Implementación de un Lenguaje de Consultas Utilizando Lógica de Primer Orden con Cuantificadores de Punto Fijo

Nora Reyes, Alejandro Grosso, Paulino Maldocena, J.M. Turull Torres [1]

{nreyes, agrosso, pmaldo, turull} @unsl.edu.ar

UNIVERSIDAD NACIONAL DE SAN LUIS
Departamento de Informática
Ejército de los Andes 950- Locales 107 y 109
5700 - San Luis
Argentina
Fax: 02652-430224

Resumen

En este trabajo se considera una extensión de la Lógica de Primer Orden, con distintos tipos de cuantificadores de punto fijo, para expresar consultas a Bases de Datos relacionales permitiendo distintos tipos de iteraciones ausentes en primer orden. Esta extensión fue propuesta en [AVV97].

La motivación es que la Lógica de Primer Orden no permite la expresión de ciertos tipos de consultas a Bases de Datos, como quedó demostrado en [CH80]. Por ello intentamos continuar ampliando su poder expresivo para cubrir una mayor parte del conjunto de consultas posibles de computar sobre una Base de Datos. Con esta extensión se pueden realizar consultas a Bases de Datos que están en la clase EXPTIME.

Se muestran aquí algunos aspectos importantes de la implementación de dicho lenguaje, como así también justificación de cotas obtenidas o establecidas.

Con este trabajo se contribuye al análisis de distintos tipos de formalismos, con diferentes grados de expresividad para formular consultas a Bases de Datos relacionales.

Palabras Claves: Lenguajes de Consulta a Bases de Datos, Lógica de Primer Orden, Bases de Datos, Cuantificadores de Punto Fijo.

[1] Universidad Tecnológica Nacional (FRBA); Universidad Nacional de San Luis; Subsidio de Universidad CAECE y Universidad Nacional de Luján. Director del Proyecto de Investigación sobre Teoría de la Computación de la UNSL.

Introducción

Por medio de una *Base de Datos* obtenemos una representación simbólica de una realidad acotada. Para tal representación utilizaremos el *Modelo Relacional*. Como es deseable obtener información de la Base de Datos, es necesario contar con un lenguaje de consultas. Para nuestro modelo, en general, se consideran los lenguajes de consulta como el *Álgebra Relacional* (AR) y el *Cálculo Relacional* (CR), ambos lenguajes son equivalentes en su poder expresivo, y más aún, equivalentes a la *Lógica de Primer Orden* (FO) [Ull88][AV95]. En este trabajo utilizaremos una extensión de FO para la expresión de consultas a las Bases de Datos, dado que el poder expresivo de FO es equivalente a los otros dos lenguajes relacionales.

También presentaremos la definición de la clase de consultas computables, llamada *Computable Queries* (*CQ*) [CH80]. Para ello observamos el *Modelo Relacional* en el marco de la *Teoría de Modelos Finitos* [EF95], tal que el esquema de la Base de Datos Relacional está definido por un vocabulario relacional finito σ, y la instancia de Base de Datos es una σ-estructura, donde cada relación es de la aridad correspondiente para cada símbolo de σ, definida en un dominio finito dado. Así se definen *Clases de Estructuras Relacionales Finitas* para Modelos Relacionales, y se definen sus instancias como *Estructuras Relacionales Finitas*. Entonces, desde este punto de vista, vemos un query como una función cuyo dominio es el conjunto de las estructuras relacionales finitas de un cierto vocabulario, y cuyo codominio es el conjunto de las relaciones definidas en el dominio de la estructura relacional finita correspondiente para alguna aridad, q: $E_{\sigma, fin} \rightarrow E_{\langle R \rangle}$. En [CH80] se propone como definición de la clase de queries computables *CQ* a la clase de funciones definidas en las clases de estructuras relacionales finitas, para cada vocabulario relacional finito, tales que son funciones recursivas parciales en alguna codificación lineal de las estructuras, y preservan isomorfismos en ellas.

Está demostrado que FO (o AR y CR) computan una clase restringida de consultas respecto de la clase *CQ*.

Utilizamos aquí un lenguaje de consultas a Bases de Datos relacionales que permite expresar queries en *FO* extendido con los *cuantificadores Puntos fijos: Determínistico Inflacionario y No Inflacionario* (PFDI y PFDN), *No Determinístico Inflacionario y No Inflacionario* (PFNI y PFNN) *y Alternado Inflaccionario* y *No Inflaccionario* (PFAI y PFAN). Este lenguaje es un subconjunto del que está siendo desarrollado e implementado por nuestro grupo de trabajo, utilizándolo como lenguaje modelo para la expresión y evaluación de consultas a bases de datos relacionales.

En este trabajo se muestra cómo FO aumentado con estos cuantificadores de punto fijo incrementa realmente su poder expresivo, permitiendo resolver más queries de *CQ*, aunque no se alcanza a cubrir la clase completa. Al extender FO, con los cuantificadores de punto fijo mencionados, se consigue capturar las clases que se mencionan sobre *estructuras finitas ordenadas*: FO(PFDI) captura PTIME [Imm86][Var82], FO(PFDN) captura PSPACE [Var82], FO(PFNI) captura NPTIME, FO(PFNN) captura PSPACE, FO(PFAI) captura PSPACE y FO(PFAN) captura EXPTIME [AVV97].

En [RGMT00] se presenta la implementación de una primera extensión a *FO*, y con ella se logra ampliar el poder expresivo de *FO*, pudiéndose capturar la clase NLOGSPACE sobre estructuras finitas ordenadas. Con esta nueva extensión debida a [AVV97] se amplía el lenguaje de consultas considerado anteriormente, se pueden analizar consultas que están en la clase EXPTIME.

Analizamos tipos de queries que no pueden ser expresados en *FO* y que si pueden expresarse en las extensiones de FO propuestas, mostrando la expresividad (o restricciones) en cada caso.

FO, BASES DE DATOS Y QUERIES

Sea $A = \{ (,), , , \wedge, \vee, \neg, \rightarrow, \exists, \forall, x_1, ..., x_n, ... \}$ un alfabeto infinito numerable.

Una *signatura* o *vocabulario relacional* σ es una secuencia finita de símbolos de relación; $\sigma = < R_1, ..., R_m >$. Asociado con cada símbolo de relación R_i está la *aridad* r_i, con $1 \leq i \leq m$.

Se define el *Lenguaje de Primer Orden*, con vocabulario σ, denotado L_σ , al lenguaje construido a partir del alfabeto infinito numerable ($A \cup \sigma$), según las reglas habituales de construcción sintáctica. Para la semántica de los lenguajes *FO* se utilizará la semántica habitual formal de Tarski.

Siempre nos referiremos a lenguajes con igualdad, aunque no haremos explícita la existencia del símbolo de relación de la igualdad en los vocabularios que consideraremos.

Emplearemos los conceptos de *variables libres* y *variables ligadas* en su significación habitual. Llamamos *sentencia* a una fórmula sin variables libres. Usaremos la notación $\varphi(x_1,..., x_r)$ para indicar que las variables libres en la fórmula φ son $x_1,..., x_r$.

Nosotros utilizaremos la Lógica desde la perspectiva de *Teoría de Modelos*.

Denotaremos con *FO* al lenguaje (o lógica) de Primer Orden, en un sentido general, es decir, prescindiendo de qué lenguaje específico de Primer Orden nos estamos ocupando. Lo que nos interesa destacar con esta simbología, es la expresividad o axiomatizabilidad del lenguaje.

Llamamos σ-*estructura* a una estructura que tiene un conjunto y tantas relaciones (de la aridad adecuada), definidas en ese conjunto, como símbolos de relación haya en σ. Sea $\sigma = \langle R_1,..., R_m\rangle$, entonces si *B* es una σ-estructura lo denotaremos de la siguiente manera:

$$B = \langle D^B, R_1{}^B,..., R_m{}^B \rangle$$

Llamamos *dominio* de *B* al conjunto D^B (*dom*(*B*)). Las relaciones $R_1{}^B,..., R_m{}^B$ están definidas en D^B.

Una σ-estructura *B* definida de esta manera es llamada *relacional*, por ser σ una *signatura relacional* y es *finita* si el conjunto *dom*(*B*) es finito. Denotaremos con $E_{\sigma, fin}$ al conjunto de todas las σ-estructuras finitas.

Diremos que *v* es una *valoración* en la σ-estructura *B*, si *v* es una función definida desde el conjunto de variables de individuo de L_σ en el conjunto D^B.

Una *interpretación I* consiste de una σ-estructura *B* y de una valoración *v* en la σ-estructura *B*. Entonces, la interpretación de una fórmula en L_σ consiste de una estructura, de signatura acorde con la del lenguaje en cuestión, y de una función de valoración que sustituye cada variable por un elemento en el dominio de la interpretación; de forma tal de obtener un enunciado o sentencia, del que puede conocerse su valor de verdad. En particular, una sentencia de L_σ es interpretada por una estructura de signatura σ; de modo que, sustituyendo cada símbolo de relación de la sentencia por la relación correspondiente en la estructura y evaluando la sentencia de la forma habitual, sabremos su valor de verdad [EFT84].

La *relación de satisfacción* ($\models$) hace precisa la noción de que una fórmula sea verdadera en una interpretación dada. Si σ es un vocabulario relacional finito, $\varphi \in L_\sigma$ es una fórmula con variables libres $x_1,x_2,...,x_r$, y *B* es una σ-estructura (con $|D^B| = n$) , se denotará con la siguiente expresión que la fórmula φ es verdadera si es interpretada en la estructura *B*, con el elemento $d_{s_j} \in D^B$ asignado a la variable libre x_j, para $1 \leq s_j \leq n$, $1 \leq j \leq r$: $B \models \varphi(x_1,\ldots,x_r)[\, d_{s_1},...,d_{s_r}\,]$.

CONSULTAS O QUERIES COMPUTABLES

Consideramos la Teoría de Modelos Finitos como marco teórico para estudiar las *consultas*, o *queries*, a Bases de Datos. Llamaremos *esquema de Base de Datos* a todo vocabulario relacional σ, e *instancia de Base de Datos* a toda σ-estructura. De esta manera, consideraremos como B*ase de Datos* a toda clase numerable (finita o infinita) de estructuras relacionales finitas de un cierto vocabulario relacional finito.

Siguiendo a [CH80], definiremos como *consulta* o *query* de aridad *r*, para algún entero $r > 0$, a toda función parcial *q* de la siguiente forma, donde σ es un vocabulario: $q : E_{\sigma, fin} \rightarrow E_{\langle R\rangle}$, donde la aridad de *R* es *r*, y tal que para toda σ-estructura *B*, $dom(q(B)) \subseteq dom(B)$. Un *query booleano*, o *propiedad* es una función de la forma: $q : E_{\sigma, fin} \rightarrow \{\bot,\mathsf{T}\}$.

Se dice que q es un *query computable*, si es una función recursiva parcial, en cualquier representación de las σ-estructuras en estructuras totalmente ordenadas, y si preserva isomorfismos.

Denotamos con *CQ* la clase de todos los queries computables. Y diremos que un lenguaje es *completo* si expresa, o computa, exactamente todos los queries en la clase *CQ*.

Si C es una clase de σ-estructuras, diremos que un lenguaje *se comporta como completo con C*, si computa todos los queries de la clase *CQ* restringida al vocabulario σ y a la clase *C*, es decir, si para todo query $q \in CQ$ de la forma q: $E_{\sigma, fin} \rightarrow E_{\langle R \rangle}$, donde R es de aridad r si q es de aridad r, para cualquier entero $r > 0$, computa el query q' que es la restricción de q a la clase C (o sea, q': $E_{\sigma, fin \mid C} \rightarrow E_{\langle R \rangle}$). Y si para todo query $q \in CQ$ de la forma $q : E_{\sigma, fin} \rightarrow \{\bot, \mathsf{T}\}$, donde q es de aridad 0, computa el query q': $E_{\sigma, fin \mid C} \rightarrow \{\bot, \mathsf{T}\}$. También diremos en este caso que la clase de queries definida de este modo es la *clase de queries computables sobre C*.

Notaremos con v_x para el vector de variables $(x_1,\ldots,x_n)$ y la longitud de dicho vector es n. Por abuso de notación, hablaremos igualmente de vectores de términos. Cuando sea necesario aclararemos el tipo de elementos del vector.

Lógicas de punto fijo[2]

INTRODUCCIÓN

Sea $\varphi(v_x, S)$ una fórmula de primer orden en la cual v_x es una r-tupla de variables y S es un símbolo de relación r-aria (no incluida en un vocabulario σ) y sea B una estructura sobre el vocabulario σ. La fórmula φ da lugar a un operador $\Phi(S)$ desde relaciones r-arias sobre el universo D^B de B a relaciones r-arias sobre B, donde $\Phi(T) = \{v_a \mid B \models \varphi(v_x,T)[v_a]\}$, para cada relación r-aria T sobre D^B.

Cada uno de tales operadores $\Phi(S)$ genera una secuencia de *etapas* que son obtenidas iterando $\Phi(S)$. Las etapas Φ^m (también denotadas por φ^m), con $m \geq 1$, de Φ sobre B, están definidas por inducción: $\Phi^1 = \Phi(\varnothing)$, $\Phi^{m+1} = \Phi(\Phi^m)$. Intuitivamente, nos gustaría asociar con un operador $\Phi(S)$ el "límite" de sus etapas. Esto es posible sólo cuando la secuencia Φ^m, con $m \geq 1$, de las etapas "converge", es decir, cuando existe un entero m_0 tal que $\Phi^{m_0} = \Phi^{m_0+1}$ y, de aquí, $\Phi^{m_0} = \Phi^m$, para todo $m \geq m_0$. Notar que en este caso Φ^{m_0} es un punto fijo de $\Phi(S)$, dado que $\Phi^{m_0} = \Phi^{m_0+1} = \Phi(\Phi^{m_0})$. La secuencia de etapas puede no ser convergente. En particular, esto ocurrirá si la fórmula $\varphi(v_x, S)$ no tiene punto fijo.

LÓGICA DE PUNTO FIJO INFLACIONARIA

Una fórmula $\varphi(v_x, S)$ es *inflacionaria en S* si $T \subseteq \Phi(T)$ para cualquier relación r-aria T. En particular, φ es inflacionaria en S si es de la forma $S(v_x) \vee \psi(v_x, S)$. Si φ es inflacionaria en S, entonces la secuencia Φ^m, con $m \geq 1$, de etapas es incrementante. Así, debe tener un "límite".

Más precisamente, si B es una estructura finita con n elementos, entonces existe un entero $m_0 \leq n^r$ tal que $\Phi^{m_0} = \Phi^m$ para cada $m \geq m_0$. Es decir, la secuencia de etapas de $\varphi(v_x,S)$ converge a Φ^{m_0}. Escribimos φ^∞ o Φ^∞ para denotar el punto fijo Φ^{m_0} de φ. *La lógica de Punto Fijo Inflacionaria FO(PFI)* es la lógica de primer orden aumentada con la regla de formación de punto fijo inflacionario para fórmulas inflacionarias[3].

Observación:

Se puede garantizar que el punto fijo también existe cuando la fórmula $\varphi(v_x, S)$ es *positiva* en S, esto es, cuando cada ocurrencia de S está gobernada por un número par de negaciones. En este

[2] Para las siguientes definiciones nos basamos en [AVV97] y en [EF95].

[3] De aquí en adelante, asumimos sin pérdida de generalidad, que cada fórmula inflacionaria es de la forma

$$S(x_1, \ldots\ldots, x_n) \vee \psi(x_1, \ldots\ldots, x_n)$$

caso, la secuencia Φ^m, con $m \geq 1$, de etapas es también incrementante, y consecuentemente tiene un límite que es un punto fijo. De hecho, ese límite es el menor punto fijo de φ. La *lógica de punto fijo positiva* es la lógica de primer orden aumentada con la regla de formación de menor punto fijo para fórmulas positivas. Es fácil ver que la lógica FO(PFI) es tan expresiva como lo es la lógica de punto fijo positiva.

FO(PFI) captura[4] la clase PTIME, pero existen problemas en PTIME , como verificar que la cardinalidad de una estructura sea par, que no son expresables en ella en el caso general [CH82].

LÓGICA DE PUNTO FIJO NO INFLACIONARIA

Podemos obtener lógicas con construcciones de iteración más expresivas que la lógica de punto fijo inflacionaria. Una lógica más poderosa resulta si se iteran operadores de primer orden generales, hasta alcanzar un punto fijo (el cual podría nunca ocurrir). En este caso podemos tener computaciones que nunca terminen.

Sea Φ^m, con $m \geq 1$, la secuencia de etapas del operador $\Phi(S)$ asociado con la fórmula $\varphi(v_x , S)$. Si existe un entero m_0 tal que $\Phi^{m_0} = \Phi^{m_0+1}$, entonces $\varphi^\infty = \Phi^\infty = \Phi^{m_0}$; en otro caso, $\varphi^\infty = \Phi^\infty = \varnothing$.[5] Llamamos φ^∞ al *punto fijo no inflacionario* de φ sobre *B*. La *lógica de Punto Fijo No inflacionaria FO*(*PFN*) es la lógica de primer orden aumentada con la regla de formación de punto fijo no inflacionario para fórmulas de primer orden arbitrarias. Notar que la lógica de punto fijo FO(PFN) es una extensión de FO(PFI). Mientras que las fórmulas FO(PFI) pueden ser evaluadas en tiempo polinomial, FO(PFN) puede expresar problemas PSPACE-completos.

Aunque FO(PFN) captura la clase PSPACE [Var82], el problema de "tener dominio con cardinalidad par" no es expresable en FO(PFN) en el caso general [CH82].

Si FO(PFI) y FO(PFN) tuvieran la misma potencia expresiva, entonces se cumpliría que P=PSPACE. No es claro, sin embargo, que la inversa se debería mantener debido a que FO(PFI) y FO(PFN) necesitan orden para "capturar completamente" a P y a PSPACE respectivamente [AV95].

LÓGICAS DE PUNTO FIJO NO DETERMINÍSTICA

FO(PFI) y FO(PFN) son obtenidas iterando operadores de primer orden inflacionarios y no inflacionarios, respectivamente. En ambos casos, sin embargo, la iteración es secuencial y determinística. No obstante, existen ciertos problemas que no pueden ser descriptos por tales tipos de iteraciones.

Como un ejemplo, consideremos el problema de la *no universalidad* de autómatas finitos sobre un alfabeto binario, el cual es conocido que es PSPACE-completo [GJ79]. Esto es, dado un autómata finito sobre un alfabeto {0,1} queremos conocer si existe una palabra rechazada por el autómata. Una instancia de este problema puede ser vista como una estructura $\langle S, S_0, F, \rho_0, \rho_1 \rangle$, donde S es el conjunto de estados, S_0 es el conjunto con el estado inicial, F es el de estados finales, ρ_o es la relación de transición para el símbolo 0 y ρ_1 es la relación de transición para el símbolo 1. Para verificar si existe una palabra rechazada por el autómata, se debe simplemente adivinar una palabra y comprobar si es rechazada por el autómata. Esto puede ser descripto fácilmente por una *iteración no determinística* de operadores de primer orden.

Sean Φ_1 y Φ_2 operadores de primer orden. Este par de operadores generan *secuencias convergentes* de etapas que son obtenidas aplicando sucesivamente, hasta alcanzar la convergencia, Φ_1 o Φ_2. Esto es, el par genera secuencias de la forma $S_0, S_1, ..., S_m$, donde $S_0 = \varnothing$, $S_{i+1} = \Phi_1(S_i)$ o $S_{i+1} = \Phi_2(S_i)$, y $\Phi_1(S_m) = \Phi_2(S_m) = S_m$. Llamamos a S_m *punto fijo no determinístico local* del par Φ_1,

[4] Recordar que el concepto de *captura* se refiere a que sobre estructuras ordenadas las consultas que son expresables en FO(PFI) coinciden con las consultas en la clase de complejidad PTIME.

[5] Realmente, se muestra en [AV90] que no existe pérdida de generalidad en restringir la atención a operadores de primer orden convergentes.

Φ_2. Notar que el par Φ_1, Φ_2 puede tener más que un punto fijo no determinístico local o ninguno. Definimos como el *punto fijo no determinístico* del par Φ_1, Φ_2 como la unión de todos los puntos fijos no determinísticos locales del par Φ_1, Φ_2. Si ningún punto fijo no determinístico local existe, entonces definimos el punto fijo no determinístico como el conjunto vacío.

La intuición subyacente es que la salida de una computación no determinística está definida como la disjunción de las salidas de todas las posibles computaciones. El número de etapas de un punto fijo no determinístico se toma como el máximo de todas las secuencias convergentes.

Lógicas de punto fijo no determinístico FO(PFND) se obtienen aumentando a FO con las reglas de formación de punto fijo (inflacionario o no inflacionario) no determinístico, bajo la restricción que la negación no puede ser aplicada a puntos fijos no determinísticos.

Observación:

El no determinismo descripto antes reside en el *control*, es decir, la elección de la fórmula a ser aplicada en cada etapa. Esto podría ser contrastado con el *no determinismo de los datos* de [AV91], el cual permite elegir no determinísticamente una tupla arbitraria desde una relación.

El no determinismo de PFND sería similar al de las Máquinas de Turing No Determinísticas (MTND). En MTND [BDG88], un elemento pertenece al lenguaje aceptado por ella si, cuando lo tiene como entrada, al menos una de las posibles computaciones es de aceptación.

LÓGICAS DE PUNTO FIJO ALTERNADO

Para un ejemplo que motiva otra extensión de la lógica de punto fijo, consideremos la satisfacibilidad de fórmulas booleanas cuantificadas, ya conocido por ser PSPACE-completo [GJ79]. Este problema, el cual puede ser resuelto por un algoritmo polinomial alternado [CKS81] (es decir, computado por una Máquina de Turing Alternada en tiempo polinomial), parece requerir *iteración alternada* de operadores de primer orden. Definamos qué es la iteración alternada.

Sean Φ y Ψ operadores de primer orden. Este par de operadores generan árboles de etapas, llamados *árboles convergentes de etapas* en [AVV97], que son obtenidos aplicando sucesivamente, hasta alcanzar la convergencia, uno de los operadores Φ y Ψ, o ambos. Más formalmente, un árbol convergente es un árbol binario rotulado tal que:

1. La raíz está rotulada por la relación vacía,
2. si un nodo x con rótulo S_x está en un nivel impar del árbol, entonces x tiene *un* hijo x' rotulado por $\Phi(S_x)$ o por $\Psi(S_x)$,
3. si un nodo x con rótulo S_x está en un nivel par del árbol, entonces x tiene *dos* hijos x_1 y x_2 rotulados por $\Phi(S_x)$ y por $\Psi(S_x)$, respectivamente, y
4. si x es una hoja con rótulo S_x, entonces $\Phi(S_x) = \Psi(S_x) = S_x$.

Definimos la *intersección* de las hojas de un árbol convergente de etapas como el *punto fijo alternado local* del par (Φ,Ψ).

Por ejemplo, un árbol de etapas convergente está representado en la Figura 1:

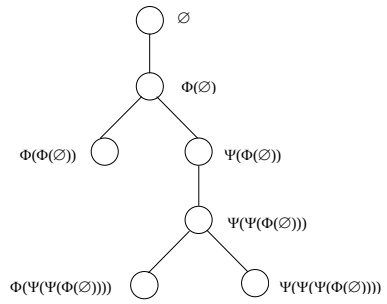

Figura 1: Árbol convergente de etapas

Su punto fijo alternado local es:

$$\Phi(\Phi(\varnothing)) \cap \Phi(\Psi(\Phi(\varnothing)))) \cap \Psi(\Psi(\Psi(\Phi(\varnothing)))).$$

La intuición subyacente es que en un árbol de computación de una máquina alternada todas las elecciones existenciales ya han sido factoreadas hacia afuera. Así, se necesita tomar la conjunción de todas las elecciones universales y luego tomar una disjunción sobre todos los árboles de computación. El número de etapas en un árbol convergente es la longitud de la rama más larga.

Notar que el par (Φ,Ψ) puede tener más de un punto fijo local o ninguno. Definimos el *punto fijo alternado* del par (Φ,Ψ) como la *unión* de todos los puntos fijos alternados locales del par (Φ,Ψ). Esta unión es esencialmente una disjunción sobre todas las elecciones existenciales que fueron factoreadas hacia afuera en cada árbol convergente. Si ningún punto fijo alternado local existe, entonces definimos el punto fijo alternado como el conjunto vacío. El número de etapas de un punto fijo alternado se toma como el número máximo de etapas sobre todos los árboles convergentes.

También aparece alguna similitud entre PFA y Máquinas de Turing Alternadas (MTA), como ocurre entre PFND y MTND. En MTA [BDG90] la Máquina de Turing posee dos tipos de estados: existenciales y universales; y alterna entre ellos en su árbol de computaciones. En PFA el árbol convergente de etapas alterna sus niveles entre aquellos en los cuáles se aplica una de las fórmulas (nivel "existencial"), y aquellos en donde se deben aplicar ambas fórmulas (nivel "universal"). Por lo tanto, en ambos formalismos se ve una alternación de etapas existenciales y universales. Sin embargo, la alternación en el caso de MTA es totalmente libre respecto de los niveles del árbol de computaciones.

Observaciones

Las lógicas de punto fijo pueden ser parametrizadas con dos dimensiones: la potencia de su construcción de iteración, *determinística* o *no determinística* o *alternada*, y la potencia de sus operadores de primer orden, *inflacionario* o *no inflacionario*. Esto da origen a seis lógicas de punto fijo. Usamos la notación $PF\alpha\beta$ para referirnos a una lógica de punto fijo con iteración de tipo α y operadores de tipo β ($\alpha \in \{D, N, A\}$ y $\beta \in \{I, N\}$). Así las lógicas FO(PFI) y FO(PFN) se denotarán como *PFDI* y *PFDN* respectivamente, y *PFAN* denota la lógica de punto fijo alternada no inflacionaria.

Sean α y β como antes. Las *fórmulas* en $PF\alpha\beta$ son obtenidas partiendo desde los átomos, por aplicaciones repetidas de operadores de primer orden $(\neg,\vee,\wedge,\rightarrow,\exists,\forall)$ y los operadores de punto fijo.

Las *sentencias* son fórmulas sin variables libres de primer y segundo orden, donde la ocurrencia libre de variables está definida de la manera usual, agregando, por ejemplo, para PFDI la cláusula: $libres([\text{PFDI } v_{x,X}\,\varphi]\,v_t\,) := libres(\,v_t\,) \cup (libres(\varphi) - \{v_x, X\})$.

La semántica se define inductivamente con respecto de las reglas de formación de las fórmulas, siendo el significado de $[\text{PFDI } v_{x,X}\,\varphi]v_t$ que $v_t \in \varphi^{\infty}$, y que $[\text{PFDN } v_{x,X}\,\varphi]v_t$, de manera similar, que $v_t \in \varphi^{\infty}$. Más precisamente: Si X es k-aria y si las variables libres en $[\text{PFDI } v_{x,X}\,\varphi]\,v_t$ están entre v_u e Y, y v_b y S son interpretaciones en B de v_u e Y, respectivamente, entonces:

$$B \models [\text{PFDI } v_{x,X}\,\varphi]v_t\ [v_b, S] \quad \text{sii } (t_1[v_b], \ldots\ldots, t_k[v_b]) \in \varphi[S]^{\infty},$$

$$B \models [\text{PFDN } v_{x,X}\,\varphi]v_t\ [v_b, S] \text{ sii } (t_1[v_b], \ldots\ldots, t_k[v_b]) \in \varphi[S]^{\infty},$$

Para $\alpha \in \{N, A\}$, las reglas de formación son como sigue. Sean $\varphi_1(X)$ y $\varphi_2(X)$ fórmulas en $PF\alpha\beta$ con k variables libres donde, X es un predicado o símbolo de relación k-ario que no pertenece al vocabulario de la estructura, o sea, una variable de relación k-aria. Entonces $[PF\alpha\beta v_{x,X}\,\varphi_1, \varphi_2]v_t$, donde v_x es una secuencia de k variables libres $x_1,\ldots, x_k$, y v_t de k variables o constantes[6]; o sea, las longitudes de v_x y v_t son la mismas y coinciden con la aridad de X. Cuando $\beta = I$, asumimos que $\varphi_1(v_x,X)$ y $\varphi_2(v_x,X)$ son fórmulas inflacionarias. Cuando $\alpha = N$ requerimos, además, que la negación

[6] En particular en nuestro lenguaje no tenemos símbolos de constantes, dado que el vocabulario relacional considerado no admite símbolos de relación 0-arios (constantes).

no sea aplicada a una fórmula que contiene un punto fijo. La semántica de las fórmulas es definida inductivamente como antes.

La restricción de no aplicar negación a puntos fijos no determinísticos parece tener relación con Máquinas de Turing No Determinísticas, porque en ellas resolver el problema de rechazo, como ya es sabido, es más complejo que resolver el de aceptación.

Las inclusiones obvias entre estas lógicas son descriptas por el siguiente diagrama:

$$\begin{array}{ccc} PFAI & \subseteq & PFAN \\ \cup| & & \cup| \\ PFNI & \subseteq & PFNN \\ \cup| & & \cup| \\ PFDI & \subseteq & PFDN \end{array}$$

Observaciones:

Se puede demostrar para estas varias lógicas de punto fijo estándar resultados tales como la clausura bajo composición y complemento (excepto para α igual a *N*). El colapso de los anidamientos de puntos fijos (en otras palabras, un operador de punto fijo por fórmula es suficiente) y la inducción simultánea (una sola variable de relación es suficiente) se mantiene en todos los casos. Estos resultados fueron demostrados para *PFDI* en [GS86] y para *PFDN* en [AV91]. La demostración es esencialmente la misma para las nuevas lógicas como se demuestra en [AVV97].

En adelante usaremos la notación arriba mencionada; o sea, $PF\alpha\beta$ para referirnos a una lógica de punto fijo con iteración de tipo α y operadores de tipo β ($\alpha \in \{D, N, A\}$ y $\beta \in \{I, N\}$).

Implementación

CUANTIFICADORES DE PUNTO FIJO DETERMINÍSTICO

El cuantificador calcula una sucesión $X_0, X_1,...$ de contenidos para X; donde X_0 es la relación vacía y donde cada relación X_i de la sucesión (para $i > 0$) se obtiene como resultado de analizar el conjunto de k-tuplas de D^k que hacen verdadera a $\varphi(v_x, X_{i-1})$. O sea, evaluando φ con el contenido X_{i-1} (la relación anterior en la sucesión) como valor de la variable de relación X en la fórmula.

El hecho de que PFDI es un cuantificador inflacionario y PFDN no, hace que la cota máxima para la cantidad de pasos a realizar, para alcanzar el punto fijo, en cada uno de ellos sea diferente; lo que justificamos a continuación:

a) Cualquier cuantificador de punto fijo inflacionario tiene la restricción que para cada X_i de la secuencia, con $i > 0$, $X_{i-1} \subseteq X_i$. Por lo tanto, en cada paso de la iteración, se pueden dar dos casos:

1. $X_{i-1} = X_i$ o
2. $X_{i-1} \subsetneq X_i$.

En el primer caso, como $X_{i-1} = \Phi(X_{i-1}) = X_i$ se ha alcanzado el punto fijo. En el segundo caso, para lograr la máxima cantidad posible de pasos en la iteración, X_i deberá diferenciarse lo mínimo posible de X_{i-1} y éste sería el caso en que sólo se agregó una nueva k-tupla a X_{i-1}. Entonces, como en D^k hay n^k k-tuplas, la cantidad máxima de pasos hasta alcanzar el punto fijo será n^k.

Así, en este caso, siempre se obtiene el punto fijo por ser inflacionario.

b) En el caso no inflacionario no hay ninguna restricción respecto del contenido de la variable de relación en etapas consecutivas. Por lo tanto, para obtener la cota máxima de pasos a efectuar para determinar si el punto fijo existe, se deberá considerar la cantidad de pasos mínima necesaria para obtener todas las relaciones posibles en D^k. Es decir, todos los posibles subconjuntos de D^k, que son 2^{n^k}. En esa cantidad de pasos existe la posibilidad de obtener todas las posibles relaciones.

En cualquier paso obviamente se obtendría como resultado una relación ya obtenida previamente, con lo que se caería en una computación infinita.

Si en 2^{n^k} pasos ninguna de las relaciones obtenidas es un punto fijo, no se alcanzará aunque siguiéramos iterando porque el punto fijo no va a existir.

CUANTIFICADORES DE PUNTO FIJO NO DETERMINÍSTICOS

Este par de operadores generará secuencias de etapas, llamadas secuencias convergentes de etapas, que son obtenidas aplicando sucesivamente Φ_1 o Φ_2.

Para encontrar el punto fijo no determinístico de un par de fórmulas φ_1, φ_2, debemos analizar todas las posibles secuencias de etapas, y ellas pueden verse como todas las ramas de un árbol binario rotulado.

Más formalmente, un árbol binario rotulado con relaciones tal que:

1. La raíz está rotulada por la relación vacía,
2. si un nodo x con rótulo S_x está en un nivel menor que la cota máxima, correspondiente al operador considerado, entonces x tiene:
   - **a)** hijo izquierdo x_I rotulado por $\Phi_1(S_x)$, sólo si $\Phi_1(S_x) \neq S_x$.
   - **b)** hijo derecho x_D rotulado por $\Phi_2(S_x)$, sólo si $\Phi_2(S_x) \neq S_x$.
   - **c)** En otro caso, el nodo x es una hoja y su rótulo S_x es un punto fijo local, ya que $\Phi_1(S_x) = \Phi_2(S_x) = S_x$.
3. si un nodo x con rótulo S_x está al nivel dado por la cota máxima, correspondiente al operador considerado, entonces x es una hoja con rótulo S_x, y se da uno de dos casos:
   - **d)** $\Phi_1(S_x) = \Phi_2(S_x) = S_x$, en cuyo caso S_x es un punto fijo local,
   - **e)** en otro caso, como S_x no es un punto fijo local (y no se cumple $\Phi_1(S_x) = \Phi_2(S_x) = S_x$), entonces S_x toma el valor $\varnothing$.

Por consiguiente, debemos analizar todas las posibles ramas de este árbol; pero, no hay necesidad de construir el árbol sino que lo recorremos implícitamente. Al analizar cada rama del árbol obtenemos una relación a la que consideramos, sin pérdida de generalidad, el punto fijo local de la secuencia que dicha rama representa. Si ningún punto fijo local existiera, que puede ocurrir sólo en el caso no inflacionario, el punto fijo será igualmente el conjunto vacío.

La Figura 2 muestra un posible árbol rotulado con los operadores Φ_1 y Φ_2:

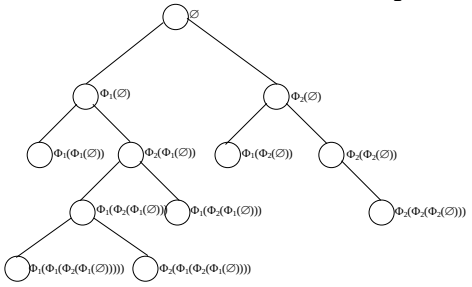

Figura 2: Árbol binario rotulado de etapas

El hecho de que *PFNI* es un cuantificador inflacionario y *PFNN* no, haría pensar que la cota máxima para la cantidad de pasos a realizar, para alcanzar el punto fijo, en cada uno de ellos debería ser diferente. Pero esto no es directo, ya que las condiciones para alcanzar un punto fijo son distintas en el caso determinístico y en el no determinístico. Por lo tanto, es necesario otro análisis para justificar cuál es la cota máxima a considerar en cada caso.

Cada rama particular de dicho árbol representa una secuencia determinística de aplicaciones de Φ_1 o Φ_2 y dicha rama alcanzará un punto fijo si el resultado de ambas aplicaciones coincide con la relación a la que se aplicaron. Por lo tanto, puede ocurrir que uno de los operadores coincida y el otro no, en cuyo caso se debería seguir, aunque por alguna de las ramas que se continúen no se logre obtener nada nuevo.

Analicemos ahora las siguientes preguntas:

1. ¿Cuándo es necesario seguir adelante con el análisis de una rama? y
2. ¿Cuál debería ser la cota máxima para cada uno de los operadores no determinísticos?
3. ¿Cualquier secuencia de etapas, que una rama del árbol representa, converge?

Para responder a la primera pregunta, consideremos estar en una situación en que hemos alcanzado un contenido X_i correspondiente a la *i*-ésima etapa de una rama. Entonces, debemos distinguir tres casos:

a) cuando Xi = Φ1(Xi) = Φ2(Xi),

b) cuando Xi ≠ Φ1(Xi) y Xi ≠ Φ2(Xi),

c) cuando Xi = Φ1(Xi) y Xi ≠ Φ2(Xi) (o cuando Xi ≠ Φ1(Xi) y Xi = Φ2(Xi)).

Las situaciones a analizar coinciden para los dos tipos de cuantificadores no determinísticos, PFNI o PFNN, por lo tanto en lo que sigue no haremos mención a ninguno de ellos en particular.

En el caso **a)** la rama se corta por haber alcanzado un punto fijo local.

En el caso **b)** se debería seguir adelante con las dos ramas (porque el nodo del árbol rotulado con X_i, en ese caso, tendrá sus dos hijos presentes).

En el caso **c)** hay que analizar más en detalle como seguir desde allí. Para ello, veamos gráficamente, en la Figura 3.a), dicha situación en un árbol:

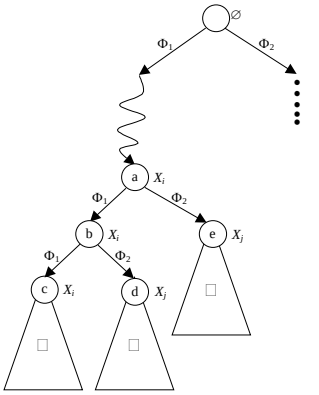

Figura: 3a) Árbol binario rotulado de etapas para el caso **c)**

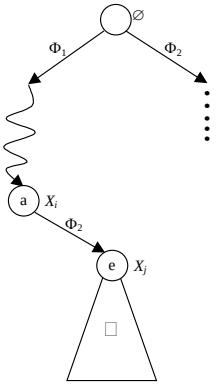

Figura 3b): Árbol binario rotulado de etapas (luego de podar el árbol de Figura 3.a)

La Figura 3.a) muestra una situación como la planteada en **c),** con todos los elementos necesarios individualizables para el análisis.

La relación que rotula los nodos “a”, “b” y “c” es la misma, pero es distinta de la que rotula los nodos “d” y “e”. Por consiguiente, todos los puntos fijos locales alcanzables desde el nodo “b”, pasando por “d”, (o sea, relaciones que rotulan las hojas del subárbol $\mathfrak{I}$) serán las mismas que se alcancen desde “e” (relaciones que rotulan las hojas del subárbol $\mathfrak{R}$). Aquellos puntos fijos locales alcanzables desde “b”, pero pasando por “c” (o sea, las relaciones que rotulan hojas del subárbol $\aleph$) serán los mismos que los obtenidos desde “d”[7]; pero, en el subárbol $\aleph$ hay claramente una rama infinita que es la obtenida aplicando siempre Φ_1.

Por lo tanto, ante una situación como la de **c)** podemos podar el árbol rotulado de etapas (ya en su definición se ha tenido en cuenta) y entonces, en el ejemplo planteado y como resultado de dicha poda, se produciría el árbol mostrado en la Figura 3.b).

Para analizar las otras preguntas debemos estudiar PFNI y PFNN por separado.

Como no van a existir, en el árbol rotulado de etapas, nodos tales que sus rótulos coincidan si ellos son padre e hijo[8] y como cada rama del árbol representa una secuencia de etapas para los operadores Φ_1 y Φ_2, podemos analizar dichas secuencias para los dos casos:

a) Inflacionario:

En este caso, la secuencia de etapas que la rama representa tendría, además de la condición de

[7] Esto puede mostrarse aplicando sucesivamente el razonamiento expuesto.

[8] Porque o el padre será una hoja, en cuyo caso es porque su rótulo coincide con el de sus dos posibles hijos, o por el análisis anterior el hijo cuyo rótulo sea distinto será el único presente.

ser incremental, la restricción que para todo $i > 0$, $X_{i-1} \neq X_i$; la cual produce finalmente que: para todo $i > 0$, $X_{i-1} \subsetneq X_i$.

Entonces, partiendo desde la relación vacía, el peor caso medido en cantidad de relaciones en la secuencia, sería aquel en que se agrega de a una k-tupla por vez (n^k pasos como máximo). En ese caso, habríamos obtenido D^k y deberíamos ver si: $D^k = \Phi_1(D^k) = \Phi_2(D^k)$; pero esto es siempre cierto por ser ambos operadores inflacionarios y por lo tanto, se alcanza el punto fijo.

En definitiva, en el caso inflacionario, cada rama del árbol rotulado de etapas contendrá un punto fijo local y éste se alcanzará en a lo sumo n^k pasos.

b) No Inflacionario:

En este caso, no hay restricción respecto de la inclusión del contenido de la variable de relación en etapas consecutivas, ellas deben ser distintas (por lo expuesto anteriormente). Por lo tanto, para obtener la cota máxima de pasos a efectuar para determinar si el punto fijo local existe, se deberá considerar la cantidad de pasos mínima necesaria para obtener todas las relaciones posibles en D^k. Es decir, todos los posibles subconjuntos de D^k, que son 2^{n^k}.

Si en 2^{n^k} pasos ninguna de las relaciones obtenidas es un punto fijo, el punto fijo no va a existir y se obtendrá el conjunto vacío. La justificación es similar al caso determinístico.

Entonces, en el caso no inflacionario, cada rama del árbol rotulado de etapas no contendrá necesariamente un punto fijo local y si éste existe se alcanzará en a lo sumo 2^{n^k} pasos.

CUANTIFICADORES DE PUNTO FIJO ALTERNADO

Sea (φ_1,φ_2) el par de fórmulas a las que se aplica el cuantificador. Cada una de las fórmulas da lugar a un *operador*; sean Φ_1 y Φ_2 dichos operadores. Como ya vimos, ellos generan árboles convergentes de etapas[9] obtenidos aplicando sucesivamente, hasta alcanzar la convergencia, o uno de los operadores Φ_1 y Φ_2 o ambos.

Nombraremos como *nodos opción* a aquellos nodos, del árbol convergente de etapas, en que se *optó* por la aplicación de uno de los operadores; o sea, aquellos nodos del árbol que se encuentran a nivel impar.

El par de operadores (Φ_1,Φ_2) genera secuencias de contenidos de X de la forma $X_0, X_1,..., X_m$, donde $X_0 = \varnothing$, $X_{i+1} = \Phi_1(X_i)$ o $X_{i+1} = \Phi_2(X_i)$ y $\Phi_1(X_m) = \Phi_2(X_m) = X_m$[10]. En ese caso X_m es un punto fijo no determinístico local del par (Φ_1,Φ_2).

A diferencia de los cuantificadores de punto fijo ya analizados, el hecho de que PFAI es un cuantificador inflacionario y PFAN no, no hace que la cota máxima para la cantidad de pasos a realizar, para alcanzar el punto fijo, en cada uno de ellos sea diferente.

Dado que los árboles convergentes de etapas pueden tener ramas infinitas, debemos forzar una cota sobre la profundidad máxima para el árbol, con el fin de asegurar que el algoritmo pare.

Como para obtener todos los puntos fijos alternados locales debemos construir todos los posibles árboles convergentes de etapas necesitamos tener en cuenta los costos, en espacio y en tiempo, del algoritmo que lo compute.

El problema que se nos presentó era qué hacer respecto de las ramas posiblemente infinitas, es decir aquellas en las que no se iban a encontrar hojas o que no se sabía a que profundidad podrían encontrarse; porque al ir construyendo un árbol si se nos presentaba cualquiera de esos casos, y no lo detectábamos, caeríamos posiblemente en un colapso de espacio o de tiempo.

Por ello, intentamos caracterizar dichos casos con el fin de poder detectarlos durante el proceso de construcción de un árbol convergente de etapas.

---

[9] En [AVV97] se lo nombra *árboles convergentes de etapas*, pero en dichos árboles las secuencias de etapas no siempre convergen.

[10] En este caso, cuando existe un m tal que $X_m = \Phi_1(X_m) = \Phi_2(X_m)$, estas secuencias realmente convergen y, por ello, reciben el nombre de *secuencias convergentes de etapas*.

Entonces, las preguntas que surgieron luego de nuestros intentos de caracterización son:

1. ¿Hasta qué profundidad examinamos una rama para asegurar que allí no vamos a encontrar hojas? O sea, ¿Hasta qué profundidad examinamos una rama para detectar que será infinita? y
2. ¿Hasta qué profundidad examinamos una rama para alcanzar una hoja?

Para responder a ambas debemos tener en cuenta nuestras limitaciones reales de espacio y tiempo. Además, por lo analizado, cualquier profundidad máxima que colocáramos como cota sería arbitraria; porque nunca sabríamos si en el nivel siguiente dicha rama convergería o no. Por ello, optamos por una de dichas cotas que, aunque también arbitraria, tiene un significado claro: 2^{n^k} .

Dicha cota expresa que es una profundidad con la cual se podría haber pasado por todas las posibles relaciones de aridad *k* sobre nuestro dominio de *n* elementos. De acuerdo a la posición adoptada sospechamos como posiblemente infinita cualquier rama cuya profundidad sea mayor que 2^{n^k} y entonces, al momento de obtener el punto fijo alternado local de dicho árbol, ignoramos dicha rama; pero, indicamos que por ello el resultado obtenido podría no ser el correcto.

El resultado no sería el correcto porque si dicha rama hubiera contenido hojas, a una profundidad mayor, las relaciones que las rotulan no se intersectaron con las demás hojas del árbol para obtener el punto fijo alternado local correspondiente. Pero si realmente hubiera sido una rama infinita, o sea que en ella no se hubiera alcanzado ninguna hoja, el resultado sería el correcto.

Se puede deducir que en un árbol de altura *h* habrá una cantidad máxima *N* de nodos opción equivalente a la suma, sobre todos los niveles impares, de la cantidad máxima de nodos opción en el dicho nivel. O sea, *N* se obtendrá como:

$$N = \sum_{i=0}^{\lfloor (h/2) \rfloor} 2^i = 2^{\lfloor (h/2) \rfloor + 1} - 1 = 2^{\lceil (h/2) \rceil} - 1$$

y esta cantidad da la idea del espacio necesario para computar los árboles de etapas que construimos durante la computación del punto fijo. Al permitir que cualquier árbol llegue a lo sumo a un nivel de 2^{n^k}, la cantidad de nodos opción que podrían aparecer como máximo en un árbol completo de esa altura sería: $N = 2^{2^{(n^k)} - 1} - 1 = 4^{(n^k) - 1} - 1$

Luego de construir el primer árbol de etapas, las relaciones contenidas en las hojas del árbol se intersectan para obtener su punto fijo alternado local. A continuación, se actualiza el punto fijo alternado global y entonces, se busca armar un nuevo árbol con el menor esfuerzo posible; es decir, a partir del árbol viejo descartar aquello que en el nuevo no debería aparecer y dejar aquellas partes que si pueden quedar. El armado del nuevo árbol, se lleva a cabo identificando cada una de las raíces de los subárboles del anterior que deben cambiar; o sea, teniendo en cuenta que sean aquellas que deban realizar la menor cantidad de cambios al viejo árbol para obtener el nuevo.

Conclusiones

Dado que el poder expresivo de una Lógica de Primer Orden está restringido a una subclase de la clase de Queries Computables (CQ), hemos utilizado una lógica extendida con seis tipos distintos de cuantificadores de Punto Fijo para la expresión y evaluación de queries que permiten la evaluación de consultas a Bases de Datos Relacionales que caen fuera de lo expresable en FO.

Esto es relevante de destacar porque considerando la clase CQ, en general los lenguajes de consultas que se utilizan en la práctica no toman en cuenta el concepto de computabilidad y son incompletos respecto del mismo, o calculan queries no computables en su defecto. Obsérvese que estos problemas de expresividad y computabilidad no son sólo de interés teórico, sino que los mismos pueden causar serios errores en aplicaciones de cierta complejidad.

En virtud de lo expuesto, la formulación de query es una abstracción precisa y rigurosa del tipo de consulta que un programa de computadora debería computar. Nosotros hemos mostrado una parte de estos resultados, exponiendo simplemente como con FO no es suficiente para la expresión de queries de cierta categoría.

Así podemos observar que sobre estructuras finitas ordenadas, con las extensiones de tales cuantificadores, capturaríamos la clase EXPTIME, mientras que con FO quedamos muy lejos del poder expresivo conseguido con las extensiones realizadas.

Cabe destacar también que hemos realizado un arduo análisis en determinar algoritmos, cotas y comportamientos de estos cuantificadores, puesto que las consultas bibliográficas a los autores de referencia, no nos aportaron soluciones.

De lo aquí presentado, varios tópicos quedan abiertos para la continuación de este trabajo. Por ejemplo: optimización de las consultas antes de ser evaluadas con el fin de realizar una evaluación eficiente de las mismas, inclusión de otros tipos de cuantificadores nuevos a nuestro lenguaje, inclusión de subconjuntos de lógicas de orden superior, determinación de cotas más adecuadas haciendo uso de algo más de información respecto de las secuencias que se generan, o determinación de si, por ejemplo, hay un cuantificador de punto fijo alternado que permita acercarse más al comportamiento de una Máquina de Turing Alternada.

Bibliografía

[AV95] Abiteboul, S.; Vianu, V.; "*Computing with First Order Logic*"; Journal of Computer and System Sciences, 50:309-335, 1995.

[AVV97] Abiteboul, S.; Vardi, M. y Vianu, V.; "*Fixpoint Logics, Relational Machines, and Computational Complexity*", Journal of ACM, 44(1), 30-56.Enero, 1997.

[BDG88] Balcazar, J.; Díaz, J. y Gabarró, J. "*Structural Complexity I* ",Springer Verlag. 1988.

[BDG90] Balcazar, J.; Díaz, J. y Gabarró, J. "*Structural Complexity II* ",Springer Verlag. 1990.

[CH80] Chandra, A.K.; Harel, D.; "*Computable Queries for Relational Data Bases*". Journal of Computer and System Sciences 21, 156-178. 1980.

[CH82] Chandra, A.K.; Harel, D.; "*Structure and Complexity for Relational Queries*". Journal of Computer and System Sciences 25,99-128. 1982.

[CKS81] Chandra, A.K.; Kozen, D. y Stockmeyer, L. "*Alternation*"; Journal of the ACM, "8:114-133.1981

[EF95] Ebbinghaus, H; Flum, J.; "*Finite Model Theory*", Springer-Verlag, 1995.

[EFT84] Ebbinghaus, H; Flum, J.;Thomas, W.; "*Mathematical Logic*", Springer-Verlag, 1984.

[GJ79] Garey, M.R.; Johnson, D.S.; "*Computers and Intractability- A Guide to the Theory of NP-Completeness*", W.H. Freeman and Co. , 1979.

[GMR99] Gagliardi, Edilma; Maldocena, Paulino y Reyes, Nora "*Utilización de Extensiones de Lógica de Primer Orden para la Computación de Queries en problemas de Redes"*; IV CACIC, Tandil, 1999.

[RGMT00] Reyes, N; Grosso, A.; Maldocena, P. y Turull, J.; "*Un intérprete de consultas a bases de datos expresadas mediante Lógica de Primer Orden con Clausura Transitiva*" Enviado a evaluar.

[GS86] Gurevich, S.y Shelah, S.; "*Fixed Point Extensions of First Order Logic"* Annals of Pure and Applied Logic; 32:265-280, 1986.

[Imm86] Immerman, Neil; "*Relational queries computable in polynomial time*"; Information and Control, 68, 1986, 86-104.

[Ull88] Ullman, Jeffrey D.; "*Principles of Database and Knowledge Base Systems*". Computers Science Press, 1988.

[Var82] Vardi, M. Y.; "*The Complexity of Relational Queries Languages*"Proc. 14th ACM Symp. On Theory of Computing, 137-146, 1982.