# Propuesta de un método para la obtención de Dependencias Funcionales basado en Dualidad de Hipergrafos

Joel Fuentes, Pablo Sáez y Gilberto Gutiérrez

Departamento de Ciencias de la Computación
y Tecnologías de la Información,
Universidad del Bío-Bío, Chillán, Chile
{jfuentes, psaezg, ggutierr}@ubiobio.cl

**Resumen** Una de las principales etapas en la obtención del modelo relacional a partir de sistemas heredados es la extracción de las dependencias funcionales por medio de técnicas de minería de datos. Se han propuesto variados métodos para este fin, siendo las soluciones más conocidas exponenciales en tiempo en el número de atributos de la relación. Pero en situaciones reales es frecuente encontrar casos en que la cantidad de atributos es alta (más de 20 ó 30 atributos). La presente propuesta está enfocada a resolver este problema utilizando el conocimiento del que se dispone en la actualidad en relación con el problema de dualidad de hipergrafos, para el cual se conocen algoritmos cuasi-polinomiales ($O(n^{\log n})$), que hasta ahora no han sido comúnmente considerados para resolver el problema de la obtención de dependencias funcionales. Se puede mostrar que este problema, si se parte de las refutaciones para estas dependencias existentes en los conjuntos de datos de los sistemas heredados, es equivalente al mencionado problema de dualidad de hipergrafos. En concreto, el objetivo de esta investigación es estudiar los algoritmos, tanto para la obtención de dependencias funcionales como de teoría de hipergrafos, para luego proponer una herramienta de la cual se pueden beneficiar los procesos de migración de los sistemas heredados. Se espera que esta herramienta haga posible el procesamiento de relaciones con gran cantidad de atributos, que escapa a las herramientas disponibles en la actualidad.

## 1. Introducción

El descubrimiento de dependencias funcionales (DF de aquí en adelante) desde una instancia de una relación es una importante técnica en la minería de datos. Es utilizada en el diseño de base de datos, optimización de consultas, ingeniería inversa, entre otros [4]. También cumple una labor fundamental en el ámbito de los sistemas de información heredados. Es en este último ámbito donde se enfoca nuestra investigación, con el objetivo de proponer una herramienta

capaz de extraer las DFs de forma automática desde estos sistemas. Estudios como [11,18] proponen métodos y estrategias para obtener el modelo de base de datos relacional desde sistemas heredados, siendo una de las etapas la obtención automática de las DFs, lo que destaca la importancia de contar con herramientas eficientes para este fin.

Actualmente existen algoritmos y herramientas que pueden realizar esta labor, pero las soluciones más difundidas requieren un tiempo exponencial en el número de atributos de la relación. Es por ello que nace la inquietud de intentar mejorar las técnicas hasta ahora utilizadas por estos algoritmos, aprovechando el conocimiento que se ha generado en la última década en relación con el problema de dualidad de hipergrafos, que está intimamente relacionado con el de extracción de DFs. Para el problema de dualidad de hipergrafos se conocen algoritmos de complejidad $O(n^{\log n})$, es decir sub exponenciales (o cuasi-polinomiales).

La presente propuesta se divide en tres secciones. En la sección 2 se expone la definición del problema y la primera discusión sobre los trabajos relacionados, para luego en la sección 3 describir el enfoque propuesto.

## 2. Definición del Problema y trabajos relacionados

Sea $R$ un conjunto de atributos de una relación. Una DF es una expresión $X \rightarrow A$, donde $X \subseteq R$ y $A \in R$, siendo $X$ el determinante y $A$ el atributo determinado. La dependencia es válida en una instancia $r$ de $R$ si para todos los pares de filas (tuplas) $t, u \in r$ tenemos: si $t[B] = u[B]$ para todos los atributos $B \in X$, entonces $t[A] = u[A]$ [9]. Entonces, el problema consiste en: *dada una instancia $r$ de $R$, encontrar todas las dependencias funcionales no triviales que se satisfacen en $r$.*

Si tenemos un conjunto de atributos $R$, por ejemplo $\{A, B, C, D, E, F\}$, el espacio de búsqueda de determinantes de DFs de la forma $X \rightarrow U$, para un $U \in R$ determinado, es exponencial en el tamaño de $R$ (llamémoslo $n$), ya que se deben formar $2^{n-1} - 1$ combinaciones, descartando el nivel 0 (en el ejemplo $2^5 - 1 = 31$ posibles subconjuntos de atributos candidatos). En la Figura 1 se muestran todas las posibles combinaciones si tomamos $U = F$. Estas combinaciones forman un *retículo booleano.*

Los algoritmos que se han propuesto para resolver este problema se pueden clasificar en tres categorías o enfoques [20]: el primer enfoque es el de generación-prueba de DFs candidatas, donde descatan los algoritmos llamados TANE [9] y FUN [16], el segundo es el enfoque de Minimal Cover, donde destacan los algoritmos propuestos por Flach et al. [7], Lopes et al. [12] y Wyss et al. [19], y finalmente el enfoque Formal Concept Analysis con los algoritmos de Baixeries [1] y el de Lopes et al. [13]. El funcionamiento de TANE se basa en la búsqueda de DFs por niveles dentro del retículo booleano, obtención de DFs aproximadas de acuerdo a un factor de error y la utilización de técnicas de pruning para optimizar la búsqueda. El problema es que su rendimiento es bajo cuando en una relación el conjunto $R$ es grande (por ejemplo, más de 20 ó 30 atributos), tal como lo muestran los resultados realizados en [9,21]. Esto, debido a que el tamaño

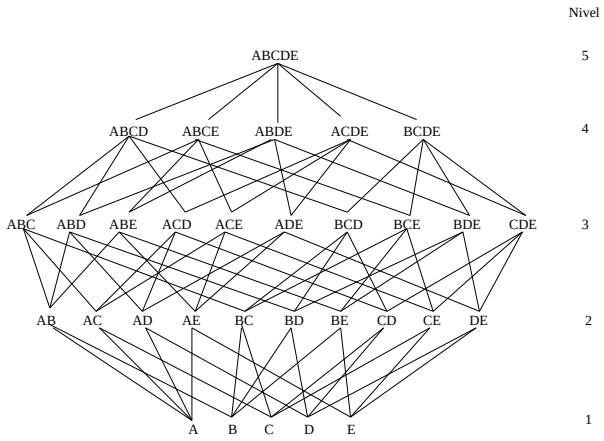


**Figura 1.** Posibles combinaciones de los atributos A, B, C, D y E

del retículo booleano crece exponencialmente con la cantidad de atributos de la relación. Los otros métodos mencionados tienen problemas similares.

El objetivo de nuestra investigación es proponer una herramienta de software basada en algoritmos de dualidad de hipergrafos para la extracción de DFs desde sistemas de información heredados. Con esta herramienta se espera obtener mejor rendimiento que los algoritmos conocidos cuando el conjunto de atributos $R$ es grande, considerando que en el estudio de dualidad de hipergrafos se han propuesto variados algoritmos, tales como [3,5,8,10,14], que utilizaremos en nuestra investigación, además de considerar los últimamente propuestos por Murakami y Uno en [15] y que muestran mejor rendimiento que los antes mencionados.

## 3. Enfoque propuesto

### 3.1. Definiciones y Notaciones

Dado un conjunto de atributos $R$ y una instancia $r$ de $R$, la expresión $X \twoheadrightarrow A$ donde $X \subseteq R$ y $A \in R$, indica que existen dos tuplas $t$ y $u$ en $r$, tales que $\forall(B \in X)\ t[B] = u[B] \wedge t[A] \neq u[A]$.

Un *hipergrafo* se define como un grafo generalizado $H = (A, E)$, donde $A$ es un conjunto finito y $E \subseteq \mathcal{P}(A)$ un conjunto de hiperaristas ($\mathcal{P}(C)$ es el conjunto de partes de $C$). Este concepto fue propuesto por Claude Berge en 1970 [2] y se puede considerar como una generalización del concepto de grafo, en el sentido de que no se requiere que las aristas tengan siempre dos nodos.

Dado un hipergrafo $H$ se pueden definir los siguientes operadores[1]:

- $\mu(H) = \{X \in H; \neg\exists Y \in H, Y \subset X \wedge X \neq Y\}$ es el conjunto de hiperaristas minimales de $H$.

[1] Estamos adoptando esencialmente la notación propuesta en [17].

- $\nu(H) = \{X \subseteq A; \exists Y \in H, Y \subset X\}$ es el conjunto de los $X$ que son respondidos por $H$.
- $\tau(H) = \{X \subseteq A; \forall Y \in H, X \cap Y \neq \emptyset\}$ es el conjunto de *transversales* de $H$; serían los modelos de $H$, si consideráramos a los elementos de $H$ como cláusulas.
- $\lambda(H) = \mu(\tau(H))$ es el conjunto de transversales minimales.
- $\nu'(H) = \{X \subseteq A; \exists Y \in H, X \subseteq Y\}$ es el conjunto de subconjuntos de elementos de $H$.

El problema de dualidad de hipergrafos consiste en: dados $H, K$ hipergrafos minimales, decidir si $K = \lambda(H)$. Para este problema Fredman y Khachiyan propusieron en 1996 un algoritmo sub-exponencial [8], lo que permite conjeturar que no pertenece a la clase $NP$-completo. Pero tampoco se sabe si pertenece a la clase $P$, o al menos $NP$ [6].

### 3.2. Método previsto

A continuación, se describe de forma general el enfoque propuesto para la extracción de DFs desde una instancia $r$ de una relación, basado en refutaciones de DFs e hipergrafos: para un atributo $D \in R$,

1. Obtener el conjunto de refutaciones de DFs de la forma $X \nrightarrow D$, con $X \subset R$; es decir, el conjunto de pares de tuplas $(t, u) \in r$ tales que $(\forall V \in X)t[V] = u[V] \wedge t[D] \neq u[D]$.
2. Obtener las refutaciones maximales, es decir, se obviarán aquellas refutaciones que sean subconjunto de otra (para el mismo consecuente). Por ejemplo, si tenemos las refutaciones $A, B, C \nrightarrow D$ y $A, B \nrightarrow D$, sólamente consideramos $A, B, C \nrightarrow D$. Al considerar todas las refutaciones maximales, se producirá un hipergrafo con hiperaristas maximales. Notar que si $H = \{ x_1, x_2, \ldots, x_k \}$ es el conjunto de refutaciones maximales, entonces todos los subconjuntos de los $x_i$ están refutados. Por ejemplo si $H = \{\{A, B, C\}\}$, es decir si tenemos $\{A, B, C\} \nrightarrow D$ entonces también tenemos $\{A, B\} \nrightarrow D$, $\{B, C\} \nrightarrow D$, etc. Por lo tanto el operador para obtener todas las refutaciones posibles es $\nu'$ (obtener todos los subconjuntos).
3. Consideremos entonces: $H' = \{A \setminus x_1, A \setminus x_2, \ldots, A \setminus x_k\}$. Si tomamos cualquier $Z \in \nu(H')$, entonces $A \setminus Z$ es una DF refutada, por lo tanto el complemento, $\mathcal{P}(A) \setminus \nu(H')$, entrega precisamente todos los $X$ tales que $A \setminus X$ es una DF.
4. Se obtendrán las DFs en forma minimal. Es decir, si tenemos por ejemplo $A \to D$ y $A, B \to D$, entonces sólo consideramos $A \to D$. En forma general si tenemos $X \to D$, todos los $Y \supseteq X$ también determinan a D, por lo tanto solamente conservamos $X$.
5. Finalmente, tomando $\mathcal{P}(A) \setminus \nu(H')$ y conservando las DFs minimales obtenemos un hipergrafo $H''$. Es decir, las DFs minimales son $H'' = \mu(\mathcal{P}(A) \setminus \nu(H')\})$.
6. Se puede demostrar que $H'' = \lambda(H')$.

De esta forma se evitará el hecho de recorrer todo el retículo booleano con todas las combinaciones posibles del conjunto de atributos (mostrado en la Figura 1), que es de tamaño exponencial.

El funcionamiento de este método se realizará sobre instancias en las que los valores de los atributos se encontrarán previamente codificados, de tal forma de no trabajar con los valores reales sino con códigos que utilicen menos espacio en memoria (por ejemplo, números en vez de texto).

*Ejemplo.* Dado el conjunto de atributos $R = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$ y la instancia $r$ que se muestra en la Figura 2 se realiza la búsqueda de refutaciones para el atributo 9.

| **ID Tupla** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** |
|---|---|---|---|---|---|---|---|---|---|
| 1 | $a_1$ | $b_3$ | $c_2$ | $d_1$ | $e_4$ | $f_3$ | $g_5$ | $h_5$ | $i_1$ |
| 2 | $a_1$ | $b_3$ | $c_3$ | $d_3$ | $e_1$ | $f_5$ | $g_3$ | $h_3$ | $i_2$ |
| 3 | $a_2$ | $b_3$ | $c_5$ | $d_1$ | $e_5$ | $f_3$ | $g_2$ | $h_4$ | $i_4$ |
| 4 | $a_3$ | $b_3$ | $c_2$ | $d_3$ | $e_3$ | $f_4$ | $g_1$ | $h_1$ | $i_1$ |
| 5 | $a_4$ | $b_2$ | $c_2$ | $d_8$ | $e_2$ | $f_1$ | $g_4$ | $h_4$ | $i_1$ |
| 6 | $a_5$ | $b_4$ | $c_4$ | $d_1$ | $e_3$ | $f_3$ | $g_1$ | $h_1$ | $i_3$ |
| 7 | $a_1$ | $b_1$ | $c_3$ | $d_7$ | $e_5$ | $f_2$ | $g_4$ | $h_1$ | $i_2$ |

**Figura 2.** Instancia del conjunto de atributos $R = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$

Observamos que las tuplas 1 y 2 producen la refutación $\{1, 2\} \nrightarrow 9$, las tuplas 1 y 3 producen la refutación $\{2, 4, 6\} \nrightarrow 9$, las tuplas 1 y 6 producen la refutación $\{4, 6\} \nrightarrow 9$, etcétera, hasta las tuplas 6 y 7, que producen $\{8\} \nrightarrow 9$. Conservando solamente las refutaciones maximales obtenemos el hipergrafo $H$ ={{1, 2}, {2, 4, 6}, {5, 7, 8}} en el primer paso del método. Luego, obtenemos el hipergrafo de los complementos de las hiperaristas de $H$, que es $H'$ ={{3, 4, 5, 6, 7, 8}, {1, 3, 5, 7, 8}, {1, 2, 3, 4, 6}}. Finalmente el operador $\lambda$ produce las dependencias funcionales: $H'' = \lambda(H')$ ={{1,4},{1,5},{1,6},{1,7},{1,8},{2,5},{2,7},{2,8}, {3},{4,5},{4,7},{4,8},{5,6},{6,7},{6,8}}. Es decir, hemos descubierto las dependencias funcionales $\{3\} \to 9, \{1, 4\} \to 9$, etcétera.

Nótese que las DFs obtenidas a partir de una instancia particular de una relación pueden no resultar siempre válidas en la realidad (es decir que pudieran ser coincidencias de los datos). En una situación real estas deberán ser confirmadas por expertos en el ámbito del negocio.

## Referencias


[1] Baixeries, J.: A formal concept analysis framework to mine functional dependencies. In: Workshop on mathematical mehtods for learning, Como, Italy (2004)

[2] Berge, C.: Graphes et hypergraphes. Dunod (1973)

[3] Boros, E., Elbassioni, K., Gurvich, V., Khachiyan, L.: An efficient incremental algorithm for generating all maximal independent sets in hypergraphs of bounded dimension. Parallel Process Letter 10(4), 253–266 (2000)

[4] Calvanese, D., De Giacomo, G., Lenzerini, M.: Identification constraints and functional dependencies in description logics. In: International Joint Conference on Artificial Intelligence. vol. 17, pp. 155–160 (2001)
[5] Eiter, T.: Exact transversal hypergraphs and application to boolean $\mu$-functions. J. Symb. Comput. 17(3), 215–225 (1994)
[6] Eiter, T., Makino, K., Gottlob, G.: Computational aspects of monotone dualization: A brief survey. Discrete Applied Mathematics 156(11), 2035–2049 (2008)
[7] Flach, P., Savnik, I.: Database dependency discovery: a machine learning approach. AI communications 12(3), 139–160 (1999)
[8] Fredman, M., Khachiyan, L.: On the complexity of dualization of monotone disjunctive normal forms. J. Algorithms 21(3), 618–628 (1996)
[9] Huhtala, Y., Karkkainen, J., Porkka, P., Toivonen, H.: Tane: An efficient algorithm for discovering functional and approximate dependencies. The Computer Journal 42(2), 100 (1999)
[10] Kavvadias, D., Stavropoulos, E.: An efficient algorithm for the transversal hypergraph generation. Journal of Graph Algorithms and Applications 9(2), 239–264 (2005)
[11] Lin, C.: Migrating to relational systems: Problems, methods, and strategies. Contemporary Management Research 4(4), 369–380 (2008)
[12] Lopes, S., Petit, J., Lakhal, L.: Efficient discovery of functional dependencies and armstrong relations. In: Advances in Database Technology — EDBT 2000, vol. 1777, pp. 350–364. Springer Berlin Heidelberg (2000)
[13] Lopes, S., Petit, J., Lakhal, L.: Functional and approximate dependency mining: database and fca points of view. Journal of Experimental & Theoretical Artificial Intelligence 14(2), 93–114 (2002)
[14] Makino, K., Ibaraki, T.: A fast and simple algorithm for identifying 2-monotonic positive boolean functions. Journal of Algorithms 26(2), 291–305 (1998)
[15] Murakami, K., Uno, T.: Efficient algorithms for dualizing large-scale hypergraphs. Arxiv preprint arXiv:1102.3813 (2011)
[16] Novelli, N., Cicchetti, R.: Fun: an efficient algorithm for mining functional and embedded dependencies. In: Database Theory — ICDT 2001, vol. 1973, pp. 189–203. Springer Berlin Heidelberg (2001)
[17] Polyméris, A.: Stability of two player game structures. Discrete Applied Mathematics 156(14), 2636–2646 (2008)
[18] Villagrán, F., Caro, A., Gutiérrez, G.: Definición de un marco de trabajo para la obtención de un modelo de base de datos relacional desde sistemas heredados. Workshop de Tesistas, Jornadas Chilenas de Computación, Antofagasta, Chile (2010)
[19] Wyss, C., Giannella, C., Robertson, E.: Fastfds: A heuristic-driven, depth-first algorithm for mining functional dependencies from relation instances. In: Data Warehousing and Knowledge Discovery, vol. 2114, pp. 101–110. Springer Berlin Heidelberg (2001)
[20] Yao, H., Hamilton, H.: Mining functional dependencies from data. Data Mining and Knowledge Discovery 16(2), 197–219 (2008)
[21] Yao, H., Hamilton, H., Butz, C.: Fd_mine: discovering functional dependencies in a database using equivalences. In: ICDM. pp. 729–732. IEEE Computer Society (2002)