Traducción con Optimización de expresiones del Cálculo Relacional al Álgebra Relacional

Gagliardi, Edilma Olinda
Turull Torres, José María *
Yrusta, Mariano Martín

{ oli, turull, myrusta } @unsl.edu.ar

Universidad Nacional de San Luis (UNSL)
Facultad de Ciencias Físico, Matemáticas y Naturales
Departamento de Informática
Avda. Ejército de los Andes 950
(5700) San Luis
Argentina

ABSTRACT

Considerando los lenguajes formales Cálculo Relacional y Álgebra Relacional para expresar consultas a bases de datos relacionales, y tomando en cuenta la equivalencia entre ambos, la motivación de este trabajo consistió en la generación de una herramienta que contuviera una estrategia propia para la traducción del Cálculo Relacional al Álgebra Relacional con optimización incluida, para toda consulta perteneciente a la familia completa de consultas expresables por ambos lenguajes. Para ello, se tomó como marco de referencia la complejidad de las operaciones algebraicas y sus influencias en las relaciones temporarias generadas en la evaluación de la consulta.

Palabras Claves: Bases de Datos Relacionales, Cálculo Relacional orientado a Dominios, Álgebra Relacional, Queries, Traducción, Optimización de queries.

* Universidad Nacional de San Luis; Universidad Tecnológica Nacional (FRBA); Subsidio de Universidad CAECE

INTRODUCCIÓN

Tomando los lenguajes de consultas definidos por Codd [COD 70], Álgebra Relacional (AR) y Cálculo Relacional orientado a Dominios (DRC), como formalismos para expresar consultas (o queries) a bases de datos relacionales, y considerando la equivalencia demostrada entre los mismos [ULLMAN 79], la motivación de este trabajo consistió en implementar un traductor del DRC al AR, y que, además, considerara durante el proceso de traducción posibles formas de optimización de las expresiones. Para ello, se tomó como marco de referencia la complejidad de las operaciones algebraicas y su influencia en las relaciones temporarias que se generan durante la evaluación del query.

A partir del teorema de traducción del DRC al AR y de varias estrategias independientes de optimización de expresiones vistas en ambos lenguajes [ABITEBOUL 95], [EBBINGHAUS 84], [MAIER 83], [ULLMAN 79], [ULLMAN 88], nuestro proyecto residió en el desarrollo de un traductor del DRC al AR con un optimizador de expresiones incluido, de manera tal que la expresión obtenida después de la traducción, es óptima respecto de la manipulación algebraica, y la misma es lograda desde el inicio hasta el fin del proceso de transformación [YRUSTA 99].

También destacamos que en especial esta herramienta está dirigida a toda la familia de queries expresables con AR, a diferencia de otras herramientas que tratan una subclase de dicha familia, como por ejemplo Queries SPJ, Queries SPC, Queries SPCU, etc. [ABITEBOUL 95], [MAIER 83], [ULLMAN 79], [ULLMAN 88]. Por ello veremos que no tenemos restricciones en cuanto al uso de concetivos, cuantificadores y/o operadores algebraicos.

Desde el punto de vista de aplicación, los lenguajes DRC y AR han contribuido al desarrollo de lenguajes de aplicación para consultar bases de datos relacionales como marcos de referencia teórica, como lo es por ejemplo SQL. Un lenguaje de aplicación que recibe un query expresado en cálculo, busca una estrategia de ejecución correspondiente a tal query que minimice las funciones de costo que incluyen procesamiento, accesos y comunicaciones. Una estrategia de ejecución debe ser especificada en términos de procedimientos u operaciones algebraicas para llevarla a cabo. Por lo tanto, la complejidad algebraica de tales operaciones que trabajan con relaciones, afectan la performance de la evaluación del query y esto resulta de gran importancia en el diseño de un evaluador de consultas a bases de datos relacionales [OSZU 91].

La implementación de nuestra herramienta tiene dos componentes básicas, donde, fundamentalmente, la segunda queda absorbida por la primera: la inicial es un traductor de un query expresado en DRC al AR; y la segunda efectúa la optimización del query desde su expresión original en DRC hasta su expresión final en AR.

Las experiencias mostraron que al hacer traducción y optimización por separado, se obtienen resultados totalmente diferentes, y las expresiones obtenidas no están totalmente optimizadas y/o su complejidad no es la esperada. Mientras que con nuestra herramienta se logran mejores optimizaciones.

Desde el punto de vista pedagógico es una buena herramienta para que alumnos observeny experimenten en la expresión y optimización de queries.

A raíz de los resultados de este trabajo, surgieron extensiones para el mismo, viendo la posibilidad de computaciones paralelas en las expresiones y la extensión de la herramienta para Bases de Datos Distribuidas.

Este artículo está organizado de la siguiente forma: Los puntos 1 y 2 están referidos a Bases de Datos Relacionales y Lenguajes relacionales (DRC y AR); el punto 3 muestra reglas puras de traducción del DRC al AR [ULLMAN 79]; el punto 4 muestra reglas de optimización para el DRC y el AR respectivamente [ABITEBOUL 95], [MAIER 83], [ULLMAN 79], [ULLMAN 88]; en el punto 5 presentamos nuestro trabajo, mostrando las reglas de traducción y optimización simultáneas [YRUSTA 99]; el punto 6 está dedicado a mostrar una aplicaicón con ejemplos de consultas, haciendo primero traducción y optimización por separado, y luego, utilizando nuestra herramienta traducción y optimización conjunta. Finalmente, presentamos nuestras conclusiones y proyecciones del trabajo.

1. BASES DE DATOS Y QUERIES

Sea A ={ (,),,, $\wedge$,$\vee$, $\neg$, $\forall$,$\exists$,x_1,x_2,…x_n,… } un alfabeto infinito numerable. Sea σ un vocabulario finito, σ = < R_1,R_2,…,R_k, c_1,c_2,…c_m>, el cual tiene definido los símbolos de relación juntamente con las aridades y símbolos de constantes. 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 habitual semántica formal de Tarski, [CHANG 92], [EBBINGHAUS 84] [EBBINGHAUS 95], [TURULL,96];

Se dice que B es una σ-estructura, denotada B=<D^B, R_1^B, R_2^B,…, R_k^B,c_1^B,c_2^B,…c_m^B>, donde D^B es un conjunto finito, D^B ={d_1,...d_n}, denominado Dominio de B; y las relaciones R_1^B,R_2^B,…,R_k^B definidas en D^B, de la aridad adecuada y únicas para cada símbolo de relación del vocabulario σ, queriendo significar con ello que si B es una interpretación para el lenguaje L_σ de dominio D^B , la interpretación en B de R_i se denotará por R_i^B. Se asume la relación de Igualdad como parte del vocabulario. Para el caso de las constantes c_1^B,c_2^B,…c_m^B, definidas en D^B y únicas para cada símbolo de constante del vocabulario σ, donde la interpretación en B de c_j se denotará c_j^B .

Observando el Modelo Relacional en el marco de la Teoría de Modelos Finitos [EBBINGHAUS 95], [TURULL,96] el esquema de la Base de Datos Relacional está dado por un vocabulario relacional finito σ, donde cada símbolo de relación en σ es de una cierta aridad; 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.

Las Bases de Datos Relacionales son consideradas como Clases de Estructuras Relacionales Finitas, y a las instancias de Bases de Datos Relacionales como Estructuras Relacionales Finitas.

Sea la fórmula φ, $\varphi \in L_\sigma$ con variables libres {x_1,x_2,...,x_r}, y B es una σ-estructura, con d_1,…,d_n $\in$ D^B, se denotará con la siguiente expresión B $\models$ φ (x_1, …,x_r)[d_{s_1},…,d_{s_r}] al valor de verdad Verdadero de la fórmula φ interpretada en la estructura B, con el elemento d_{s_j} asignado a la variable libre x_j, para $1 \le s_j \le n$, $1 \le j \le r$.

Sea $\varepsilon_{\sigma,fin}$ el conjunto de las estructuras finitas de un cierto vocabulario σ. Y sea $\varepsilon_{<R>}$ el conjunto de las relaciones definidas en el dominio de la estructura finita correspondiente, para alguna aridad r. Entonces, un query r-ario, para un entero r>0 , es una función de la forma $f: \varepsilon_{\sigma,fin} \to \varepsilon_{<R>}$, para algún vocabulario σ, donde la aridad de R es r, y donde para toda estructura B $\in$ $\varepsilon_{\sigma,fin}$, f(B) está definida en D^B. De modo análogo se denomina query 0-ario, o query booleano, a toda función de la forma $f: \varepsilon_{\sigma,fin} \to \{0,1\}$.

Se dice que f es un Query Computable [CHANDRA 80] si es una función recursiva parcial, en cualquier representación de las σ-estructuras en estructuras totalmente ordenadas, y si preserva isomorfismos, es decir, para todo par de σ-estructuras I_1 e I_2 , y para todo isomorfismo h:$I_1 \to I_2$ debe ser f(I_2) = h(f(I_1)).

Nótese que de acuerdo a la semántica del lenguaje considerado, si φ es tal como se definió arriba, define o expresa un query r-ario f_φ sobre la estructura B. Es decir f_φ (B), denotado por φ^B, es una relación finita definida en la estructura B por la fórmula φ, cuya aridad estará determinada por la cantidad de variables libres de φ. En símbolos:

$$\varphi^B = \{(d_{s_1},\ldots,d_{s_r}) : d_{s_1},\ldots,d_{s_r} \in D^B \wedge B \models \varphi(x_1,\ldots,x_r)[d_{s_1},\ldots,d_{s_r}],\ 1 \le s_j \le n,\ 1 \le j \le r\}$$

Con φ nos referiremos, indistintamente, a la fórmula y a la relación que la misma expresa.

2. LENGUAJES DE CONSULTA EN BASES DE DATOS RELACIONAL

El DRC no es más que la formulación de la Lógica de Primer Orden, [COD 70] por ello este lenguaje está basado en el cálculo de predicados, comúnmente llamado lenguaje no procedural. La

expresión de consultas se hace describiendo cuál es el conjunto de tuplas deseadas por medio de la especificación de algún predicado, al que las tuplas resultantes deben satisfacer.

Mientras que el AR es un lenguaje algebraico, o comúnmente llamado lenguaje procedural, dado que las consultas son especificadas por una secuencia de operaciones que construyen la respuesta. Las expresiones del AR están compuestas por operandos que son relaciones y los operadores se clasifican en Primitivos y No Primitivos.

Recordemos cuáles son los operadores y sus respectivas complejidades computacionales, considerando a n como la cardinalidad de una relación:

- Operadores Primitivos: Unión (U) y Diferencia (-) con complejidad O(nlogn); Producto Cartesiano (x) con complejidad $O(n^2)$; Proyección (π) y Selección (σ_F), con complejidad O(n).
- Operadores No Primitivos: Intersección ($\cap$), División ($\div$) y Join (Ξ_F) con complejidad O(nlogn).

El cuidado que debe tomarse en la expresión de queries es que no sólo respondan a la petición, sino que además no involucren el cálculo de grandes cantidades de datos inútiles. Por ello se han desarrollado técnicas para optimizar queries ya sean del Cálculo o del Álgebra, donde los optimizadores producen expresiones menos obvias respecto de la semántica pero más eficientes al momento de su evaluación [ABITEBOUL 95], [MAIER 83], [OSZU 91], [ULLMAN 79], [ULLMAN 88].

La equivalencia de ambos lenguajes es utilizada por los lenguajes de aplicación que necesitan hallar una estrategia de ejecución que minimice los costos de procesamiento, accesos y comunicaciones. Puesto que una estrategia de ejecución debe ser especificada en términos de operaciones algebraicas, es necesario que la complejidad de las operaciones correspondientes a la evaluación del query se minimice. Por ello las complejidades de tales operadores inducen algunos principios útiles tal que para la evaluación de la misma ayudan a elegir una estrategia final adecuada.

3. TRADUCCIÓN

El teorema presentado en [ULLMAN 79], expresa: "Para cada expresión segura del cálculo relacional orientado a dominio, existe una expresión equivalente en álgebra relacional". Este teorema demuestra inductivamente sobre el número de operadores de una expresión segura en DRC, que existe una expresión equivalente en AR.

En la siguiente tabla se muestra en la primera columna una expresión en DRC, y en la segunda, su expresión equivalente en AR. Estas equivalencias presentadas son las que se desprenden del teorema de equivalencia entre ambos lenguajes.

Expresión E' en DRC ≡ Expresión E'' en AR	
Expresión E'	***Expresión E''***
Fórmula Atómica	D: Dominio de la estructura $\theta \in \{ =, \leq, \geq, \neq, <, > \}$
$\mathbf{R(x_1, x_2, ..., x_l)}$	$\boldsymbol{\pi_{j1, j2, ..., jk} (\sigma_F (R))}$ Donde F es una fórmula que tiene átomos u=v siempre que x_u y x_v sean la misma variable y u<v; todos los atómos están conectados por el operador $\wedge$. La lista $j_1, j_2, ..., j_k$ es cualquier lista tal que $x_{j1}=x_1,..., x_{jk}=x_k$.
$\mathbf{x_1 \theta x_2}$	$\boldsymbol{\sigma_{1\theta 2} (D \times D)}$
$\mathbf{x_1 \theta x_1}$	$\boldsymbol{\sigma_{1\theta 1} (D)}$
$\mathbf{x_1 \theta a}$	$\boldsymbol{\sigma_{1\theta a} (D)}$
Fórmula Bien Formada	
$\mathbf{\omega(y_1,...,y_m) \equiv \omega_1(u_1,...,u_n) \vee \omega_2(v_1,...,v_p)}$	$\boldsymbol{\pi_{i1, i2, ..., im} (E_1 \times D^{m-n}) \cup \pi_{j1, j2, ..., jm} (E_2 \times D^{m-p})}$

donde cada u_i es un y_j distinto y cada v_i es un y_j distinto, aunque algunos u's y v's pueden ser el mismo y_j.	Donde E_1 y E_2 son expresiones algebraicas para ω_1 y ω_2 respectivamente.
$\omega(y_1,...,y_m) \equiv \neg\omega_1(y_1,...,y_m)$	$\mathbf{D^m - E_1}$ Donde E_1 es una expresión algebraica para ω_1
$\omega(y_1,...,y_m) \equiv (\exists\ y_i)\ (\omega_1(y_1,...,y_n))$ con $1<=i<=n$	$\pi_{1, 2, ..., i-1, i+1, ..., m}\ (E_1)$ Donde E_1 es una expresión algebraica para ω_1

Tabla I. Expresiones en DRC equivalentes a expresiones en AR.

Obsérvese que las equivalencias presentadas anteriormente sólo se limitan a traducir expresiones desde DRC al AR, sin tomar en cuenta el costo de la consulta obtenida, ni tampoco hace optimización sobre la expresión final. Por ello, recordamos que nuestro objetivo es traducción con optimización simultánea.

4. OPTIMIZACIÓN

El DRC y el AR aportan un marco teórico adecuado debido a su amplitud y generalidad para el estudio de queries a bases de datos relacionales, aunque sólo expresen una subclase de la clase CQ.

En términos de lenguajes de aplicación, un lenguaje para ingresar el query, generalmente está basado en cálculo y es de alto nivel para facilidad del usuario. Un query expresado en términos del cálculo es más sencillo de comprender puesto que se restringe a conocer las especificaciones que la respuesta debe satisfacer. Pero, el mismo debe ser necesariamente descompuesto en una serie de operaciones relacionales, tal que la misma sea equivalente a la expresión original, por lo que el evaluador del query implementa la estrategia de ejecución correspondiente a tales operaciones.

En consecuencia, la transformación debe lograrse con correctitud y eficiencia; es decir, respetar la semántica del query y obtener una estrategia de procesamiento del query que minimice el uso de los recursos.

Elegir para un query dado la expresión equivalente mejor optimizada es un problema intratable con un gran número de relaciones, entonces generalmente la idea se reduce a elegir la solución más cercana a la óptima.

Las optimizaciones pueden ser hechas *estáticamente*, es decir antes de ejecutar el query, o *dinámicamente*, es decir durante la ejecución del mismo [OSZU 91].

Para el primer caso, el concepto de optimización puede ser abordado desde el punto de vista de las transformaciones y manipulaciones algebraicas que son aplicadas a las expresiones algebraicas, sin considerar las instancias particulares de las relaciones. Por ello, las cardinalidades de las relaciones permanentes o de las temporarias son estimadas basándose en las complejidades de las manipulaciones algebraicas [OSZU 91].

Mientras que cuando la optimización se realiza dinámicamente, entonces en cualquier momento de la ejecución se mejora la estrategia haciendo uso del conocimiento acumulado acerca de los resultados de las operaciones realizadas previamente. Esto implica realizar en cada ejecución la optimización buscando la mejor alternativa en cada momento, por lo cual obviamente el costo es más elevado que en el caso estático, y no necesariamente garantizan la mejor optimización. Mientras que el estático, al hacer uso de las complejidades algebraicas y al hacer estimaciones acerca de cómo serán los resultados intermedios, minimizan la probabilidad de una mala elección en la estrategia de optimización.

Hay algunas estrategias *híbridas*, es decir que toman las ventajas de la estática, permitiendo que en tiempo de procesamiento del query se analice la mejor estrategia a seguir según casos de cardinalidades de relaciones intermedias no previstas [ABITEBOUL 95], [MAIER 83], [OSZU 91], [ULLMAN 79], [ULLMAN 88].

Nuestra herramienta de optimización está basada en optimización estática. Cabe destacar que no sólo se optimiza sobre la expresión final, sino que la optimización comienza desde el mismo

instante en que se toma la expresión en DRC y se efectúa el proceso de traducción [YRUSTA 99].

4.1 ESTRATEGIAS PARA LA OPTIMIZACIÓN DE EXPRESIONES ALGEBRAICAS

Mencionamos a continuación algunas estrategias habituales en la optimización estática de expresiones algebraicas, las cuales, serán de utilidad en la estrategia general de traducción con optimización presentada en la siguiente sección [ABITEBOUL 95], [MAIER 83], [OSZU 91], [ULLMAN 79], [ULLMAN 88].

Estas son:

1. Realizar las selecciones tan rápido como sea posible, dado que reduce las cardinalidades de las relaciones intermedias obtenidas en las evaluaciones de las subfórmulas. Idem para las proyecciones.
2. Si existe un Producto Cartesiano como argumento de una selección que involucra comparaciones entre atributos pertenecientes a ambas relaciones, entonces se traduce en un Join con fórmula. En el caso en que la fórmula no involucra atributos comunes, la selección puede ser aplicada a uno u otro de los operandos según corresponda, descomponiendo la fórmula si es necesario.
3. Combinar secuencias de operaciones unarias, unificar selecciones, eliminar proyecciones triviales, introducir proyecciones, otros.

Nótese que son reglas que apuntan a las operaciones algebraicas, sin mirar el aspecto de la expresión original. Esto es mirar si hay expresiones triviales o redundantes, si pueden cambiarse algunas subexpresiones y otros.

4.2 EXPRESIONES EQUIVALENTES

A continuación mostramos formatos de *expresiones algebraicas* equivalentes entre sí (Ver Tabla II), y formatos de *expresiones lógicas* equivalentes entre sí (Ver Tabla III), que son tomadas en cuenta en nuestra estrategia de traducción con optimización simultánea.

Notación: con Ξ_F se denota Join con Fórmula.

Expresiones Algebraicas Equivalentes E' ≡ E''		
Regla	**Expresión E'**	**Expresión E''**
1	$E_1 \times E_2$	$E_2 \times E_1$
	$E_1 \Xi_F E_2$	$E_2 \Xi_F E_1$
2	$(E_1 \Xi_F E_2) \Xi_{F'} E_3$	$E_1 \Xi_F (E_2 \Xi_{F'} E_3)$
	$(E_1 \times E_2) \times E_3$	$E_1 \times (E_2 \times E_3)$
3	$\pi_{A1,...,An} (\pi_{B1,...,Bm} (E))$	$\pi_{A1,...,An} (E)$ Nótese que en los atributos $A_1,...,A_n$ deben estar los B_i`s para que la cascada sea legal.
4	$\sigma_{F1} (\sigma_{F2} (E))$	$\sigma_{F1 \wedge F2} (E)$ Obsérvese que si E es un átomo, entonces no existe una expresión equivalente más óptima que ésta.
	$\sigma_{F1} (\sigma_{F2} (E))$	$\sigma_{F2} (\sigma_{F1} (E))$
5	$\pi_{A1,...,An} (\sigma_F (E))$	$\sigma_F (\pi_{A1,...,An} (E))$ La condición F debe involucrar solamente atributos $A_1,...,A_n$.
	$\pi_{A1,...,An} (\sigma_F (E))$	$\pi_{A1,...,An} (\sigma_F (\pi_{A1,...,An,B1,...,Bm} (E)))$ Si la condición F también involucra atributos $B_1,...,B_m$ que no se encuentran en $A_1,...,A_n$
6	$\sigma_F (E_1 \times E_2)$	$\sigma_F (E_1) \times E_2$ Si F involucra solamente atributos de E_1.

	$\sigma_F (E_1 \times E_2)$	$\sigma_{F1} (E_1) \times \sigma_{F2} (E_2)$ Si F es de la forma $F_1 \wedge F_2$, donde F_1 involucra solo atributos de E_1, y F_2 involucra sólo atributos de E_2
	$\sigma_F (E_1 \times E_2)$	$\sigma_{F2} (\sigma_{F1} (E_1) \times E_2))$ Si F_1 involucra sólo atributos de E_1, pero F_2 involucra atributos de E_1 y E_2
7	$\sigma_F (E_1 \cup E_2)$	$\sigma_F (E_1) \cup \sigma_F (E_2)$
8	$\sigma_F (E_1 - E_2)$	$\sigma_F (E_1) - \sigma_F (E_2)$
9	$\pi_{A1,...,An} (E_1 \times E_2)$	$\pi_{B1,...,Bm} (E_1) \times \pi_{C1,...,Ck} (E_2)$ Los atributos $B_1,...,B_m$ y $C_1...C_k$ se encuentran en el conjunto $A_1...A_n$
10	$\pi_{A1,...,An} (E_1 \cup E_2)$	$\pi_{A1,...,An} (E_1) \cup \pi_{A1,...,An} (E_2)$

Tabla II. Expresiones Algebraicas Equivalentes.

Expresiones Lógicas Equivalentes E' ≡ E''		
Regla	**Expresión E'**	**Expresión E''**
1	$p_1 \wedge p_2$	$p_2 \wedge p_1$
2	$p_1 \vee p_2$	$P_2 \vee p_1$
3	$p_1 \wedge (p_2 \wedge p_3)$	$(p_1 \wedge p_2) \wedge p_3$
4	$p_1 \vee (p_2 \vee p_3)$	$p_1 \vee (p_2 \vee p_3)$
5	$p_1 \wedge (p_2 \vee p_3)$	$(p_1 \wedge p_2) \vee (p_1 \wedge p_3)$
6	$p_1 \vee (p_2 \wedge p_3)$	$(p_1 \vee p_2) \wedge (p_1 \vee p_3)$
7	$\neg (p_1 \wedge p_2)$	$\neg p_1 \vee \neg p_2$
8	$\neg (p_1 \vee p_2)$	$\neg p_1 \wedge \neg p_2$
9	$\neg(\neg p_1)$	p_1
10	$p_1 \vee p_1$	p_1
11	$p_1 \wedge p_1$	p_1
12	$p_1 \vee \neg p_1$	Verdadero
13	$p_1 \wedge \neg p_1$	Falso
14	$\exists x \varphi$	$\neg \forall x \neg \varphi$
15	$\forall x \varphi$	$\neg \exists x \neg \varphi$

Tabla III. Expresiones Lógicas Equivalentes

En las Tablas II y III se muestran expresiones equivalentes, las cuales sirven como reglas de optimización para expresiones de un mismo lenguaje. Recordemos que nuestro objetivo es mostrar un traductor desde AR a DRC con optimización simultánea [ABITEBOUL 95], [EBBINGHAUS 84], [MAIER 83], [OSZU 91], [ULLMAN 79], [ULLMAN 88].

5. TRADUCCIÓN CON OPTIMIZACIÓN DE EXPRESIONES

En esta sección presentamos nuestra estrategia para traducir desde el DRC al AR con optimización incluida [YRUSTA 99]. Este método surge del análisis efectuado acerca de la equivalencia de los lenguajes relacionales, las estrategias de optimización habituales en tales lenguajes, buscando una combinación de ellas, y haciendo provecho de las *manipulaciones algebraicas*, las *búsquedas de subexpresiones equivalentes* y del uso de las *estrategias de optimización* mencionadas.

Como mencionamos anteriormente, nuestra estrategia apunta a la optimización de toda la familia de queries expresables con AR. Es decir, evitamos tomar una expresión en DRC que utilice únicamente un conjunto adecuado de conectivos, como por ejemplo $\{\vee,\neg\}$, y el cuantificador

existencial, puesto que nos interesa aprovechar las reglas presentadas en la sección anterior, como así también evitamos trabajar con queries algebraicos que pertenezcan a una subclase de queries del AR.

5.1. REGLAS DE TRADUCCIÓN CON OPTIMIZACIÓN

En la siguiente tabla mostramos las reglas utilizadas en nuestra estrategia.

Expresión E' en DRC ≡ Expresión E'' en AR		
Regla	**Expresión E'**	**Expresión E''**
	Fórmula Atómica	D: dominio de la estructura ∅: relación vacía $\theta \in \{ =, \leq, \geq, \neq, <, > \}$
1	$\mathbf{R(t_1,...,t_n)}$	$\boldsymbol{\pi_{j1, j2, ..., jk}\ (\sigma_F\ (R))}$ Donde F es una fórmula del cálculo de predicados que contiene átomos u=v siempre que los términos t_u y t_v sean los mismos y éstos sean variables y u<v; si el término t_k es una constante se traduce en un átomo k=c; y todos los átomos están conectados por el conectivo ∧. La lista j_1, j_2, ..., j_k es cualquier lista tal que $x_{j1}=x_1$,..., $x_{jk}=x_k$,
2	$\mathbf{x_i\ \theta\ x_j}$**, con i≠j**	• Se traduce como átomo de una fórmula correspondiente a una selección cuyo operando es una expresión algebraica para todas las fórmulas bien formadas $\omega(t_1,...,t_n)$ que tengan a x_i y a x_j como términos. Es decir, si $t_k = x_i$ y $t_p = x_j$ y E_1 es una expresión algebraica para ω, se traduce: $\boldsymbol{\sigma_{(k\ \theta\ p)}\ (E_1)}$ En caso de que haya más de una subfórmula atómica $x_i\ \theta\ x_j$ que restrinja algún término de ω, la fórmula de la selección tendrá más de un átomo, cada uno separado por el conectivo lógico que le corresponda; esto es, el conectivo con el que se encontraba en la subfórmula. • Si se tienen dos subfórmulas atómicas de la forma $R_k(t_1,...,t_n)$ y $R_p(t'_1,...,t'_n)$ conectados por el conectivo lógico ∧ y se tiene el átomo $x_i\ \theta\ x_l$ tal que x_i sea un término de R_k y x_l sea un término de R_p también conectado por el conectivo lógico ∧ y sean E_1 y E_2 expresiones algebraicas para R_k y R_p respectivamente se traduce: $\mathbf{E_1\ \Xi_F\ E_2}$ **,** con $F \equiv \mathbf{x_i\ \theta\ x_j}$ En caso de que haya más de una subfórmula atómica de la forma $x_i\ \theta\ x_j$ tal que x_i sea un término de R_k y x_l sea un término de R_p, la fórmula F tendrá más de un átomo, cada uno separado por el conectivo lógico que le corresponda; esto es, el conectivo con el que se encontraba en la subfórmula. • Si no se puede aplicar algunos de los puntos anteriores, se traduce $\boldsymbol{\sigma_{(1\ \theta 2)}\ (D^2)}$

		Esta regla se aplica para todas las subfórmulas que son restringidas por $x_i \ \theta \ x_j$.
3	$\mathbf{x_i \ \theta \ x_i}$	• Si el operador de comparación θ es el = , esta subfórmula no se traduce, tampoco se traduce si θ no es el operador de comparación = y existe al menos una subfórmula atómica de la forma $x_i \ \theta \ a$ con el conectivo ∨ a la izquierda. En caso contrario, se traduce como una relación vacía en cada una de las fórmulas atómicas $R(t_1,...,t_n)$ que tengan a x_i como uno de sus términos. • Si no se puede aplicar algunos de los puntos anteriores, se traduce $\mathbf{\sigma_{(1 \ \theta 1)} (D)}$ Esta regla se aplica para todas las subfórmulas que son restringidas por $x_i \ \theta \ x_i$.
4	$\mathbf{x_i \ \theta \ a}$	• Se traduce como fórmula de una selección cuyo operando es cada una de las fórmulas atómicas $R(t_1,...,t_n)$ que tengan a x_I como uno de sus términos, es decir, si $t_j=x_I$ se traduce: $\mathbf{\sigma_{j \ \theta a} (R)}$ En caso de que haya más de una subfórmula atómica que restrinja algún término de R, la fórmula de la selección tendrá más de un átomo, cada uno separado por el conectivo lógico que le corresponda, esto es el conectivo con el que se encontraba en la subfórmula. • Si no se puede aplicar algunos de los puntos anteriores, se traduce $\mathbf{\sigma_{(1 \ \theta a)} (D)}$ Esta regla se aplica para todas las subfórmulas que son restringidas por $x_i \ \theta$ a.
	Fórmula Bien Formada	
5	$\mathbf{\omega(y_1,...,y_m) \equiv}$ $\mathbf{\omega_1(u_1,...,u_n) \vee \omega_2(v_1,...,v_p)}$ donde cada u_i es un y_j distinto y cada v_i es un y_j distinto (aunque algunos u's y v's pueden ser el mismo y_j)	$\mathbf{\pi_{i1, i2, ..., im} (E_1 \times D^{m-n}) \cup \pi_{j1, j2, ..., jm} (E_2 \times D^{m-p})}$ Donde E_1 y E_2 son expresiones algebraicas para ω_1 y ω_2 respectivamente.
6	$\mathbf{\omega(y_1,...,y_m) \equiv \neg\omega_1(y_1,...,y_m)}$	$\mathbf{D^m - E_1}$ Donde E_1 es una expresión algebraica para ω_1
7	$\mathbf{\omega(y_1,...,y_m) \equiv}$ $\mathbf{(\exists \ y_i) \ (\omega_1(y_1,...,y_n))}$	Sea E_1 es una expresión algebraica para ω_1 • Si $E_1 \equiv (\sigma_F (E_2))$ la traducción es

	o **$\omega(y_1,...,y_m) \equiv$** **$\neg(\forall\ y_i)\ \neg(\omega_1(y_1,...,y_n))$** con $1 \leq i \leq n$	**$\sigma_F\ (\pi_{1, 2, ..., i-1, i+1, ..., m}\ (E_2))$** La condición F debe involucrar solamente atributos **1, 2, .., i-1, i+1, .., m**. Si la condición involucra el atributo i, la traducción es **$\pi_{1, 2, ..., i-1, i+1, ..., m}\ (E_1)$** • Si $E_1 \equiv (\pi_{1, 2, ..., n}\ (E_2))$ la traducción es **$\pi_{1, 2, ..., i-1, i+1, ..., m}\ (E_2)$** • Si $E_1 \equiv (E_2 \times E_3)$ Sea X_{E2} el conjunto de índices correspondientes a los atributos de E_2. Sea X_{E3} el conjunto de índices correspondientes a los atributos de E_3. Sea $A = X_{E2} \cap \{1, 2, ..., i-1, i+1, ..., m\}$ Sea $B = X_{E3} \cap \{1, 2, ..., i-1, i+1, ..., m\}$ La traducción es: **$\pi_A\ (E_2) \times \pi_B\ (E_3)$** Esta traducción también se utiliza si la expresión E es un Join con fórmula y el atributo i no está involucrado en la fórmula. • Si $E_1 \equiv (E_2 \cup E_3)$ la traducción es **$\pi_{1, 2, ..., i-1, i+1, ..., m}\ (E_2) \cup \pi_{1, 2, ..., i-1, i+1, ..., m}\ (E_3)$** Nota: Cuando se encuentra un cuantificador existencial se lo debe introducir lo más adentro posible de la subexpresión y recién en ese lugar aplicar esta regla.
8	**$\omega(y_1,...,y_m) \equiv$** **$\omega_1(u_1,...,u_n) \wedge \omega_2(v_1,...,v_p)$** donde cada u_i es un y_j distinto y cada v_i es un y_j distinto y $\{u_1,...,u_p\} \cap \{v_1,...,v_p\} = \varnothing$	**$E_1 \times E_2$** Donde E_1 y E_2 son expresiones algebraicas para ω_1 y ω_2 respectivamente.
9	**$\omega(y_1,...,y_m) \equiv$** **$\omega_1(u_{i1},...,u_{in}) \wedge \omega_2(v_{i1},...,v_{ip})$** donde cada u_{ik} es un y_k distinto y cada v_{ik} es un y_k distinto y $\{u_{i1},...,u_{in}\} \cap \{v_{i1},...,v_{ip}\} \neq \varnothing$	**$\pi_{i1, i2, ..., mi}\ (E_1\ \Xi_F\ E_2)$** Donde E_1 y E_2 son expresiones algebraicas para ω_1 y ω_2 respectivamente y F es una fórmula que contiene átomos x=y siempre que u_{ix} y v_{iy} sean la misma variable, todos los términos están conectados por el operador $\wedge$. La lista $j_1, j_2, ..., j_m$ es una lista donde se proyectan todas las componentes de E_1, de E_2 sólo se proyectan las componentes que no se encuentran en la parte derecha de cada uno de los átomos de F.
10	**$\omega(y_1,...,y_m) \equiv (\forall\ y_i)\ (\omega_1(y_1,...,y_n))$** ó **$\omega(y_1,...,y_m) \equiv$** **$\neg(\exists\ y_i)\ \neg(\omega_1(y_1,...,y_n))$**	**$E_1 \div D{:}I$** Donde E_1 es una expresión algebraica para ω_1 e I indica cuál es la componente por la que se debe realizar la división.

	con $1 \leq i \leq n$	
11	$\omega(y_1,...,y_m) \equiv \neg(\neg(\omega_1\ (u_1,...,u_n)))$ ó $\omega(y_1,...,y_m) \equiv$ $\omega_1\ (u_1,...,u_n) \vee \omega_1\ (v_1,...,v_n)$ ó $\omega(y_1,...,y_m) \equiv$ $\omega_1\ (u_1,...,u_n) \wedge \omega_1\ (v_1,...,v_n)$	$\mathbf{E_1}$ Donde E_1 es una expresión algebraica para ω_1, con $\{u_1,...,u_n\} = \{v_1,...,v_n\}$.
12	$\omega(y_1,...,y_m) \equiv$ $\omega_1(u_1,...,u_n) \wedge \omega_2(v_1,...,v_n)$	$\mathbf{E_1}$ Donde E_1 y E_2 son expresiones algebraicas para ω_1 y ω_2 respectivamente tal que E_2 es D^m , con $\{u_1,...,u_n\} = \{v_1,...,v_n\}$.
13	$\omega(y_1,...,y_m) \equiv$ $\omega_1(u_1,...,u_n) \wedge \omega_2(v_1,...,v_n)$	$\mathbf{E_2}$ Donde E_1 y E_2 son expresiones algebraicas para ω_1 y ω_2 respectivamente tal que E_2 es $\varnothing^m$, con $\{u_1,...,u_n\} = \{v_1,...,v_n\}$.
14	$\omega(y_1,...,y_m) \equiv$ $\omega_1\ (u_1,...,u_n) \wedge \neg\omega_1\ (v_1,...,v_n)$	$\mathbf{\varnothing^m}$ Donde E_1 es una expresión algebraicas para ω_1, con $\{u_1,...,u_n\} = \{v_1,...,v_n\}$.
15	$\omega(y_1,...,y_m) \equiv$ $\omega_1\ (u_1,...,u_n) \vee \omega_2(v_1,...,v_n)$	$\mathbf{E_1}$ Donde E_1 y E_2 son expresiones algebraicas para ω_1 y ω_2 respectivamente tal que E_2 es $\varnothing^m$, con $\{u_1,...,u_n\} = \{v_1,...,v_n\}$.
16	$\omega(y_1,...,y_m) \equiv$ $\omega_1\ (u_1,...,u_n) \vee \neg\omega_1\ (v_1,...,v_n)$	$\mathbf{D^m}$ Donde E_1 es una expresión algebraicas para ω_1, con $\{u_1,...,u_n\} = \{v_1,...,v_n\}$.
17	$\omega(y_1,...,y_m) \equiv$ $\omega_1\ (u_1,...,u_n) \vee \omega_2(v_1,...,v_n)$	$\mathbf{D^m}$ Donde E_1 y E_2 son expresiones algebraicas para ω_1 y ω_2 respectivamente tal que E_2 es D^m , con $\{u_1,...,u_n\} = \{v_1,...,v_n\}$.

Tabla IV. Expresiones en DRC equivalentes a expresiones en AR con estrategias de optimización incluidas

Las reglas 2, 3, 4 y 7 están relacionadas a la aplicación de los operadores proyección y selección; por lo tanto han sido creadas con el objetivo de reducir las cardinalidades de las relaciones intermedias obtenidas en las evaluaciones de las subfórmulas. Las mismas están basadas fundamentalmente en la estrategia de optimización N° 1 y, complementariamente, en la Nº3.

La regla 2 (segundo punto) y la regla 9 han sido creadas con el objetivo de realizar Join en todos los sitios que sea posible, en lugar de producto cartesiano, dado que apuntan a reducir las

cardinalidades de las relaciones generadas.

Las reglas 11, 12, 13, 14, 15, 16, 17 y las fórmulas alternativas de las reglas 7 y 10 (expresadas en DRC), surgen de la equivalencia de expresiones lógicas vistas en la Tabla III, y ellas apuntan a quitar subexpresiones inútiles.

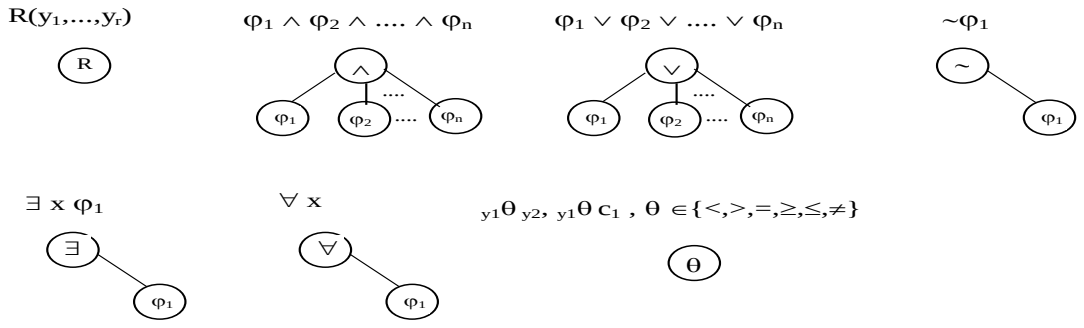
La estrategia de traducción con optimización simultánea consta de la aplicación de las reglas presentadas en la Tabla IV, para cada una de las subexpresiones de la expresión en DRC a traducir, obteniéndose una expresión en AR optimizada y equivalente.

5.2 ESTRATEGIA

A continuación describiremos cómo se realiza la transformación con la optimización incluida. En cuanto a las reglas que se nombrarán corresponden a las presentadas en la sección 5.1, que son las implementadas en nuestra herramienta.

Para la traducción con optimización simultánea, se toma la fórmula φ , se le realiza el análisis sintáctico y se construye un árbol r-ario de expresión correpondiente a φ .

El árbol se construye inductivamente de la siguiente forma: Si R (y_1, …,y_r) es una fórmula atómica, φ_1, φ_2, …, φ_n son fórmulas bien formadas, e y_1, …,y_r , x son variables de individuo, se muestran los subárboles correspondientes para cada caso:



Las variables de individuo y/o constantes forman parte de la información del nodo en cuestión.

Se definen nodos de distinto tipo que determinarán la acción semántica a seguir en el proceso de traducción con optimización.

El árbol se recorre recursivamente en postorder, dado que con este recorrido cada vez que visita un nodo es porque ya recorrió sus subárboles, y por lo tanto tiene los resultados de sus análisis disponibles para realizar la acción semántica que le requiere el tipo de mismo.

Acciones Semánticas

a) Tipo de nodo Átomo de Comparación: Este nodo se traduce aplicando las reglas 2,3 o 4 al momento de traducir nodos que contienen variables que participan en la comparación .
b) Tipo de nodo Átomo de Relación (R): En este punto se aplica la regla 1.
c) Tipo de nodo Conectivo $\wedge$: En el análisis de esta conectiva, se toman de a pares los subárboles dependientes. Esto es, transformando el árbol r-ario a binario temporalmente. Entonces se busca la aplicación de las reglas 8, 9, 12, 13 o 14. También se trata de asociar subárboles buscando la aplicación de la regla 9.
d) Tipo de nodo Conectivo $\vee$: En el análisis de esta conectiva, se toman de a pares los subárboles dependientes. Esto es, transformando el árbol r-ario a binario temporalmente. Entonces, se busca la aplicación de las reglas 5, 15, 16 o 17.

e) Tipo de nodo Conectivo ~ :Se aplica la regla 6, 7, 10 o 11.
f) Tipo de nodo Cuantificador ∀: Se aplica la regla 10.
g) Tipo de nodo Cuantificador ∃: Se aplica la regla 7.

En cada nivel del árbol se analiza el tipo de nodo y busca la aplicación de las reglas vistas, asi hasta terminar con la raíz del mismo, en cuyo momento se ha logrado la finalización del pasaje del DRC al AR y además la expresión ya está optimizada.

6. EJEMPLOS

Presentamos una base de datos con fines didácticos para mostrar dos casos de consultas.

Dado el vocabulario $\boldsymbol{\sigma}$ = < Cursos$^{(2)}$, Profesores$^{(2)}$, Alumnos$^{(2)}$, Dictado-por$^{(2)}$, Tomado-por$^{(2)}$, c_1 , c_2 >
Sea la σ-estructura $\boldsymbol{B}$ = <D^B, Cursos^B, Profesores^B, Alumnos^B, Dictado-por^B, Tomado-por^B, $c_1{}^B$, $c_2{}^B$ >
Sea el dominio $\mathbf{D^B}$ = { d_1, d_2,…,d_n } con |D^B | = n.
$c_1{}^B$ = d_i ; $c_2{}^B$ = d_j con d_i, $d_j \in D^B$.
Dictado-por^B = { (x,y) / x ∈ Cursos ∧ y ∈ Profesores ∧ "el curso x es dictado por el profesor y"};
Es una relación definida entre Cursos y Profesores, que satisface las propiedades inyectividad y suryectividad, tipo (1:n).
Tomado-por^B = { (x,y) / x ∈ Cursos ∧ y ∈ Alumnos ∧ "el curso x es tomado por el alumno y"};
Es una relación definida entre Cursos y Alumnos, irrestricta, tipo (n:m).

6.1 EJEMPLO Nº 1 DE CONSULTA

- Query: *"Obtener los Nombres de los alumnos que toman al menos un curso dictado por los profesores $c_1{}^B$ o $c_2{}^B$"*

- Expresado en DRC:
  $\varphi(x) \equiv \exists y(\exists z(\exists w(\text{Dictado-por}(z,w) \wedge \text{Tomado-por}(z,y) \wedge \text{Alumnos}(y,x)\ (w = c_1 \vee w = c_2))))$

- Hagamos una Traducción con posterior Optimización
  Aplicaremos traducción y a posteriori la optimización, [ULLMAN 79] [ULLMAN 88], para luego mostrar la diferencia con nuestra estrategia:

  Paso 1: Traducción del DRC al AR
  1º: Transformación del query φ a otro φ' equivalente utilizando únicamente el conjunto adecuado de conectivos {∨, ¬} y el cuantificador existencial.

$$\varphi'(x) \equiv \exists y(\exists z(\exists w(\neg(\neg\text{Dictado-por}(z,w) \vee \neg\text{Tomado-por}(z,y) \vee \neg\text{Alumnos}(y,x) \vee \neg(w = c_1 \vee w = c_2)))))$$

  2º: Traducción de la expresión φ' al AR. Con E_1, E_2 y E_3 denotaremos tres subexpresiones del AR, para mayor claridad.

$E_1 \equiv ((D^2 - \text{Dictado-por}) \times D) \cup (\pi_{1,3,2}((D^2 - \text{Tomado-por}) \times D))$
$E_2 \equiv (E_1 \times D) \cup (\pi_{3,4,1,2}((D^2 - \text{Alumnos}) \times D^2))$
$E_3 \equiv ((D - (\sigma_{(1=C1)}(D) \cup \sigma_{(1=C2)}(D))) \times D^3)$

  Finalmente la expresión en AR obtenida es:

$$E_{AR} \equiv \pi_2 (\pi_{2,3} (\pi_{1,3,4} ((D^4) - (E_2 \cup \pi_{2,1,3,4} (E_3))))) \equiv \varphi' (x)$$

Paso 2: Optimización

$E_1 \equiv ((D^2 \text{ - Dictado-por}) \times D) \cup (\pi_{1,3,2} ((D^2 \text{ - Tomado-por}) \times D))$

$E_2 \equiv (E_1 \times D) \cup (\pi_{3,4,1,2} ((D^2 \text{ - Alumnos}) \times D^2))$

$E_3 \equiv ((D - (\sigma_{(1=C1)} (D) \cup \sigma_{(1=C2)} (D))) \times D^3)$

Finalmente la expresión en AR Optimizada obtenida es:

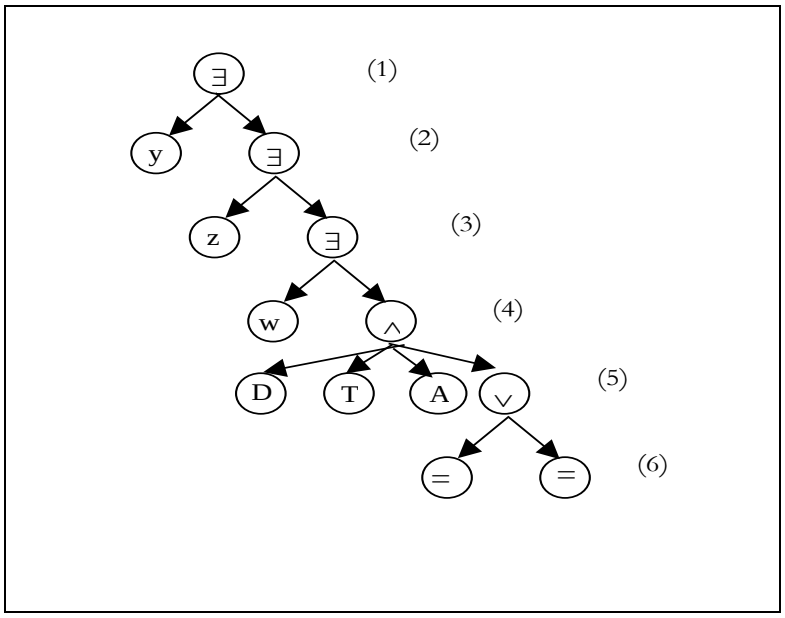
$$\mathbf{E_{ARO} \equiv \pi_4 ((D^4) - (E_2 \cup \pi_{2,1,3,4} (E_3))) \equiv \varphi' (x)}$$

- Aplicando nuestra estrategia: Traducción con Optimización Simultánea

A continuación se aplicará la *traducción con optimización simultánea* al query φ (x), de manera tal que se pueda comparar la expresión resultante con la expresión obtenida anteriormente.

Paso 1: hacer el análisis sintáctico y obtener el árbol de expresión para φ (x),

Árbol de Expresión para φ (*x*)



Observar: La relación Dictado-por se representa en el árbol por D, la relación Tomado-por se representa por T y la relación Alumnos se representa por A. En los nodos D, T y A se encuentran asociadas las variables que cada uno posee en el query. En los nodos del nivel 6, se encuentran las variables y/o constantes correspondientes a los átomos de comparación.

Paso 2: al árbol obtenido se lo recorre en postorden y se van aplicando las reglas de la Tabla IV según la subexpresión que se esté tratando.

- Los nodos de los subárboles (6) se traducen aplicando la regla 4.

$$E_1 \equiv (\sigma_{(2=C1 \vee 2=C2)} (\text{Dictado-por}))$$

- Los nodos del subárbol (4) se traducen aplicando la regla 9.

$$E_2 \equiv \pi_{1,2,3,5} ((\pi_{1,2,4} (E_1 \, \Xi_{1=1} \text{ Tomado-por})) \, \Xi_{3=1} \text{ Alumnos}))$$

- Los nodos del subárbol (1,2 y 3) se traducen aplicando la regla 7.

$$E_3 \equiv \pi_3 ((\pi_3 (\pi_1 (\sigma_{(2=C1 \vee 2=C2)} (\text{Dictado-por})) \Xi_{1=1} \text{Tomado-por})) \Xi_{1=1} \text{Alumnos})$$

Luego de recorrer todo el árbol y haber aplicado la reglas el query resultante y optimizado es:

$$\mathbf{E_{ARO''} \equiv \pi_3 ((\pi_3 (\pi_1 (\sigma_{(2=C1 \vee 2=C2)} (Dictado\text{-}por)) \Xi_{1=1} Tomado\text{-}por)) \Xi_{1=1} Alumnos)}$$

6.2 EJEMPLO Nº 2 DE CONSULTA

- Query: *"Obtener los Nombres de todos los cursos que fueron tomados por solamente un alumno"*

- Expresado en DRC

  $\varphi(x) \equiv \exists z(\exists w(\text{Tomado-por}(z,w) \wedge \neg\exists y(\text{Tomado-por}(z,y) \wedge w<>y) \wedge \text{Cursos}(z,x)))$

- Hagamos una Traducción con posterior Optimización

  Paso 1: Traducción del DRC al AR

  1º: Transformación del query φ a otro φ' equivalente utilizando únicamente el conjunto adecuado de conectivos $\{\vee, \neg\}$ y el cuantificador existencial.

  $\varphi(x) \equiv \exists z(\exists w(\neg(\neg\text{Tomado-por}(z,w) \vee \exists y(\neg(\neg\text{Tomado-por}(z,y) \vee w=y)) \vee \neg\text{Cursos}(z,x))))$

  2º: Traducción de la expresión φ' al AR. Denotaremos con E_1 y E_2 dos subexpresiones en AR para mayor claridad.

  $E_1 \equiv (D^2 - \text{Tomado-por}) \cup (\pi_{1,3} (D^3 - (((D^2 - \text{Tomado-por}) \times D) \cup \pi_{3,2,1} ((\sigma_{(1=2)} (D \times D)) \times D))))$

  $E_2 \equiv (E_1 \times D) \cup (\pi_{1,3,1} ((D^2 - \text{Cursos}) \times D))$

  La expresión en AR es:

  $$E_{AR} \equiv \pi_2 (\pi_{1,3} (D^3 - E_2)) \equiv \varphi'(x)$$

  Paso 2: Optimización

  $E_1 \equiv (D^2 - \text{Tomado-por}) \cup (\pi_{1,3} (D^3 - (((D^2 - \text{Tomado-por}) \times D) \cup \pi_{3,2,1} ((\sigma_{(1=2)} (D \times D)) \times D))))$

  $E_2 \equiv (E_1 \times D) \cup (\pi_{1,3,1} ((D^2 - \text{Cursos}) \times D))$

  Finalmente la expresión en AR Optimizada obtenida es:

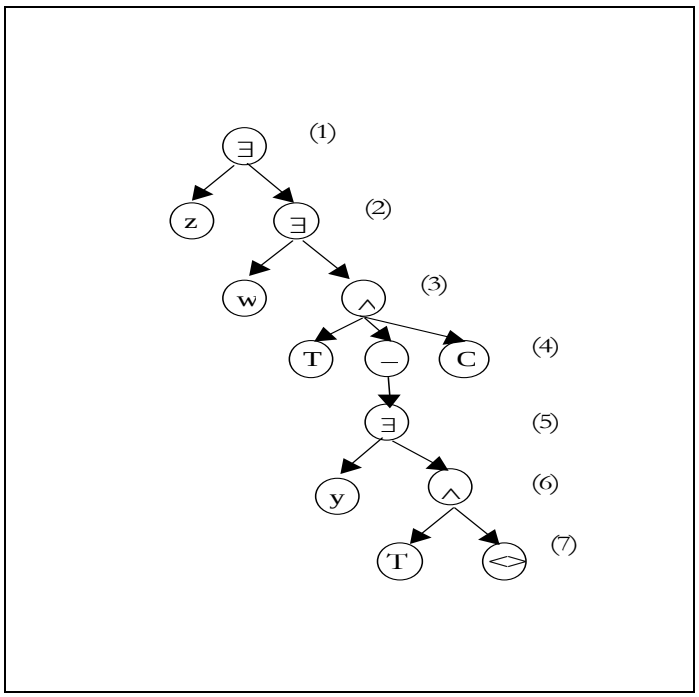
  $$\mathbf{E_{ARO} \equiv \pi_3 (D^3 - E_2) \equiv \varphi'(x)}$$

- Aplicando nuestra estrategia: Traducción con Optimización Simultánea

  Paso 1: hacer el análisis sintáctico y obtener el árbol de expresión para $\varphi(x)$,

  Observar: La relación Tomado-por se representa en el árbol por T y la relación Cursos se representa por C. En los nodos T y C se encuentran asociadas las variables que cada uno posee en el query.

Árbol de Expresión para φ (*x*)



Paso 2: al árbol obtenido se lo recorre en postorden y se van aplicando las reglas de la Tabla IV según la subexpresión que se esté tratando.

- Los nodos de los subárboles (6) se traducen aplicando la regla 2.

$$E_1 \equiv (\sigma_{(1<>2)} (D^2))$$

- Los nodos del subárbol (5) se traducen aplicando la regla 9 y la regla 7.

$$E_2 \equiv \pi_{1,3} (\text{Tomado-por } \Xi_{2=2} E_1)$$

- Los nodos del subárbol (3) se traducen aplicando la regla 9, la regla 6 y la regla 7.

$$E_3 \equiv \pi_{1,2,4} ((\pi_{1,2} (\text{Tomado-por } \Xi_{(1=1 \wedge 2=2)} (D^2 - E_2))) \Xi_{1=1} \text{Cursos})$$

- Los nodos del subárbol (1 y 2) se traducen aplicando la regla 7.

$$E_3 \equiv \pi_3 ((\pi_1 (\text{Tomado-por } \Xi_{(1=1 \wedge 2=2)} (D^2 - E_2))) \Xi_{1=1} \text{Cursos})$$

Luego de recorrer todo el árbol y haber aplicado la reglas el query resultante es:

$$\mathbf{E_{ARO''} \equiv \pi_3 ((\pi_1 (Tomado\text{-}por\ \Xi_{(1=1 \wedge 2=2)} (D^2 - E_2))) \Xi_{1=1} Cursos)}$$

6.3 EJEMPLO Nº 3 DE CONSULTA

- Query: *"Obtener el Nombre de los alumnos que tomaron todos los cursos"*

- Expresado en DRC:

$$\varphi (x) \equiv \exists w(\text{Alumno}(w,x) \wedge \forall z(\neg(\exists y(\text{cursos}(z,y))) \vee \text{Tomado-por}(z,w)))$$

- Hagamos una Traducción con posterior Optimización

Aplicaremos traducción y a posteriori la optimización, para luego en el siguiente punto mostrar la

diferencia con nuestra estrategia:

Paso 1: Traducción del DRC al AR

1º: Transformación del query φ a otro φ' equivalente utilizando únicamente el conjunto adecuado de conectivos $\{\vee, \neg\}$ y el cuantificador existencial.

$$\varphi(x) \equiv \exists w(\neg(\neg(Alumno(w,x)) \vee \exists z(\neg(\neg(\exists y(cursos(z,y))) \vee Tomado\text{-}por(z,w)))))$$

2º: Traducción de la expresión φ' al AR. Denotaremos con E_1 y E_2 dos subexpresiones en AR, para mayor claridad.

$E_1 \equiv (D^2 - Alumno) \cup ((\pi_2 (D^2 - (((D - \pi_1 (cursos)) \times D) \cup Tomado\text{-}por))) \times D)$
$E_2 \equiv \pi_2 (D^2 - E_1)$

Finalmente la expresión en AR obtenida es:

$$\mathbf{E_{AR} \equiv \pi_2 (D^2 - E_1)}$$

Paso 2: Optimización

No se le puede realizar ninguna transformación que implique optimización.

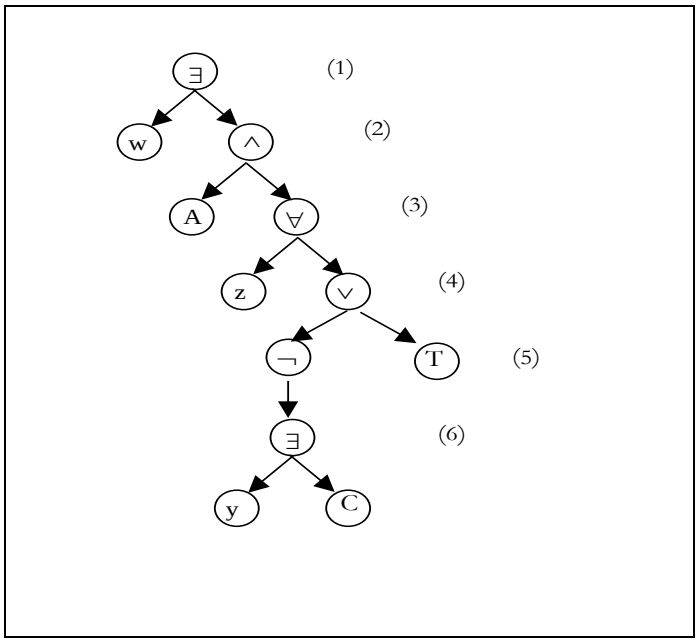
$$\mathbf{E_{AR} \equiv \pi_2 (D^2 - E_1)}$$

- Aplicando nuestra estrategia: Traducción con Optimización Simultánea

A continuación se aplicará la *traducción con optimización simultánea* al query φ **(x)**, de manera tal que se pueda comparar la expresión resultante con la expresión obtenida anteriormente.

Paso 1:

Árbol de Expresión para φ (*x*)



Observar: La relación Tomado-por se representa en el árbol por T, la relación Alumnos se representa por A y la relación Cursos se representa por C. En los nodos T, A y C se encuentran asociadas las variables que cada uno posee en el query.

Paso 2: al árbol obtenido se lo recorre en postorden y se van aplicando las reglas de la Tabla IV según la subexpresión que se esté tratando.

- Los nodos de los subárboles (7) se traducen aplicando la regla 4.

$$E_1 \equiv \pi_1 \text{ (Cursos)}$$

- Los nodos del subárbol (6) se traducen aplicando la regla 9 y la regla 7.

$$E_2 \equiv (D - E_1)$$

- Los nodos del subárbol (5) se traducen aplicando la regla 9, la regla 6 y la regla 7.

$$E_3 \equiv (E_2 \text{ x } D) \text{ U Tomado-por}$$

- Los nodos del subárbol (10) se traducen aplicando la regla 7.

$$E_4 \equiv E_3 \div D{:}1$$

- Los nodos del subárbol (9) se traducen aplicando la regla 9 y la regla 7.

$$E_5 \equiv \pi_{1,2} \text{ (Alumno } \Xi_{1=1} E_4)$$

- Los nodos del subárbol (7) se traducen aplicando la regla 7.

$$E_3 \equiv \pi_2 (E_5)$$

Luego de recorrer todo el árbol y haber aplicado la reglas el query resultante es:

$$\mathbf{E_{ARO''} \equiv \pi_2 (\pi_{1,2} (Alumno\ \Xi_{1=1} ((((D - \pi_1 (Cursos)) \text{ x } D) \text{ U Tomado-por}) \div D{:}1)))}$$

Observando las expresiones obtenidas de las dos estrategias utilizadas podemos concluir que la expresión que se obtuvo de la “estrategia de traducción con optimización simultánea” cumple con más condiciones de optimización que las otras.

CONCLUSIONES

Con este trabajo, nos propusimos implementar un traductor del DRC al AR con optimización de expresiones, considerando toda la familia de queries expresables en AR, y no subclases de dicha familia de la misma.

Para ello, se tomó como marco de referencia la complejidad de las operaciones algebraicas y su influencia en los volúmenes o cardinalidades de las relaciones temporarias que se generan en la evaluación del query, haciendo optimizaciones de manera estática, desde el comienzo de la traducción hasta su finalización.

Nótese que tomamos como base el teorema de traducción del DRC al AR y técnicas de estrategias de optimización, pero que no se utilizaron sino que de ellos se obtuvo una herramienta diferente dado que ésta traduce y optimiza simultáneamente.

Por las experiencias realizadas, dadas expresiones en DRC que pueden contener subfórmulas con conectivo conjuntivo, cuantificadores Universales o subfórmulas atómicas que contienen operadores de comparación, se les aplicó los dos algoritmos de traducción y optimización separada, y el resultado fue totalmente diferente al que se obtuvo de aplicarle esta herramienta.

Futuras extensiones al presente trabajo están pensadas para bases de datos distribuidas, en donde el optimizador pueda desglosar el query en subqueries de acuerdo a la fragmentación de la base, y sobre ellos buscar la traducción con optimización de expresiones.

Con esta herramienta se contribuye a la construcción de una biblioteca de distintos tipos de formalismos existentes en nuestra área de investigación, con diferentes grados de expresividad para formular consultas a Bases de Datos Relacionales. Estos trabajos se encuentran disponibles como herramientas didácticas y de investigación [BARROSO 97], [GAGLIARDI 88], [MALDOCENA 99], [PEREYRA 98].

REFERENCIAS BIBLIOGRÁFICAS

[ABITEBOUL 95] Abiteboul,S; Hull and Vianu, V.; "Foundations of Databases". Addison-Wesley Publishing Company, 1995.

[ABITEBOUL 91] Abiteboul,S; Vianu, V.; "Generic Computation and Its Complexity", STOC 1991.

[BARROSO 97] Barroso, L.; Molina, G.; Quiroga, J.; "Implementación de un Interprete para el Lenguaje QL", Reporte Licenciatura en Cs. de la Computación, UNSL, 1997.

[CHANG 92] Chang C.C.; Keisler H.J.; "Model Theory" Elsevier Science Publishers3º edition 1992.

[CHANDRA 80] Chandra, A.K.; Harel, D. "Computable Queries for Relational Data Bases". Journal of Computer and System Sciences 21, 156-178. 1980.

[COD 70] Codd, E.F.; "A relational model of data for a large shared data banks". Com of ACM 13/6):377-387,1970.

[EBBINGHAUS 84] Ebbinghaus, H; Flum, J.;Thomas, W.; "Mathematical Logic", Springer-Verlag, 1984.

[EBBINGHAUS 95] Ebbinghaus, H; Flum, J.; "Finite Model Theory" , Springer-Verlag, 1995.

[GAGLIARDI 88] Gagliardi, E.; Quintas, S.; "Evaluador de Expresiones Algebraicas", Reporte Licenciatura en Cs. de la Computación, UNSL, 1998.

[HAMILTON 81] Hamilton, A.G.; "Lógica para matemáticos", Paraninfo, 1981.

[MAIER 83] Maier, D.. "The Theory of Relational Databases". Computers Science Press. 1983.

[MALDOCENA 99] Maldocena,P.; Reyes,N; "Expresiones de Consultas a Bases de Datos mediante Extensiones de Lógica de Primer Orden", Reporte Licenciatura en Cs. de la Computación, UNSL, 1999.

[OZSU 91] Ozsu, T.; Valduriez, P. "Principles of Distributed Database Systems", Prentice Hall, Upper Saddle River, New Jersey 07458, 1991

[PEREYRA 98] Pereyra,S; Piffaretti,P. Reporte Licenciatura en Cs. de la Computación, UNSL, 1998.

[TURULL 96] Turull Torres, J.M. "Clases de Bases de Datos L-Rígidas y Expresividad de

Lenguajes Relacionales Incompletos". Tesis Doctoral, UNSL, 1996.

[ULLMAN 79] Ullman, Jeffrey D. "Principles of Database Systems". Second Edition - Computers Science Press.

[ULLMAN 88] Ullman, Jeffrey D. "Principles of Database and Knowledge Base Systems". Computers Science Press, 1988.

[YRUSTA 99] Yrusta,M. Reporte Licenciatura en Cs. de la Computación, UNSL, 1999.