alternativa para cuando esta situación es detectada posteriormente a la modelación de la semejanza. Esta alternativa no tiene en cuenta, en el cálculo de la frecuencia de cada subdescripción, valores de semejanza pequeños. Además, se propuso cómo estimar el umbral de mínima semejanza. En esta sección tratamos el problema de las bajas semejanzas y los muchos patrones frecuentes, estimando el umbral de mínima semejanza β como se propuso en el capítulo 4.

Como función de semejanza no Booleana tomamos la función (5.8) con los criterios de comparación (5.3) para los atributos no numéricos y (5.6) para los atributos numéricos. Dicha función no cumple la propiedad de f_S-Clausura Descendente y por tanto los algoritmos *STree*DC-Miner*, *STree*NDC-Miner* y *RP*-Miner* obtienen diferentes conjuntos de patrones similares frecuentes.

En la figura 5.18 y la tabla 5.8 son mostradas las calidades de los conjuntos de patrones similares frecuentes obtenidos por los algoritmos *STree*DC-Miner*, *STree*NDC-Miner*, *RP*-Miner* para las colecciones *Diabetes*, *Liver Disorders*, *Iris* y *Page Blocks*. Por cada colección la clasificación es realizada para diferentes valores del umbral $minFreq$ a partir de $minFreq = 0{,}80$ hasta $minFreq = 0{,}98$ con incrementos de 0,02.

En la figura 5.18 puede observarse que análogamente a cuando el problema de las bajas semejanzas y los muchos patrones frecuentes no es tratado, los patrones similares frecuentes perdidos pueden afectar la precisión del clasificador y que a la vez, encontrar más patrones similares frecuentes o incluso encontrar todos no es siempre la mejor opción.

Los patrones similares frecuentes obtenidos por *STree*DC-Miner* alcanzaron los peores resultados, mientras la mejor precisión lograda por el clasificador, (celdas grises en la tabla 5.8) para 3 colecciones, fue usando los patrones similares frecuentes obtenidos por *RP*-Miner* y en la restante colección la mejor precisión lograda por el clasificador fue usando los patrones similares frecuentes obtenidos por *STree*NDC-Miner*.

5.4. Experimentos de minado de reglas de asociación

A continuación, se muestra que al minar reglas de asociación usando como función de semejanza la igualdad pueden perderse reglas de asociación interesantes y generar falsas reglas de asociación interesantes.

Tabla 5.8: Calidad de los conjuntos de patrones similares frecuentes encontrados por *STree*DC-Miner*, *RP*-Miner* y *STree*NDC-Miner*, al atacar problema de las bajas semejanzas y los muchos patrones frecuentes, en las colecciones de datos *Diabetes*, *Liver Disorders*, *Iris* y *Page Blocks*.

Colección de Datos	$minFreq$	*STree*DC-Miner*	*RP*-Miner*	*STree*NDC-Miner*
Diabetes	0,80	64, 62	63, 36	58, 74
	0,82	64, 50	65, 02	58, 96
	0,84	64, 74	65, 38	59, 02
	0,86	64, 96	53, 90	60, 56
	0,88	63, 80	51, 38	62, 90
	0,90	51, 66	59, 74	64, 56
	0,92	26, 08	68, 26	65, 88
	0,94	0, 00	**69,30**	69, 28
	0,96	0, 00	39, 50	39, 50
	0,98	0, 00	0, 00	0, 00
Precisión Promedio		33, 57	47, 25	48, 07
Liver Disorders	0,80	51, 84	**60,95**	58, 21
	0,82	51, 84	**60,95**	58, 16
	0,84	51, 84	**60,95**	58, 31
	0,86	51, 84	50, 00	57, 86
	0,88	51, 84	50, 00	56, 87
	0,90	22, 74	52, 74	54, 38
	0,92	12, 74	40, 90	46, 37
	0,94	0, 00	43, 58	42, 44
	0,96	0, 00	28, 16	28, 16
	0,98	0, 00	0, 00	0, 00
Precisión Promedio		24, 28	38, 73	40, 26
Iris	0,80	41, 60	**53,47**	52, 00
	0,82	43, 47	52, 53	49, 33
	0,84	46, 67	48, 80	47, 20
	0,86	45, 33	51, 47	46, 40
	0,88	44, 27	49, 73	46, 80
	0,90	42, 13	43, 20	46, 13
	0,92	38, 93	42, 67	48, 80
	0,94	26, 67	46, 13	42, 27
	0,96	26, 67	40, 53	36, 93
	0,98	13, 33	33, 33	33, 33
Precisión Promedio		64, 96	40, 83	39, 71
Page Blocks	0,80	28, 37	28, 35	49, 78
	0,82	28, 17	28, 17	49, 57
	0,84	28, 17	28, 16	49, 01
	0,86	28, 16	28, 17	48, 35
	0,88	26, 78	19, 37	47, 89
	0,90	19, 29	28, 51	**52,16**
	0,92	19, 30	27, 63	48, 34
	0,94	14, 93	17, 97	43, 66
	0,96	11, 17	15, 32	35, 77
	0,98	10, 48	27, 91	28, 17
Precisión Promedio		18, 65	22, 12	40, 29

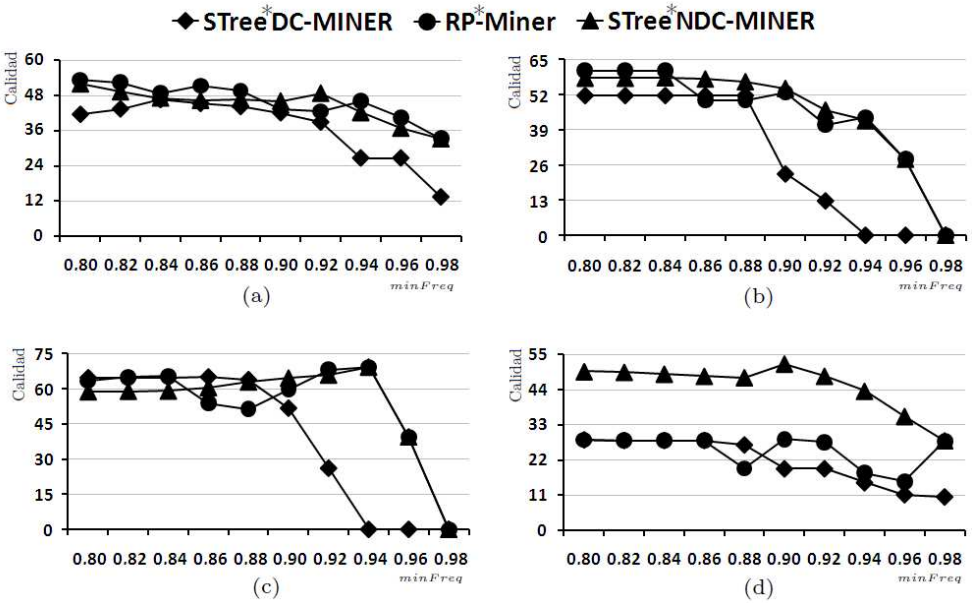


Figura 5.18: Calidad de los conjuntos de patrones similares frecuentes encontrados por *STree*DC-Miner*, *RP*-Miner* y *STree*NDC-Miner*, al atacar problema de las bajas semejanzas y los muchos patrones frecuentes, en las colecciones de datos (a) *Diabetes*, (b) *Liver Disorders*, (c) *Iris* y (d) *Page Blocks*.

En las tablas 5.9 y 5.10 se muestran las reglas de asociación generadas a partir de las colecciones *Contraceptive Method Choice* y *Diabetes*, tanto usando la igualdad como función de semejanza, como usando la función de semejanza diferente de la igualdad (5.5) con el criterio de comparación (5.3) para los atributos no numéricos y el criterio de comparación (5.7) con $\alpha = 0{,}9$ para los atributos numéricos. Para la colección *Contraceptive Method Choice* fueron considerados los umbrales $minFreq = 0{,}25$ y $minConf = 0{,}98$ y para la colección *Diabetes* $minFreq = 0{,}25$ y $minConf = 0{,}96$. Los valores de estos umbrales fueron tomados así, para poder mostrar todas las reglas de asociación generadas.

En las tablas 5.9 y 5.10, las reglas de asociación que no pudieron ser encontradas por el enfoque tradicional están en **negrita**.

Puede apreciarse que para la colección *Contraceptive Method Choice* (tabla 5.9) 2 de las 11 reglas de asociación interesantes no pudieron ser obtenidas al usar como función de semejanza la igualdad, mientras que para la colección *Diabetes* (tabla 5.10) 7 de las 8 reglas de asociación interesantes no pudieron ser obtenidas al usar como función de semejanza la igualdad. En ambos casos al usar como función de semejanza la igualdad no se generaron falsas reglas de asociación.

Un efecto análogo al que sucede al usar la igualdad como función de semejanza, puede suceder al emplear una Booleanización de una función de semejanza no Booleana.

En las tablas 5.11 y 5.12 se muestran las reglas de asociación generadas para las mismas colecciones tanto usando una función de semejanza no Booleana como una

Tabla 5.9: Reglas de asociación generadas para *Contraceptive Method Choice* dado $minFreq = 0{,}25$ y $minConf = 0{,}98$.

Reglas de asociación generadas usando la igualdad como función de semejanza
$(Wife's\ education = High) \rightarrow (Media\ exposure = Good)$
$(Husband's\ education = High, Standard\ of\ living = High) \rightarrow (Media\ exposure = Good)$
$(Wife's\ education = High, Standard\ of\ living = High) \rightarrow (Media\ exposure = Good)$
$(Husband's\ education = High, Wife's\ education = High) \rightarrow (Media\ exposure = Good)$
$(Wife's\ education = High, Wife's\ now\ working? = No) \rightarrow (Media\ exposure = Good)$
$(Wife's\ education = High, Wife's\ religion = Islam) \rightarrow (Media\ exposure = Good)$
$(Husband's\ education = High, Wife's\ education = High, Standard\ of\ living = High) \rightarrow (Media\ exposure = Good)$
$(Husband's\ education = High, Wife's\ education = High, Wife's\ now\ working? = No) \rightarrow (Media\ exposure = Good)$
$(Husband's\ education = High, Wife's\ education = High, Wife's\ religion = Islam) \rightarrow (Media\ exposure = Good)$
Reglas de asociación interesantes generadas usando función de semejanza diferente de la igualdad
$(Wife's\ education = High) \rightarrow (Media\ exposure = Good)$
(Children ever born=1 ,Husband's education=High) $\rightarrow$ ***(Media exposure=Good)***
$(Husband's\ education = High, Standard\ of\ living = High) \rightarrow (Media\ exposure = Good)$
$(Wife's\ education = High, Standard\ of\ living = High) \rightarrow (Media\ exposure = Good)$
$(Husband's\ education = High, Wife's\ education = High) \rightarrow (Media\ exposure = Good)$
$(Wife's\ education = High, Wife's\ now\ working? = No) \rightarrow (Media\ exposure = Good)$
$(Wife's\ education = High, Wife's\ religion = Islam) \rightarrow (Media\ exposure = Good)$
(Children ever born=2,Husband's education=High,Wife's now working?=No) $\rightarrow$ ***(Media exposure=Good)***
$(Husband's\ education = High, Wife's\ education = High, Standard\ of\ living = High) \rightarrow (Media\ exposure = Good)$
$(Husband's\ education = High, Wife's\ education = High, Wife's\ now\ working? = No) \rightarrow (Media\ exposure = Good)$
$(Husband's\ education = High, Wife's\ education = High, Wife's\ religion = Islam) \rightarrow (Media\ exposure = Good)$

Tabla 5.10: Reglas de asociación generadas para *Diabetes* dado $minFreq = 0{,}25$ y $minConf = 0{,}96$.

Reglas de asociación generadas usando la igualdad como función de semejanza
$(triceps\ skin = 0) \rightarrow (Insulin = 0)$
Reglas de asociación interesantes generadas usando función de semejanza diferente de la igualdad
(Plasma glucose concentration=81) $\rightarrow$ ***(Insulin=76)***
$(triceps\ skin = 0) \rightarrow (Insulin = 0)$
(Plasma glucose concentration=87, Diabetes=No) $\rightarrow$ ***(Insulin=77)***
(Plasma glucose concentration=83, Diabetes=No) $\rightarrow$ ***(Insulin=66)***
(Plasma glucose concentration=83, Diabetes=No) $\rightarrow$ ***(Insulin=50)***
(Plasma glucose concentration=84, Diabetes=No) $\rightarrow$ ***(Insulin=76)***
(Plasma glucose concentration=83, Diabetes=No) $\rightarrow$ ***(Insulin=71)***
(Plasma glucose concentration=88, Diabetes=No) $\rightarrow$ ***(Insulin=76)***

Booleanización de la misma. Como función de semejanza no Booleana usamos (5.5) con los criterios de comparación (5.3) para los atributos no numéricos y (5.6) para los atributos numéricos y como Booleanización de la función de semejanza tomamos la misma función (5.5) pero con el criterio de comparación (5.7) con $\alpha = 0{,}9$ para los atributos numéricos. Para la colección *Contraceptive Method Choice* fueron considerados los umbrales $minFreq = 0{,}23$ y $minConf = 0{,}994$ y para la colección *Diabetes* $minFreq = 0{,}4$ y $minConf = 0{,}932$. Los valores de estos umbrales fueron tomados así, para poder mostrar todas las reglas de asociación generadas.

En las tablas 5.11 y 5.12, las falsas reglas de asociación generadas al usar la Booleanización de la función de semejanza no Booleana están contenidas en celdas grises y las reglas de asociación que no pudieron ser encontradas por dicha función están en **negrita**.

Puede apreciarse que para la colección *Contraceptive Method Choice* (tabla 5.11) 4 de

las 6 reglas de asociación interesantes no pudieron ser obtenidas al usar la Booleanización de la función de semejanza, mientras que para la colección *Diabetes* (tabla 5.12) no sólo ninguna regla de asociación interesante pudo ser obtenida al usar la Booleanización de la función de semejanza, sino que se generaron 2 falsas reglas de asociación interesantes.

Tabla 5.11: Reglas de asociación generadas para *Contraceptive Method Choice* dado $minFreq = 0{,}23$ y $minConf = 0{,}994$.

Reglas de asociación generadas usando Booleanización de la función de semejanza
$(Wife's\ education = High, Standard\ of\ living = High) \rightarrow (Media\ exposure = Good)$
$(Husband's\ education = High, Wife's\ education = High, Standard\ of\ living = High) \rightarrow (Media\ exposure = Good)$
Reglas de asociación interesantes generadas usando función de semejanza no Booleana
$(Wife's\ education = High, Standard\ of\ living = High) \rightarrow (Media\ exposure = Good)$
(Children ever born=4, Wife's education=High, Standard of living=High) → (Media exposure=Good)
(Children ever born=2, Wife's education=High, Standard of living=High) → (Media exposure=Good)
$(Husband's\ education = High, Wife's\ education = High, Standard\ of\ living = High) \rightarrow (Media\ exposure = Good)$
(Children ever born=3, Wife's education=High, Standard of living=High) → (Media exposure=Good)
(Children ever born=3, Husband's education=High, Wife's education=High, Standard of living=High) → (Media exposure=Good)

Tabla 5.12: Reglas de asociación generadas para *Diabetes* dado $minFreq = 0{,}4$ y $minConf = 0{,}932$.

Reglas de asociación generadas usando Booleanización de la función de semejanza
$(Plasma\ glucose\ concentration = 98) \rightarrow (Insulin = 84)$
$(Plasma\ glucose\ concentration = 97) \rightarrow (Insulin = 82)$
Reglas de asociación interesantes generadas usando función de semejanza no Booleana
(Plasma glucose concentration=82, Triceps skin=0, Diabetes=No) → (Insulin=0)
(Plasma glucose concentration=80, Triceps skin=0, Diabetes=No) → (Insulin=0)
(Plasma glucose concentration=83, Triceps skin=0, Diabetes=No) → (Insulin=0)
(Plasma glucose concentration=84, Triceps skin=0, Diabetes=No) → (Insulin=0)
(Plasma glucose concentration=78, Triceps skin=0, Diabetes=No) → (Insulin=0)
(Plasma glucose concentration=85, Triceps skin=0, Diabetes=No) → (Insulin=0)

5.5. Síntesis y Conclusiones

En este capítulo se presentó el desempeño de los algoritmos propuestos y una comparación de los mismos contra *ObjectMiner* y el enfoque tradicional de minado de patrones frecuentes para varias colecciones de datos y funciones de semejanza. La experimentación fue dividida en tres grupos de experimentos.

En el primer grupo fueron evaluados los algoritmos propuestos para el minado de patrones similares frecuentes *STreeDC-Miner*, *STreeNDC-Miner* y *RP-Miner* respecto al algoritmo *ObjectMiner*. Para ello se usaron funciones de semejanza Booleana tanto que cumplen la propiedad de f_S-Clausura Descendente como que no la cumplen. Además se compararon los algoritmos propuestos con enfoque tradicional de minado de patrones frecuentes. De los experimentos realizados se pudo constatar que:

- *STreeDC-Miner*. Es la mejor alternativa cuando la función de semejanza es Booleana y cumple la propiedad de f_S-Clausura Descendente, pues es más eficiente que *ObjectMiner*. La eficacia de *STreeDC-Miner* es igual a la de *ObjectMiner* y la calidad

de los patrones encontrados por *STreeDC-Miner* es igual a la de los patrones encontrados por *ObjectMiner* y superior a la calidad de los patrones encontrados por el enfoque tradicional de minado de patrones frecuentes.

Cuando la función de semejanza es Booleana y no cumple la propiedad de f_S-Clausura Descendente, la eficiencia de *STreeDC-Miner* es menor que la de *ObjectMiner* y mayor que la del resto de los algoritmos, y la calidad de los patrones obtenidos por *STreeDC-Miner* sólo es sobrepasada por la calidad de los patrones obtenidos por *RP-Miner*.

- *STreeNDC-Miner*. Es una alternativa cuando se desean encontrar todos los patrones similares frecuentes en una colección de datos de hasta 2000 objetos y 10 atributos, usando una función de semejanza Booleana que no cumple la propiedad de f_S-Clausura Descendente.

- *RP-Miner*. Es una alternativa intermedia entre *STreeDC-Miner* y *STreeNDC-Miner*. La eficiencia de *RP-Miner* es mayor que la de *STreeNDC-Miner*, lo cual permite emplearlo en colecciones donde *STreeNDC-Miner* se demora demasiado, mientras su eficacia es mayor que la de *STreeDC-Miner*. Además, la calidad de los patrones encontrados por él es mayor que la calidad de los patrones encontrados por *STreeDC-Miner*, *STreeNDC-Miner* y *ObjectMiner*.

En el segundo grupo de experimentos, fueron evaluados los algoritmos propuestos para el minado de patrones similares frecuentes *STree*DC-Miner*, *STree*NDC-Miner* y *RP*-Miner* usando funciones de semejanza no Booleana tanto las que cumplen como las que no cumplen la propiedad de f_S-Clausura Descendente. También se comparó contra los resultados del enfoque tradicional de minado de patrones frecuentes. De los experimentos realizados se pudo constatar que:

- *STree*DC-Miner*. Es la mejor alternativa cuando la función de semejanza no es Booleana y cumple la propiedad de f_S-Clausura Descendente pues es más eficiente que los algoritmos *STree*NDC-Miner* y *RP*-Miner*. La eficacia de *STree*DC-Miner* es igual a la de estos algoritmos. Como consecuencia, la calidad de los patrones encontrados por *STree*DC-Miner* también es igual a la de los patrones encontrados por estos algoritmos *STree*NDC-Miner* y *RP*-Miner*. Además, la calidad de los patrones encontrados por él, es superior a la calidad de los patrones obtenidos por el enfoque tradicional de minado de patrones frecuentes y a la calidad de los patrones encontrados por el algoritmo *STreeDC-Miner* usando una Booleanización de la función de semejanza.

  Cuando la función de semejanza no es Booleana y no cumple la propiedad de f_S-Clausura Descendente su eficiencia es mayor que la de *STree*NDC-Miner* y *RP*-Miner* y tanto su eficacia como la calidad de los patrones encontrados por *STree*DC-Miner* son menores que las de *STree*NDC-Miner* y *RP*-Miner*.

- *STree*NDC-Miner*. Es una alternativa cuando se desean encontrar todos los patrones similares frecuentes en una colección de datos de hasta 1000 objetos y 10 atributos usando una función de semejanza no Booleana que no cumple la propiedad de f_S-Clausura Descendente, pues no sólo encuentra todos los patrones similares frecuentes sino que la calidad del conjunto de patrones es superior a la del conjunto de patrones generado por *STree*DC-Miner* y *RP*-Miner*.

- *RP*-Miner*. Es una alternativa intermedia entre *STree*DC-Miner* y *STree*NDC-Miner* pues su eficiencia es mayor que la de *STree*NDC-Miner*, lo cual permite emplearlo en colecciones donde *STree*NDC-Miner* se demora demasiado. Su eficacia es mayor que la eficacia de *STree*DC-Miner* y la calidad de los patrones encontrados por *RP*-Miner* también es mayor que la calidad de los patrones encontrados por *STree*DC-Miner*.

Adicionalmente, en este grupo de experimentaciones se mostró que al atacar el problema de los muchos patrones y las bajas frecuencias, la calidad de los patrones encontrados por *STree*NDC-Miner* es la mejor, seguida por la calidad de los patrones encontrados por *RP*-Miner*.

En el tercer grupo de experimentos fueron comparadas las reglas de asociación interesantes encontradas por el enfoque tradicional respecto a las reglas de asociación interesantes encontradas cuando se usa una función de semejanza Booleana o no Booleana. En ambos casos se mostró que al Booleanizar una función de semejanza no Booleana, se pierden reglas de asociación interesantes y se generan falsas reglas de asociación.

Capítulo 6

Conclusiones, aportaciones y trabajo futuro

Descubrir patrones frecuentes ha sido y continua siendo una tarea que ayuda al análisis de datos y a la toma de decisiones. Adicionalmente, descubrir patrones frecuentes es una etapa previa de otras tareas de minería de datos como son la minería de reglas de asociación, la clasificación y el agrupamiento. Particularmente para la minería de reglas de asociación, descubrir patrones frecuentes es generalmente la etapa más costosa.

Mucho se ha avanzado en cuanto a mejorar la eficiencia de los algoritmos de minado de patrones frecuentes del enfoque tradicional, en el cual las descripciones de los objetos son comparadas teniendo en cuenta si son o no exactamente iguales. Sin embargo, en las ciencias poco formalizadas los objetos de estudio comúnmente no son comparados de esta forma, sino usando el concepto de analogía o semejanza. Cuando en un problema real los especialistas del área emplean una función de semejanza diferente de la igualdad y la igualdad es usada en la búsqueda de los los patrones frecuentes para comparar las descripciones de los objetos y calcular su frecuencia, pueden perderse patrones frecuentes. Como consecuencia, en dependencia de para qué es usada esta información puede desvirtuarse el análisis, o tomarse malas decisiones, o generarse falsas reglas de asociación y perderse otras, o afectarse la calidad de las tareas de clasificación o agrupamiento.

Anterior a esta tesis sólo había sido desarrollado el algoritmo *ObjectMiner* para minar patrones frecuentes usando funciones de semejanza diferentes de la igualdad, y a partir de dichos patrones encontrar reglas de asociación interesantes. Sin embargo, *ObjectMiner* restringe su uso a funciones de semejanza Booleana para las cuales se cumple que si una descripción de un objeto no es frecuente ninguna superdescripción de ésta es frecuente. Además, otro tipo de funciones de semejanza tanto Booleanas como no Booleanas que no satisfacen dicha restriction son comúnmente usadas en el trabajo diario de especialistas prácticos. Por lo tanto, esta tesis se ha enfocado en diseñar algoritmos de minado de patrones similares frecuentes (*STreeDC-Miner*, *STreeNDC-Miner*, *RP-Miner*, *STree***DC-Miner*, *STree***NDC-Miner* y *RP***-Miner*) que cubran el espacio de posibles funciones de semejanza tanto Booleanas como no Booleanas, y en diseñar un algoritmo (*FSP-*

GenRules) que permita minar las reglas de asociación interesantes a partir de los patrones similares frecuentes encontrados. Los algoritmos *STreeDC-Miner*, *STreeNDC-Miner* y *RP-Miner* fueron diseñados para minar patrones similares frecuentes con funciones de semejanza Booleana, mientras que *STree*DC-Miner*, *STree*NDC-Miner* y *RP*-Miner* son extensiones de los algoritmos anteriores para funciones de semejanza no Booleana.

6.1. Conclusiones

El comportamiento de los algoritmos propuestos fue explorado y comparado tanto con el algoritmo de minado de patrones similares frecuentes *ObjectMiner* como con el enfoque tradicional de minado de patrones frecuentes. Teniendo en cuenta los experimentos realizados se pudo llegar a las siguientes conclusiones:

- Cuando la función de semejanza cumple la propiedad de f_S-Clausura Descendente, si la misma es Booleana, el algoritmo *STreeDC-Miner* obtiene los mejores resultados en términos de la eficiencia, la eficacia y la calidad de los patrones encontrados; mientras que si la función de semejanza no es Booleana, el algoritmo *STree*DC-Miner* obtiene los mejores resultados en términos de la eficiencia, la eficacia y la calidad de los patrones encontrados.

- Cuando la función de semejanza no cumple la propiedad de f_S-Clausura Descendente, si la misma es Booleana, el algoritmo *RP-Miner* obtiene los mejores resultados en términos de la calidad de los patrones encontrados; mientras que si la función de semejanza no es Booleana, el algoritmo *STree*NDC-Miner* obtiene los mejores resultados en términos de la calidad de los patrones encontrados.

- Cuando la función de semejanza no cumple la propiedad de f_S-Clausura Descendente y es de interés maximizar la eficacia, si la función de semejanza es Booleana, el algoritmo *STreeNDC-Miner* obtiene los mejores resultados de eficacia, mientras que si la función de semejanza no es Booleana, el algoritmo *STree*NDC-Miner* obtiene los mejores resultados de eficacia.

- Al usar funciones de semejanza diferentes de la igualdad para comparar las descripciones de los objetos, la calidad de los patrones similares frecuentes encontrados fue superior que la calidad de los patrones frecuentes encontrados mediante el enfoque tradicional de minado de patrones frecuentes. Por lo tanto, es importante contar con algoritmos que permitan funciones de semejanza diferentes de la igualdad.

- Al usar funciones de semejanza no Booleanas para comparar las descripciones de los objetos, la calidad de los patrones similares frecuentes encontrados fue superior que la calidad de los patrones similares frecuentes encontrados mediante la Booleanización de las mismas. Por lo tanto, es importante contar con algoritmos que permitan funciones de semejanza no Booleanas.

- Al sustituir una función de semejanza Booleana diferente de la igualdad, por la igualdad, se pierden reglas de asociación interesantes y se generan falsas reglas de asociación.
- Al Booleanizar una función de semejanza no Booleana, se pierden reglas de asociación interesantes y se generan falsas reglas de asociación.

6.2. Aportaciones del trabajo de investigación

Las aportaciones de este trabajo de investigación son la siguientes:

- Definición de nuevas propiedades y proposiciones que permiten podar el espacio de búsqueda de patrones similares frecuentes cuando las funciones de semejanza Booleana o no Booleana son monótonas no crecientes.
- Un nuevo algoritmo de minado de patrones similares frecuentes (*STreeDC-Miner*) para funciones de semejanza Booleana monótonas no crecientes, basado en las propiedades de poda.
- Nuevo algoritmo de minado de patrones similares frecuentes (*STreeNDC-Miner*) para funciones de semejanza Booleana que no son monótonas no crecientes.
- Un nuevo algoritmo de minado de patrones similares frecuentes (*RP-Miner*) para funciones de semejanza Booleana que no son monótonas no crecientes, basado en un relajamiento el mecanismo de poda de *STreeDC-Miner*.
- Un nuevo algoritmo de minado de patrones similares frecuentes (*STree***DC-Miner*) para funciones de semejanza no Booleana monótonas no crecientes, basado en las propiedades de poda.
- Un nuevo algoritmo de minado de patrones similares frecuentes (*STree***NDC-Miner*) para funciones de semejanza no Booleana que no son monótonas no crecientes.
- Un nuevo algoritmo de minado de patrones similares frecuentes (*RP***-Miner*) para funciones de semejanza no Booleana que no son monótonas no crecientes, basado en un relajamiento del mecanismo de poda de *STree***DC-Miner*.
- Adaptación del algoritmo de minado de reglas de asociación Binarias *GenRules* al minado de reglas de asociación incorporando el concepto de semejanza Booleana y no Booleana entre descripciones y subdescripciones de objetos con datos mezclados (*FSP-GenRules*).

6.3. Trabajo futuro

En esta tesis se han propuesto varios algoritmos de minado de patrones similares frecuentes, 2 algoritmos eficientes para funciones de semejanza monótonas no crecientes, los cuales podan el espacio de búsqueda de patrones similares frecuentes y 4 para funciones de semejanza no monótonas no crecientes, 2 de los cuales podan relajadamente el espacio de búsqueda y por tanto pueden perder patrones similares frecuentes y 2 que no podan dicho espacio y por tanto su eficiencia se ve afectada.

Un punto importante que con esta tesis aún no se ha resuelto, y que por tanto se propone como trabajo futuro, es diseñar algoritmos eficientes que poden el espacio de búsqueda sin perder patrones similares frecuentes cuando la función de semejanza no es monótona no creciente.

Por otro lado, es sabido que el número de patrones frecuentes encontrados por los algoritmos del enfoque tradicional de minado de patrones frecuentes puede ser muy grande y como consecuencia el análisis de estos patrones por parte de los humanos puede dificultarse. Para atacar este problema, en la literatura se han reportado varios trabajos en los cuales el número de patrones similares frecuentes encontrados se reduce. Fundamentalmente han sido 2 los subconjuntos de patrones frecuentes que son buscados: los patrones frecuentes maximales y los patrones frecuentes cerrados. Cuando se usan funciones de semejanza para el cálculo de las frecuencias de los patrones, este problema se acentúa pues entre las ocurrencias de los patrones, se contarán también las ocurrencias de sus semejantes y con ello puede aumentar la frecuencia y por tanto la cantidad de patrones similares frecuentes.

Por tanto, como trabajo futuro se propone diseñar algoritmos eficientes de minado de patrones similares frecuentes maximales y de patrones similares frecuentes cerrados.

Otro trabajo futuro que se deriva de la presente tesis es el diseño de clasificadores basados en patrones similares frecuentes, pues los resultados alcanzados mediante un clasificador simple para medir la calidad de los patrones encontrados por los algoritmos de minado de patrones similares frecuentes evidencian que al emplear este tipo de patrones pueden obtenerse resultados superiores a los alcanzados usando los patrones frecuentes del enfoque tradicional.

Anexos

Notaciones

$\wedge$	Operador lógico de conjunción
$\vee$	Operador lógico de disjunción
$\subseteq$	Subconjunto
$\cap$	Intersección de conjuntos
$\cup$	Unión de conjuntos
$\in$	Pertenencia
$\|\|$	Operador de cardinalidad
$\emptyset$	Conjunto vacío
$\forall$	Cuantificador universal
$\exists$	Cuantificador existencial
$\equiv$	Equivalencia lógica
$\Rightarrow$	Implicación lógica
$\mu_A(a)$	Grado de pertenencia del elemento a al conjunto difuso A
Ω	Colección de datos mezclados
R	Conjunto de atributos que describen a los objetos en Ω
$O[r]$	Valor del atributo r en el objeto O
S	Subconjunto del conjunto de atributos R
$\hat{S}$	Superconjunto del conjunto de atributos S
$\check{S}$	Subconjunto del conjunto de atributos S
$I_S(O)$	Proyección de los valores de O en términos de los atributos en S
$f_S(O, O^{'})$	Semejanza entre las subdescripciones $I_S(O)$ y $I_S(O^{'})$
$f_S freq(O)$	Frecuencia de la subdescripción $I_S(O)$
$minFreq$	Umbral de mínima frecuencia
$f_S conf(I_{S_1}(O) \rightarrow I_{S_2}(O))$	Confianza de la regla de asociación $I_{S_1}(O) \rightarrow I_{S_2}(O)$
$minConf$	Umbral de mínima confianza
β_S	Umbral parcial de semejanza entre las subdescripciones de objetos respecto al subconjunto de atributos S de R

β	Umbral de mínima semejanza
$I_S(O).\mathcal{S}$	Para semejanza Booleana: Conjunto de subdescripciones de objetos a las cuales $I_S(O)$ es estrictamente semejante. Para semejanza no Booleana: Conjunto de pares (*subdescripción, semejanza*) tal que $I_S(O)$ es estrictamente semejante a *subdescripción* con grado mayor o igual que el umbral de semejanza β
$\mathcal{I}_S$	Conjunto de subdescripciones (no idénticas) de objetos respecto al subconjunto de atributos S de R
$STree_S$	Estructura de datos arbórea, donde cada camino desde la raíz hasta una hoja, representa una subdescripción de un objeto de Ω respecto al subconjunto de atributos S de R
$STree_S.I_S(O)$	Subdescripción de un objeto de Ω respecto al subconjunto de atributos S contenida en $STree_S$
$STree_S.I_S(O).\tilde{c}$	Número de subdescripciones semejantes a $I_S(O)$ que no son iguales a $I_S(O)$
$STree_S.I_S(O).\mathcal{O}$	Conjunto de objetos de Ω que contienen a la subdescripción $I_S(O)$
$STree_S.I_S(O).\mathcal{S}$	Conjunto de subdescripciones contenidas en $STree_S$ a las cuales $I_S(O)$ es semejante pero no igual
$STree^*_S$	Extensión de la estructura $STree_S$ para permitir manipular semejanzas no Booleanas
$STree^*_S.I_S(O)$	Subdescripción de un objeto de Ω respecto al subconjunto de atributos S contenida en $STree^*_S$
$STree^*_S.I_S(O).\tilde{c}$	Número de subdescripciones semejantes a $I_S(O)$ que no son iguales a $I_S(O)$
$STree^*_S.I_S(O).\mathcal{O}$	Conjunto de objetos de Ω que contienen a la subdescripción $I_S(O)$
$STree^*_S.I_S(O).\mathcal{S}$	Conjunto de pares $(subdescripción, semejanza)$ tales que el grado de semejanza *semejanza* de $I_S(O)$ a *subdescripción* es mayor o igual que β, pero $I_S(O)$ y *subdescripción* no son iguales
null	Apuntador nulo
W	Conjunto de subdescripciones, de tamaño mayor o igual que 1, analizadas
F	Conjunto de patrones similares frecuentes encontrados
RA	Conjunto de reglas de asociación interesantes encontradas

Publicaciones

Los artículos publicados derivados de este trabajo de investigación son los siguientes:
Congresos:

1. A.Y. Rodríguez-González, J.F. Martínez-Trinidad, J.A. Carrasco-Ochoa, J. Ruiz-Shulcloper. **Mining Frequent Similar Patterns on Mixed Data**. In *Proceedings of CIARP 2008*, LNCS 5197, pp. 136-144, © Springer-Verlag Berlin Heidelberg, 2008.

2. A.Y. Rodríguez-González, J.F. Martínez-Trinidad, J.A. Carrasco-Ochoa, J. Ruiz-Shulcloper. **Using Non Boolean similarity Functions for Frequent Similar Pattern Mining.**. In *Proceedings of 23th Canadian Conference on Artificial Intelligence 2010*, LNCS 6085, pp. 374-378, © Springer-Verlag Berlin Heidelberg, 2010.

Revistas JCR (*Journal Citation Reports*):

3. A.Y. Rodríguez-González, J.F. Martínez-Trinidad, J.A. Carrasco-Ochoa, J. Ruiz-Shulcloper. **RP-Miner: A Relaxed Prune Algorithm for Frequent Similar Pattern Mining**. To appear in the Journal Knowledge and Information System, 2010. (Available online) DOI: 10.1007/s10115-010-0309-9.

En preparación:

4. A.Y. Rodríguez-González, J.F. Martínez-Trinidad, J.A. Carrasco-Ochoa, J. Ruiz-Shulcloper. **Mining Frequent Patterns and Association Rules using Similarities**.

5. A.Y. Rodríguez-González, J.F. Martínez-Trinidad, J.A. Carrasco-Ochoa, J. Ruiz-Shulcloper. **Softening Frequent Similar Pattern Mining Using Non Boolean similarity Functions**.

Referencias

R. Agrawal and R. Srikant. Fast Algorithms for Mining Association Rules. *In Proceedings of the 1994 International Conference on Very Large Data Bases (VLDB'94), Santiago, Chile*, pages 487–499, 1994.

R. Agrawal, T. Imielinski, and A.N. Swami. Mining Association Rules Between Sets of Items in Large Databases. *In Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, Washington, DC*, pages 207–216, 1993.

S. Ahmed and F. Coenen. Tree-based partitioning of data for association rule mining. *Knowledge and Information Systems Journal*, 10(3):315–331, 2006.

B. Alatas, E. Akin, and A. Karci. MODENAR: Multi-objective differential evolution algorithm for mining numeric association rules. *Applied Soft Computing*, 8(1):646–656, 2008.

C. Borgelt. Efficient implementations of apriori and eclat. *In Proceedings of the IEEE ICDM Workshop on Frequent Itemset Mining Implementation (FIMI'03), Melbourne, Florida, USA*, 2003.

S. Born and L. Schmidt-Thieme. Optimal Discretization of Quantitative Attributes for Association Rules. *In Proceedings of the Meeting of the International Federation of Classification Societies (IFCS), Chicago, USA*, pages 287–296, 2004.

G. Chen and Q. Wei. Fuzzy association rules and the extended mining algorithms. *Information Sciences*, 147(1–4):201–228, 2002.

J.M. De-Graaf, W.A. Kosters, and et al. Interesting Fuzzy Association Rules in Quantitative Databases. *In Proceedings of the 5th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD 2001), Freiburg, Germany*, pages 140–151, 2001.

R. Dánger, J. Ruiz-Shulcloper, and R. Berlaga. Objectminer: A New Approach for Mining Complex Objects. *In Proceedings of the 6th International Conference on Enterprise Information Systems (ICEIS'2004), Oporto, Portugal*, pages 42–47, 2004.

A. Erwin, R.P. Gopalan, and et al. A bottom-up projection based algorithm for mining high utility itemsets. *In Proceedings of the 2nd International Workshop on Integrating Artificial Intelligence and Data Mining (AIDM '07), Gold Coast, Australia*, 2007.

Z. Farzanyar, M. Kangavari, and et al. A New Algorithm for Mining Fuzzy Association Rules in the Large Databases Based on Ontology. *In Proceedings of the Sixth IEEE International Conference on Data Mining - Workshops (ICDMW'06), Hong Kong, China*, pages 65–69, 2006.

T. Fukuda, Y. Morimoto, and et al. Mining optimized association rules for numeric attributes. *In Proceedings of the Fifteenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of Database Systems, Montreal, Quebec, Canada*, 1996.

L. Geng and H.J. Hamilton. Interestingness maesure for data mining: a Survey. *ACM Computing Surveys*, 38(6):9, 2006.

J. Gómez, O. Rodríguez, S. Valladares, J. Ruiz-Shulcloper, and et al. Prognostic of Gas-oil Deposits in the Cuban Ophiological Association, Applying Mathematical Modeling. *Geophys. Int.*, 33(3):447–467, 1994.

R.P. Gopalan and Y.G. Sucahyo. High Performance Frequent Patterns Extraction using Compressed FP-Tree. *In Proceedings of the International Workshop on High Performance and Distributed Mining (SIAM'2004), Orlando, USA.*, 2004.

G. Grahne and J. Zhu. Fast Algorithms for Frequent Itemset Mining Using FP-Trees. *IEEE Trans. on Knowl. and Data Eng.*, 17(10):1347–1362, 2005.

A. Gyenesei. A Fuzzy Approach for Mining Quantitative Association Rules. *Technical Report, Turku Centre for Computer Science*, 2000.

J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. *In Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, Dallas, Texas*, pages 1–12, 2000.

R. Hernández-León, J. Hernánde-Palancar, J.A. Carrasco-Ochoa, and J.F Martínez-Trinidad. Algorithms for mining frequent itemsets in static and dynamic dataset. *Intelligent Data Analysis*, 14(3):419–435, 2010.

J.D. Holt and S.M. Chung. Efficient mining of association rules in text databases. *In Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data, Kansas City, Missouri, USA*, 1999.

J.D. Holt and S.M. Chung. Multipass algorithms for mining association rules in text databases. *Knowledge and Information Systems*, 3(2):168–183, 2001.

J.D. Holt and S.M. Chung. Mining association rules using inverted hashing and pruning. *Information Processing Letters*, 83(4):211–220, 2002.

T. Hong and Y. Lee. An overview of mining fuzzy association rules. *Studies in Fuzziness and Soft Computing*, 220:397–410, 2008.

T. Huand, S.Y. Sung, H. Xiong, and Q. Fu. Discovery of maximum length frequent itemsets. *Inf. Sci.*, 178(1):69–87, 2008.

A. Fuentes-Rodríguez J. Ruiz-Shulcloper. A cybernetic model to analyze juvenile delinquency. *Revista Ciencias Matemticas*, 2(1):141–153, 1981.

L. Jian-min and W. Xiao-ding. Research on User Groups Features of Mobile Payment: An Empirical Analysis Based on Association Rules of Data Mining. *In Proceedings of the 2010 International Conference on E-Business and E-Government, Guangzhou, China*, pages 129–134, 2010.

B. Kalpana and R. Nadarajan. Incorporating heuristics for efficient search space pruning in frequent itemset mining strategies. *Current Science*, 94(1):97–101, 2008.

F. Karel. Quantitative and Ordinal Association Rules Mining (QAR Mining). *In Proceedings of the 10th International Conference on Knowledge-Based and Intelligent Information and Engineering Systems (KES 2006), South Coast, UK*, 2006.

K. Ke, J.C., and W. Ng. MIC framework: an information-theoretic approach to quantitative association rule mining. *In Proceedings of the ICDE '06*, pages 112–114, 2006.

M. Kim, G.H. Kim, and et al. A Virtual Join Algorithm for Fast Association Rule Mining. *In Proceedings of the 4th International Conference on Intelligent Data Engineering and Learning (IDEAL 2003), Hong Kong, China*, 2003.

C.M. Kuok, A. Fu, and et al. Mining fuzzy association rules in databases. *BMC Bioinformatics*, 27(1):41–46, 1998.

C. LaRosa, L. Xiongand, and K. Mandelberg. Frequent pattern mining for kernel trace data. *In Proceedings of the 2008 ACM symposium on Applied computing (SAC'08), Fortaleza, Ceara, Brazil*, pages 880–885, 2008.

J.H. Lee and H. Lee-Kwang. An extension of association rules using fuzzy sets. *In Proceedings of the Seventh IFSA World Congress (IFSA'97), Prague, Czech Republic*, 1997.

F.J. López, A. Blanco, F. Garcia, C. Pino, and A. Marin. Fuzzy association rules for biological data analysis: A case study on yeast. *BMC Bioinformatics*, 9(107), 2008.

J.F. Martínez-Trinidad, J. Ruiz-Shulcloper, and M.S. Lazo-Cortés. Structuralization of Universes. *Fuzzy Sets*, 112(3), 2000.

J. Mata, J.L.A. Macías, and et al. Discovering Numeric Association Rules via Evolutionary Algorithm. *In Proceedings of the 6th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining (PAKDD 2002), Taipei, Taiwan*, 2002a.

J. Mata, J.L.A. Macías, and et al. An evolutionary algorithm to discover numeric association rules. *In Proceedings of the 2002 ACM Symposium on Applied Computing (SAC'2002), Madrid, Spain*, 2002b.

R.J. Miller and Y. Yang. Association rules over interval data. *In Proceedings of the 1997 ACM SIGMOD international conference on Management of Data, Tucson, Arizona, USA*, 2002.

L. Nan, Z. Chun-Guang, and C. Lai-Zhong. The Application of Association Rules Algorithm On Web Search Engine. *In Proceedings of the 2009 International Conference on Computational Intelligence and Security, Beijing, China*, pages 102–108, 2009.

M.R. Ortiz-Posadas, L. Vega-Alvarado, and B. Toni. A mathematical function to evaluate surgical complexity of cleft lip and palate. *Comput. Methods Prog. Biomed.*, 94(3): 232–238, 1994.

S. Papadimitriou and S. Mavroudi. The fuzzy frequent pattern Tree. *In Proceedings of the 9th WSEAS International Conference on Computers, Athens, Greece*, 2005.

J.S. Park, M.S. Chen, and P.S. Yu. Using a Hash-Based Method with Transaction Trimming for Mining Association Rules. *IEEE Transaction on Knowledge and Data Engineer*, 9(5):813–825, 1997.

B.M. Patil, R.C. Joshi, and D. Toshniwal. Association Rule for Classification of Type-2 Diabetic Patients. *In Proceedings of the 2010 Second International Conference on Machine Learning and Computing, Bangalore, India*, pages 330–334, 2010.

A. Pietracaprina and D. Zandolin. Mining Frequent Itemsets Using Patricia Tries. *In Proceedings of the IEEE ICDM Workshop on Frequent Itemset Mining Implementation (FIMI'03), Melbourne, Florida, USA*, 2003.

J. Ruiz-Shulcloper. Pattern Recognition with Mixed and Incomplete Data. *Journal Pattern Recognition and Image Analysis*, 18(4):563–576, 2009.

A. Salleb-Aouissi, C. Vrain, and C. Nortet. QuantMiner: A Genetic Algorithm for Mining Quantitative Association Rules. *In Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI'07), Hyderabad, India*, pages 1035–1040, 2007.

A. Savasere, E. Omiecinski, and S.B. Navathe. An Efficient Algorithm for Mining Association Rules in Large Databases. *In Proceedings of the 21th International Conference on Very Large Data Bases (VLDB'95), Zurich, Switzerland*, pages 432–444, 1995.

M. Serrurier, D. Dubois, and et al. Learning fuzzy rules with their implication operators. *Data and Knowledge Engineer*, 60((1)):71–89, 2007.

B. Shen, M. Yao, Z. Wu, and Y. Gao. Mining dynamic association rules with comments. *Knowledge and Information Systems*, 24(1):73–98, 2010.

W. Shitong, K.F.L. Chung, and et al. Fuzzy taxonomy, quantitative database and mining generalized association rules. *Intelligent Data Analysis*, 9(2):207–217, 2005.

M. Song and S. Rajasekaran. A Transaction Mapping Algorithm for Frequent Itemsets Mining. *IEEE Transactions on Knowledge and Data Engineering*, 18(4):472–481, 2006.

R. Srikant and R. Agrawal. Mining quantitative association rules in large relational tables. *In Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data (SIGMOD'95), Montreal, Quebec, Canada*, pages 1–12, 1996.

Y.G. Sucahyo and R.P. Gopalan. CT-ITL: efficient frequent item set mining using a compressed prefix tree with pattern growth. *In Proceedings of the 14th Australasian Database Conference (ADC 2003), Adelaide, Australia*, 2003.

Y.G. Sucahyo and R.P. Gopalan. CT-PRO: A Bottom-Up Non Recursive Frequent Itemset Mining Algorithm Using Compressed FP-Tree Data Structure. *In Proceedings of the IEEE ICDM Workshop on Frequent Itemset Mining Implementation (FIMI'04), Brighton, UK*, 2004.

W. Takashi, M. Yuki, and et al. Mining Quantitative Frequent Itemsets Using Adaptive Density-Based Subspace Clustering. *In Proceedings of the Fifth IEEE International Conference on Data Mining (ICDM'05), Houston, Texas, USA*, 2005.

H. Tzung-Pei, L. Kuei-Ying, and et al. Fuzzy data mining for interesting generalized association rules. *Fuzzy Sets and System*, 138(2):255–269, 2003.

L. Yunyan and C. Juan. Application of Association Rules Mining in Marketing Decision-Making Based on Rough Set. *In Proceedings of the 2010 International Conference on E-Business and E-Government, Guangzhou, China*, pages 3749–3752, 2010.

L.A. Zadeh. Fuzzy Sets. *Information and Control*, 8(3):338–353, 1965.

M. Zaki, S. Parthasarathy, and W. Li. New Algorithms for Fast Discovery of Association Rules. *Technical Report TR651, University of Rochester*, 1997.

M. Zhang, B. Kao, D.W. Cheung, and K.Y. Yip. Mining periodic patterns with gap requirement from sequences. *ACM Transactions on Knowledge Discovery from Data*, 1(2):7, 2007.

Z. Zhang, Y. Lu, and et al. An Effective Partitioning-Combining Algorithm for Discovering Quantitative Association Rules. *In Proceedings of the First Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD-97)*, 1997.