다중 정규 경로 질의 처리를 위한 효율적 기법
(An Efficient Technique for Evaluating Queries with Multiple Regular Path Expressions)

정 태 선 † 김 형 주 ††
(Tae-Sun Chung) (Hyoung-Joo Kim)

요 약 최근에 XML이 웹 상에서 문서 교환의 표준으로 등장하면서 XML로 표현된 데이타에 대한 질의 처리 분야가 주목받고 있다. 이때 XML 질의는 그래프로 표현된 데이타 그래프에서 특정 정규식으로 도달되는 객체를 찾는 정규 경로 질의(regular path query)를 기반으로 한다. 그런데 사용자의 다양한 형태의 질의를 처리하기 위해서는 질의에 하나 이상의 정규식을 가지는 질의의 처리가 필요함에도 기존의 연구 즉, 비정형 데이타 모델 하에서의 뷰를 이용한 질의 변환(query rewriting)이나, 질의 최적화 기법에서는 주로 단일 정규식으로 이루어진 질의를 다루었다. 본 논문에서는 이러한 다중 정규식을 가지는 질의의 처리에서 1. 뷰의 몸체에서 질의 몸체로의 변수 매핑을 통한 질의 변환 과정과 2. 변환된 질의의 각 조각(conjunct)의 질의 결과를 효율적으로 구하고 결과를 조합하는 두 단계로 이루어진 효율적인 질의 처리 기법을 제안한다. 제안하는 질의 변환 알고리즘이 안전성(soundness)을 가짐을 보이고, 질의 처리 기법이 기존 질의 처리 방식에 비하여 효율적임을 보인다.

Abstract As XML has become an emerging standard for information exchange on the World Wide Web, it has gained attention in database communities to extract information from XML seen as a database model. XML queries are based on regular path queries, which find objects reachable by given regular expressions. To answer many kinds of user queries, it is necessary to evaluate queries that have multiple regular path expressions. However, previous work such as query rewriting and query optimization in the frame work of semistructured data has dealt with a single regular expression. For queries that have multiple regular expressions we suggest a two phase optimizing technique: 1. query rewriting using views by finding the mappings from the view's body to the query's body and 2. for rewritten queries, evaluating each query conjunct and combining them. We show that our rewriting algorithm is sound and our query evaluation technique is more efficient than the previous work on optimizing semistructured queries.

1. 서 론

최근에 XML이 웹 상에서 문서 교환의 표준으로 채택되면서 XML을 데이타베이스의 데이타로 보고 정보를 추출하는 분야가 주목받고 있다. 즉, XML은 스스로 자신을 묘사하는(self describing) 성질을 가지므로, 우리는 네트워크 상에 분산된 이질의(heterogeneous) 정보 근원으로부터 질의를 수행하고 필요한 정보를 추출해 낼 수 있다.

XML 데이타는 레이블이 있는 그래프에 기반 한 비정형 데이타 모델의 한 인스턴스로 볼 수 있으므로 비정형 데이타 모델에 매핑되어 질의가 수행될 수 있다. 이때 질의는 정규식으로 표현되는 정규 경로 질의(regular path query)를 기반으로 한다. 예를 들면 정규 경로 질의 (_*.movie).(_*.actor._*.(Tom Cruise|Brad Pitt))는 데이타 그래프의 경로 중 어떤 시점에서 에지 movie를 가지고, 이어서 임의의 에지를 가진 후, actor에지를 가지고 다시 임의의 에지를 가지고 Tom Cruise나 Brad Pitt 에지를 마지막으로 가지는 모든 경로를 의미한다.

그런데 사용자의 질의는 하나의 정규 경로식을 가지는

• 본 연구는 두뇌한국21 사업에 의하여 지원 받았음.
† 비 회 원 : 서울대학교 컴퓨터공학부
tschung@papaya.snu.ac.kr
†† 종신회원 : 서울대학교 컴퓨터공학부 교수
hjk@oopsla.snu.ac.kr
논문접수 : 2000년 7월 6일
심사완료 : 2001년 3월 23일

질의보다는 다중 정규 경로식을 가지는 경우가 많다. 예를 들면 위의 예에서 사용자가 "배우 톰 쿠르즈 혹은 브래드 피트가 주연한 영화 중 2000년에 만들어진 영화를 찾아라"라는 질의를 던진다면 질의는 다음과 같을 것이다(질의어와 그 의미는 2절에서 구체적으로 정의된다).

$q(p_2)$:- p_1 (_*.movie) p_2,p_2 (_*.year.2000) p_3, p_2 (_*.actor._*.(Tom Cruise|Brad Pitt)) p_4

위와 같은 형태의 질의는 정규식을 하나 이상 포함한 형태로 XML 문서에 대한 질의에서 많이 나타나는 형태의 질의이지만 기존의 연구, 즉 비정형 데이타 모델에서의 질의 변환[1,2]이나 질의 최적화 기법[3]에서는 주로 단일 정규식으로 이루어진 문제를 다루었다.

우리는 이러한 다중 정규식을 가지는 질의의 처리에서 1. 뷰의 몸체에서 질의 몸체로의 변수 매핑을 통한 질의 변환 과정과 2. 변환된 질의의 각 조각(conjunct)의 질의 결과를 효율적으로 구하고 결과를 조합하는 두 단계로 이루어진 향상된 질의 처리 기법을 제안한다.

논문의 구성은 다음과 같다. 2절에서는 관련 연구를 다루고, 3절에서는 데이타 모델과 질의 언어를 정의한다. 4절에서는 뷰의 정의와 질의 변환 알고리즘을 다룬다. 5절에서는 제안하는 질의 처리 알고리즘을 보이고, 6절에서 결론을 맺는다.

2. 관련 연구

먼저 우리의 알고리즘이 단계 1에서 사용하는 형상화된 뷰(materialized view)는 관계형 모델[4,5,6,7]이나 객체 모델[8,9]에서 뷰의 정의, 점진적인 유지(incremental maintenance), 뷰를 이용한 질의 변환 등의 활발한 연구가 수행되었다. 특히 [7]에서는 질의 처리 계획 영역에 형상화된 뷰를 첨가한 후, 전체 질의 처리 계획 영역에서 최적의 질의 처리 계획을 찾는 알고리즘을 제안하였다. 본 논문의 연구는 [7]의 연구를 비정형 데이타 모델로 확장한 것이라고 할 수 있다.

비정형 데이타 모델에서 뷰를 이용한 질의 변환에 대한 연구로는 [1,2,10,11]가 있다. [11]에서는 TSL이라는 언어를 기반으로 뷰를 이용한 질의 변환 알고리즘을 제안하고, 그 알고리즘의 안전성(soundness)과 완전성(completeness)을 보였다. TSL은 근원 데이타의 재구성(restructuring)이 가능하지만 정규 경로식(regular path expression)은 지원하지 못한다. [10]에서는 비정형 데이타 모델에서 $STRUQL_0$으로 표현된 질의의 포함(containment) 문제의 복잡도를 다뤘다. [1,2]에서는 질의 q와 뷰의 집합 $V=\{v_1,...,v_n\}$가 주어졌을 때 질의 q를 V에 속하는 하나 이상의 뷰를 이용한 질의 q′으로 바꾸는 알고리즘을 제안하였다. 우리의 알고리즘과 다른 점은 [1,2]에서는 질의와 뷰가 단일 정규식으로만 이루어져 있다는 점이다.

뷰를 직접 다루지는 않았지만 비정형 데이타 모델에서 효율적인 질의 처리를 위한 여러 기법이 제안되었다[3,12,13,14,15]. 이 중에서 그래프 스키마[14]와 DataGuide[3]는 비정형 데이타 모델에서 스키마를 추출하여 질의 처리에 이용하는 방법이다. 그런데 이 방법은 질의상에 여러 개의 정규식이 있을 때 적용될 수 없다는 단점을 가진다.

[15]의 연구에서는 1-index, 2-index, T-index의 세 가지 방법을 제안하였다. 1-index는 루트 노드에서 정규 경로식에 의하여 도달되는 객체의 집합을 효율적으로 찾는 인덱스 기법으로 DataGuide와 비슷한 기능을 하지만 인덱스 그래프를 오토마타로 보았을 때 비결정적(nondeterministic) 성질을 가짐으로써 인덱스의 크기가 근원 데이타베이스의 크기에 선형적으로 비례하도록 하였다. 2-index는 특정 정규 경로식을 만족하는 임의의 두 노드를 찾는 효율적인 기법으로 본 연구에서 이용하였다. T-index는 다중의 정규식에 대해서도 템플릿 형태로 인덱스를 걸 수 있도록 한다. 그렇지만 처리해야 할 질의가 주어진 템플릿을 만족해야만 한다.

비정형 데이타 모델에서도 비용 기반(cost-based) 질의 최적화 기법이 제안되었다[12,13]. 여기서는 비정형 데이타 모델 특유의 비용 모델, 데이타베이스 통계 정보(statistics), 새로운 인덱스 등을 이용하여 비용 계산에 기반한 질의 처리 계획을 생성한다. 우리의 연구도 하나의 질의 처리 계획으로 첨가될 수 있을 것이다.

3. 데이타 모델과 질의 언어

비정형 데이타(semi-structured data)는 일반적으로 레이블이 있는 그래프(labeled graph)로 표현되어, 그래프의 노드는 객체에 대응되고, 에지는 그 객체의 애트리뷰트에 대응된다. 본 논문에서는 데이타 그래프 DB를 다음과 같이 정의한다. 먼저 O를 무한한(infinite) 객체 식별자(object identity)의 집합, C를 O와 교집합을 가지지 않는 무한한 상수의 집합이라 가정한다.

정의 1 데이타 그래프 DB=(V,E,R)는 루트 노드를 가진 그래프로서 $V \subset O$는 노드의 집합, $E \subseteq V \times C \times V$는 방향성 있는 에지(directed edge)의 집합, 그리고 $R \in V$는 루트 노드를 의미한다.

XML 데이타는 XML의 요소(element)를 노드 V에, 요소-애트리뷰트 관계와 요소-부요소(subelement) 관계, 요소간 참조 관계를 에지 E에 매핑하고 XML 문서

의 값을 그래프의 단말(leaf) 노드에 매핑하면 비정형 데이타 모델에 매핑될 수 있다. 이때, 그래프 모델은 요소-부요소 관계와 요소간의 참조 관계(IDREF)를 구분하지 않고, 애트리뷰트와 부요소 관계를 구분하지 않으므로 XML 문서의 특정 정보를 잃어버릴 수 있지만 간단한 데이타구조를 첨가함으로써 XML 데이타를 그래프 모델로 매핑할 수 있다.

이러한 DB 그래프에서 질의는 다음의 정규 경로식(regular path expression)을 기반으로 한다.

정의 2 정규 경로식은 r =ε|a|_|($r_1.r_2$)|(r_1|r_2)|r* 으로 표현된다. 여기서 r, r_1, r_2는 정규 경로식, a∈C는 특정 상수, _은 임의의 레이블, ε은 빈 문자열(empty string)을 나타낸다.

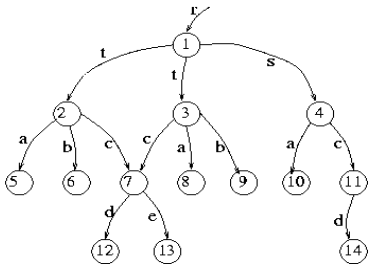

그림 1 예제 데이타 그래프

이러한 정규 경로식을 기반으로 하는 질의 언어로는 UnQL[16], Lorel[17], XML-QL[18], $STRUQL_0$[10] 그리고 마이크로소프트의 XQL 등이 제안되었다. 본 논문에서는 이중에서 논리곱 질의(conjunctive query)의 형태인 $STRUQL_0$의 구문을 사용하였다. 단, XML 질의 언어 중에서 XML-QL은 언어 자체에 질의의 결과를 XML 문서의 형태로 재구성할 수 있는 특징을 가지는데 반하여 우리의 언어는 질의 언어 자체에 질의의 결과를 XML 문서 형태로 만들어 주는 특징은 지니고 있지 못하다. 하지만 질의 처리기의 관점에서 우리의 기법은 정규 경로식을 기반으로 하는 모든 질의 언어에 적용될 수 있을 것이다. 질의어의 의미는 다음과 같이 재정의 된다.

정의 3 질의 q는 q(**u**) :- $y_0r_0z_0$, ..., $y_{n-1}r_{n-1}z_{n-1}$로 표현된다. nvar(q)={y_0, z_0, y_1, z_1, ..., z_{n-1}}은 데이타 그래프 DB의 노드에 매핑되는 변수를 의미한다. 이때 각 변수는 서로 같을 수 있다. r_0, ..., r_{n-1}은 정규 경로식을 나타낸다. **u**⊆nvar(q)는 질의 결과에 포함되는 DB 상의 노드에 매핑되는 변수를 의미한다. 여기서 질의는 다음과 같은 분규 정규 경로식(branching regular path expression)[1)]의 성질을 만족한다. 각 질의 조각 y_i r_i z_i ($0 \le i \le n-1$)에 대하여 y_i를 시작 변수(source variable), z_i를 종료 변수(destination variable)라 한다.

1. 첫 번째 변수를 제외한 모든 시작 변수는 이전 스텝의 종료 변수로 나타난다.
2. 한 변수는 하나 이상의 스텝에서 시작 변수로 나타날 수 있다.
3. 한 변수는 하나 이상의 스텝에서 종료 변수로 나타날 수 없다.

그리고 각 질의 조각 y_i r_i z_i에 대하여 노드 변수에서 데이타 그래프 DB로의 치환 함수 δ를 정의했을 때(즉, 함수 δ는 δ(u)=o 단, u∈nvar(q)이고, o∈O임), δ(y_i)와 δ(z_i) 사이에 정규 경로식 ri를 만족하는 경로가 데이타 그래프 DB에 존재한다.

예제 1 그림 1의 데이타 그래프에 대한 질의 q(p_2,p_3) :- p_1 (t) p_2, p_2 (c.d) p_3을 계산하면 (p_1,p_2)={(1,2), (1,3)}, (p_2,p_3)={(2,12),(3,12),(4,14)}이므로 질의의 결과는 (p_2,p_3)={(2,12), (3,12)}이 된다.

4. 뷰 정의와 질의 변환

4.1 뷰의 정의

본 논문에서 뷰는 질의와 마찬가지로 정의된다. 즉 v(**w**) :- $y_0r_0z_0$, ..., $y_{n-1}r_{n-1}z_{n-1}$의 형태로 정의된다. 질의 정의 시와 다른 점은 뷰의 헤드 변수의 집합 **w**가 고유한(distinct) nvar(v)={y_0, z_0, y_1, z_1, ..., z_{n-1}}를 모두 포함한다는 점이다. 이것은 뷰를 포함하는 질의를 처리할 때 질의의 각 조각에 대한 죠인 연산을 위해 필요하다.

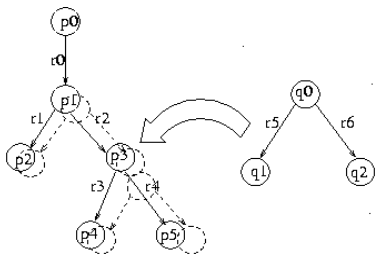

그림 2 질의 그래프와 뷰 그래프

4.2 질의 변환

우리가 다루는 질의 변환 문제는 다음과 같은 형태의 질의 q와 뷰의 집합 V={v_1,v_2,...,v_n}가 주어졌을 때 질의 q와 같은 결과를 내면서 하나 이상의 뷰를 이용하는 질의 q′을 찾는 문제이다.

1) 이것은 [12]의 분규 경로식(branching path expression)을 확장한 개념이다.

$q(\mathbf{u})$:- $p_0\ r_0\ p_1, p_1\ r_1\ p_2, p_1\ r_2\ p_3, p_3\ r_3\ p_4, p_3\ r_4\ p_5$
$v(\mathbf{w})$:- $q_0\ r_5\ q_1,\ q_0\ r_6\ q_2$ (1)

주어진 뷰 v와 질의 q에 대하여 뷰 v를 이루는 질의 조각(conjunct)을 $c^1_1, \ldots, c^1_m$이라 하고, 질의 q를 이루는 질의 조각을 $c^2_1, \ldots, c^2_n$라고 하자. 우리의 알고리즘은 먼저 다음 조건을 만족하는 심벌 매핑의 집합 Π를 찾는다. 즉, $\pi \in \Pi$는 모든 $c^1_i (1 \le i \le m)$에 대하여 $\pi(c^1_i) \equiv c^2_j (1 \le j \le n)$를 만족한다. 다음으로 $\pi \in \Pi$를 v에 적용시킨 v′을 이용하여 변환된 질의 q′을 생성한다.

4.2.1 심벌 매핑

질의 q가 n개의 질의 조각으로 이루어지고 뷰 v가 m개의 질의 조각으로 이루어졌다고 가정하면 가능한 총 매핑의 수는 n^m이다. 실제로 [11]의 매핑 알고리즘에서는 n^m의 매핑을 고려한다.

그리고, $\pi(c^1_i) \equiv c^2_j (1 \le j \le n, 1 \le i \le m)$을 만족하려면 c^1_i의 정규식을 r^1_i, c^2_j의 정규식을 r^2_j라고 했을 때, $L(r^1_i) = L(r^2_j)$를 만족하여야 하는데, 이 검사 문제는 PSPACE-complete[19] 문제이다. 본 논문에서는 이러한 잠재적인 매핑 알고리즘의 복잡도를 개선하는 다음과 같이 두 단계로 이루어진 매핑 알고리즘을 제안한다.

단계 1: 후보 매핑 찾기 먼저 질의 q와 뷰 v에 대하여 다음과 같이 질의 그래프 G_q와 G_v를 정의한다. D를 정규식을 나타내는 심벌의 집합이라 한다.

정의 4 질의 그래프 G_q=(V,E,R)는 루트 노드를 가지는 그래프로서 V∈nvar(q)인 노드의 집합, E⊆V x D x V은 방향성 있는 에지의 집합, R∈V는 루트 노드를 의미한다. 이때 각 질의 조각 a r b 에 대하여 d∈D를 정규식 r을 나타내는 심벌이라 하면 $a \xrightarrow{d} b$의 에지가 존재한다. G_v도 마찬가지로 정의된다.

예제 2 그림 2와 식 (1)의 질의와 뷰에 대한 G_q와G_v를 나타낸다. 단 r_i는 해당 정규식을 나타내는 심벌이라 가정한다.

질의 q와 뷰 v는 분규 정규 경로식의 성질을 만족하므로 그래프 G_q와 G_v는 사이클이 없는 트리이다. 이 성질로부터 우리는 후보 매핑(candidate mapping)의 수를 줄일 수 있다. 예를 들어 식 (1)의 뷰의 몸체로부터 질의의 몸체로의 매핑은 5^2개가 있지만 그림 2는 4개의 매핑만이 가능함을 보여준다[2)]

알고리즘 1은 주어진 그래프 G_q와 G_v로부터 후보 매핑(candidate mapping)을 찾는다.

단계 2: 후보 매핑의 타당성 검사 단계 1에서 구한 후보 매핑은 질의와 뷰의 구조적인 정보로부터 뽑아낸 단순한 심벌 매핑이므로 해당 정규식이 표현하는 언어가 서로 같은지를 검사하여야 한다. $L(r_1)=L(r_2)$을 검사하는 알고리즘의 복잡도가 크므로 단계 2-1에서 한번 더 매핑을 걸러낸다.

단계 2-1: 후보 매핑의 여과 그래프 G_q와 G_v로부터 $G2_q$ 와 $G2_v$를 다음과 같이 정의한다.

정의 5 그래프 $G2_q$는 그래프 G_q로부터 r_i로 레이블된 에지를 $L(A_i)=L(re(r_i))$[3)]을 만족하는 오토마타 A_i로 치환함으로써 생성된다. 특히, A_i는 에지의 시작을 시작 상태(start state)로 하고 에지의 종료 점을 종료 상태(final state)로 가진다.

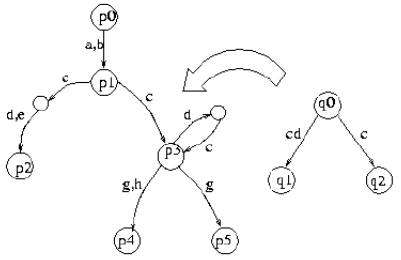

그림 3 $G2_q$ 그래프와 $G2_v$ 그래프

정의 6 그래프 $G2_v$는 그래프 G_v의 r_i로 레이블된 각 에지를 $L(re(r_i))$에 속하는 임의의 언어 하나로 치환한 그래프이다.

예제 3 그림 3은 r_0=(a|b), r_1=c(d|e), r_2=c(dc)*, r_3=g|h, r_4=g, r_5=cd|ce, r_6=(cd)*c 일 때, 그래프 G_q와 G_v로부터 얻어진 그래프 $G2_q$와 $G2_v$를 보인다.

알고리즘 2는 후보 매핑으로부터 여과된 후보 매핑(filtered candidate mapping)을 구하는 알고리즘을 나타낸다.

단계 2-2: 최종 매핑 찾기 단계 2-1에서 구한 여과된 후보 매핑은 최종 매핑(final mapping)의 필요조건이다. 그러므로 존재하는 $L(r_1)=L(r_2)$ 검사 알고리즘을 알고리즘 2에 첨가함으로써 최종 매핑을 얻는다.

예제 4 예제 3의 질의와 뷰에 대하여 후보 매핑은 {{$q_0 \to p_1$, $q_1 \to p_2$, $q_2 \to p_3$}, {$q_0 \to p_1$, $q_1 \to p_3$, $q_2 \to p_2$}, {$q_0 \to p_3$, $q_1 \to p_4$, $q_2 \to p_5$}, {$q_0 \to p_3$, $q_1 \to p_5$, q_2

2) 그림 2는 두개의 가능한 매핑을 보여주지만 자식의 매핑 순서를 바꿀 수 있으므로 네 개의 매핑이 존재한다.

3) re(r)는 r이 표현하는 정규식을 의미한다.

알고리즘 1 후보 매핑 찾기

```
1:  입력: 그래프 Gq, Gv
2:  출력: 후보 매핑
3:  Procedure find-candidate-mappings(Gq, Gv)
4:  큐 Q를 비어있는 상태로 초기화;
5:  rq ← Gq의 루트;
6:  rv ← Gv의 루트;
7:  rv를 가지고 rq를 방문하고 rq를 방문 표시;rq를 Q에 삽입;
8:  while Q가 비어있지 않음 do
9:        x ← Q.getFront();
10:     for x에 인접하고 방문 표시 안된 각 노드 w do
11:       if w's depth < rv's depth then
12:          continue;
13:       else if w의 자식 노드 수< rv의 자식 노드 수 then
14:          w를 방문 표시하고 Q에 삽입;
15:       else
16:          w를 방문 표시하고 Q에 삽입;
17:          루트 노드가 고정된 새로운 매핑 c를 생성;
18:          find-sub-mappings(w,rv,c);
19:       endif
20:     end for
21: end while
22:
23:  입력: 노드 q, v, 매핑 c
24:  출력: 후보 매핑
25:  Procedure find-sub-mappings(q,v,c)
26:  if v가 자식이 없음 then
27:       return;
28:  endif
29:  nq ← q의 자식 노드 수;
30:  nv ← v의 자식 노드 수;
31:  for 각 Permutation(nq,nv)개의 자식 매핑 do
32:     if 매핑 조건 만족 then
33:       c로 초기화된 새로운 매핑 d를 생성;
34:       현재 매핑을 d에 첨가;
35:       if d 채워짐 then
36:          d를 후보 매핑에 첨가;
37:       else
38:         for 각 자식 매핑 (qc,vc) do
39:            find-sub-mappings(qc,vc,d);
40:         end for
41:       end if
42:     end if
43:  end for
```

알고리즘 2 여과된 후보 매핑 찾기

```
1  입력: 그래프 G2q, G2v, 후보 매핑 c
2  출력: 여과된 후보 매핑
3  여과된 후보 매핑을 c로 초기화;
4  for 각 후보 매핑 do
5     for G2v 안의 각 에지 e do
6       if e의 레이블을 G2q의 해당 오토마타가 받아들이지 못함 then
7            그 매핑을 여과된 후보 매핑에서 삭제;
8            break;
9       end if
10    end for
11 end for
```

→ p_4}이고, 여과된 후보 매핑과 최종 매핑은 서로 같고 {{$q_0 \rightarrow p_1$, $q_1 \rightarrow p_2$, $q_2 \rightarrow p_3$}}이다.

4.2.2 질의 변환

Π를 이전 절에서 구한 최종 매핑(final mapping)이라 하자. 주어진 뷰 v(**u**,**w**) :- p(**u**,**w**)와 질의 q(**u**):- p′(**x**, **y**), s(**y**,**z**)에 대하여 π∈Π인 매핑을 v에 적용하여 v′(**x**,**y**) :- p(**x**,**y**)을 얻을 수 있다. 여기서 우리의 매핑 알고리즘에 의하여 p≡p′를 만족한다. 마지막으로 질의 q를 변환하여 q′(**u**):- v′(**x**,**y**), s(**y**,**z**)를 얻는다. 다음의 정리는 질의 q와 뷰 v가 주어졌을 때 심볼 매핑과 질의 변환 과정에 의하여 생성된 질의 q′의 결과가 같음을 보인다.

정리 1 질의 q에 대하여 이름 재 부여와 질의 변환 과정을 거쳐 생성된 질의 q′은 그 질의 결과가 같다.

증명: 각 질의 조각을 서술식(predicate)으로 보고 질의 변환 과정에서 p′(**x**,**y**)=$q_1'(x_1',...,x_i'),...,q_m'(y_1',...,y_j')$, s(**y**,**z**)=$r_1(v_1,...v_k),...,r_n(w_1,...w_l)$이라 가정한다. 그리고, 서술식 $q_1',...,q_m'$에 대응하는 릴레이션을 $Q_1',...,Q_m'$, $r_1,...,r_n$에 대응하는 릴레이션을 $R_1,...,R_n$이라 하자. 마지막으로 v′에서 p(**x**,**y**)=$q_1(x_1,...,x_i),...,$ $q_m(y_1,...,y_j)$이라 하고, $Q_1,...,Q_m$을 대응하는 릴레이션이라 하자. 그러면, 질의 q(**u**):- p′(**x**,**y**), s(**y**,**z**)의 결과는 $\pi_{\mathbf{u}}((Q_1'\bowtie...\bowtie Q_m')\bowtie(R_1\bowtie...\bowtie R_n))$가 된다. 반면에 질의 q′(**u**):- v′(**x**,**y**), s(**y**,**z**)의 결과는 $\pi_{\mathbf{u}}(\pi_{\mathbf{x,y}}(Q\bowtie_1...\bowtie Q_m)\bowtie(R_1\bowtie...\bowtie R_n))$이다. 여기서 우리의 매핑 알고리즘에 의하여 p≡p′이고, nvar(p)=**x**∪**y**이므로 $\pi_{\mathbf{x,y}}(Q_1'\bowtie...\bowtie Q_m')$= $Q_1'\bowtie...\bowtie Q_m'$이다. 따라서 q′의 결과는 q의 결과와 같다.

5. 질의 처리 기법

본 논문에서 다루는 다중 정규식을 처리하기 위해서는 다음과 같이 선형적으로 연결된 두개의 정규식을 처리하는 기법이 기반이 된다.

$$q(\mathbf{u}) :- p_0\ r_0\ p_1,\ p_1\ r_1\ p_2 \qquad (2)$$

이러한 질의를 처리하기 위해서는 정규식을 하나의 경로 레이블로 간주하면 일반적으로 다음과 같이 세 가지 방법을 생각할 수 있다[13].

1. 순방향 탐색(forward scan) 방식: 객체 p_0에 바인딩되는 객체에서 시작하여 정규식 r_0에 의하여 도달되는 객체를 p_1에 바인딩시킨 후 다시 이 객체들로부터 정규식 r_1으로 도달되는 객체를 p_2에 바인딩시키는 방식이다.
2. 역방향 탐색(backward scan) 방식: 순방향 탐색 방식의 반대 방향의 스캔 방식이다.
3. 복합(hybrid) 방식: 순방향 탐색 방식과 역방향 탐색 방식을 조합한 방식이다.

그렇지만 이러한 방식은 근원 그래프의 탐색을 기반으로 하므로 비용이 크고, 변환된 질의의 처리에 있어서는 질의에 뷰를 포함하므로 그대로 적용할 수 없다. 따라서 우리는 기존의 2-index[15] 방식이 임의의 정규식에 대하여 그 정규식을 만족하는 시작 객체(source object)와 종료 객체(destination object)를 효율적으로 찾는다는 점에 착안하여, 2-index 기법을 발전시킨 2^*-index 기법으로 각 정규식을 만족하는 노드 쌍을 찾은 후 결과를 조인하는 2^*-index-join 방법을 제안한다.

5.1 2-index의 복습

2-index는 select x_1, x_2 from * x_1 r x_2 형태의 질의를 효율적으로 처리하기 위한 인덱스이다. 여기서 r은 정규 경로식을 나타내고, x_1, x_2는 데이타 그래프의 노드에 바인딩되는 변수를 의미한다. 먼저 데이타 그래프 DB에서 임의의 두 노드에 대하여 다음과 같이 $L_{(v,u)}$(DB)를 정의한다.

정의 7 $L_{(v,u)}$(DB)={w|w=a_1, ..., a_n, 그리고 DB에 경로 $v \xrightarrow{a_1} \cdots \xrightarrow{a_n} u$가 존재}

그리고 노드의 각 쌍간의 동치 관계(equivalent relation) ≡를 다음과 같이 정의한다.

정의 8 (v,u)≡(v′,u′) ⇔ $L_{(v,u)}=L_{(v',u')}$

이때 관계 연산 ≡의 계산 복잡도가 크지만 계산 복잡도가 낮은 시뮬레이션(simulation), 바이시뮬레이션(bisimulation)을 이용한 관계 연산 ≈가 존재하여 다음 식을 만족한다.

$$(v,u)\approx(v',u') \Rightarrow (v,u)\equiv(v',u')$$

즉, 관계 연산 ≈의 계산으로 ≡의 효과를 얻을 수 있다. 마지막으로 2-index는 루트를 가진 그래프로 다음과 같이 정의된다. 단, [(v,u)]는 (v,u)의 동치 클래스(equivalent class)를 의미한다.

정의 9 2-index를 I^2(DB)라고 하면 I^2(DB)는 다음과 같이 구성된다. 먼저 노드는 ≈에 대한 동치 클래스(equivalent class)로 구성되고, 루트 노드는 [(x,x)] 형태의 노드이다. 에지는 DB의 각 노드 v와 각 에지 $u \xrightarrow{a} u'$에 대하여 [(v,u)] $\xrightarrow{a}$ [(v,u′)]로 구성된다.

이때, $L_{(v,u)}$(DB)=$L_{[(v,u)]}$(I^2(DB))가 되므로[4] 질의 select x,y from *x r y를 처리하기 위해서는 I^2에서 r y를 계산한 후 노드의 익스텐트의 합집합을 구하면 된다.

5.2 2^*-index

비정형 데이타에 대한 질의에서 많이 나타나면서 질의 처리기의 성능을 저하시키는 질의가 질의에 _* 연산

4) 증명은 [15]를 참조.

이 존재하는 질의이다. 예를 들어 정규 경로식 person._*.name과 같은 질의를 처리한다고 하면 _*가 임의의 경로에 매칭 되므로 name 레이블을 찾기 위하여 거의 전 그래프 영역을 탐색하여야 한다. 2-index 방법에서도 _* 연산이 질의에 포함되어 있으면 I^2 그래프 전체를 탐색하여야 한다.

2^*-index 기법은 XML의 DTD로부터 각 요소로부터 파생될 경로식을 구하여 _* 연산을 실제 경로식으로 치환하는 방법이다. 먼저 DTD를 (k, n, p)의 집합으로 추상화한다. 여기서 k는 요소 혹은 애트리뷰트를 나타내고, L을 요소 이름의 집합이라고 했을 때, n∈L이고, k=e 즉, 요소일 때 p는 L에 대한 정규식 또는 문자열을 나타내는 PCDATA이고, k=a 즉, 애트리뷰트일 때 p는 애트리뷰트 이름이다. 그리고 다음과 같이 DTD 그래프 G_D를 정의한다. 단, N은 DTD에 존재하는 요소와 애트리뷰트 이름의 집합이다.

정의 10 DTD 그래프 G_D=(V,E)는 노드와 방향성 에지를 가지는 그래프로 V∈N으로 각 요소는 그래프 상에 한번씩만 나타나고, 애트리뷰트는 DTD에 나타나는 만큼 나타난다. 에지 E⊆V x V으로 각 (k, n, p)에 대하여 k=e일때, 노드 n 에서 p에 존재하는 각 요소 이름 m에 대하여 n→m의 에지가 존재하고, k=a일 때는 노드 n에서 p의 애트리뷰트 이름 a로의 에지 n→a가 존재한다.

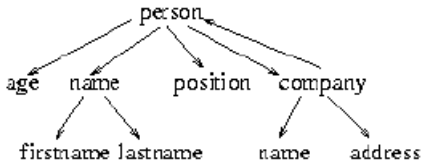

그림 4 DTD 그래프 예제

예제 5 DTD={(e,person,(name,position?,company)), (e,company, (name,address,person+)), (e,name,(firstname?, lastname)), (e,position,PCDATA), (e,address,PCDATA), (a,person, age), (a,company, name)}라고 하면 이에 대한 DTD 그래프 G_D는 그림 4와 같다.

주어진 그래프 G_D와 _* 연산을 포함한 질의가 주어졌을 때 알고리즘 3은 _* 연산이 없는 질의로 바꾸어준다.

예제 6 예제 3의 DTD에 대하여 질의 person._*.name은 알고리즘 3에 의하여 (person.name|person.company.name)의 질의로 변환된다.

알고리즘 3 '*' 연산의 제거

```
1   입력: 그래프 G_D, 질의 r=r1._*.r2
2   출력: _* 연산이 없는 질의
3   Procedure Star-Flattening(G_D, q)
4   if r1= ϕ then
5      for DTD상의 각 요소(element) e do
6        Star-Flattening(G_D,e._*.r2);
7      end for
8   else
9     head ← r1의 마지막 레이블;
10    tail ← r2의 첫째 레이블;
11    큐 Q를 비어있는 상태로 초기화;
12    head에 방문 표시하고 Q에 삽입;
13    while Q가 비어있지 않음 do
14       x←GetFront(Q);
15       for x에 인접한 방문 표시 안된 노드, 사이클을 만들지 않는 방문된 노드 w do
16         if w=tail then
17            head에서 tail까지의 경로를 출력;
18            w에 방문 표시;
19         else
20            w를 방문 표시하고 Q에 삽입;
21         end if
22       end for
23    end while
24  end if
```

5.3 2^*-index-join 알고리즘

2^*-index-join 방법은 다음과 같은 형태의 질의를 처리하는 기법이다.

$$q'(\mathbf{u}) :- w(\mathbf{x},\mathbf{y}),\ c(\mathbf{y},\mathbf{z}) \quad (3)$$

여기서 w(**x**,**y**)는 질의 조각 중 뷰로 치환되는 식을 나타내고 c(**y**,**z**)는 뷰로 치환되지 않은 질의 조각을 나타낸다. 완전 변환(complete rewriting)[4]이 존재하는 경우는 c(**y**,**z**)= ϕ인 경우이다. w(x,y)= ϕ인 경우는 뷰에서 질의로의 매핑이 없는 경우이다. 이때, w(**x**,**y**)에 참가하는 뷰를 $v_{i_1}, \ldots v_{i_l}$라고 하고, 대응하는 릴레이션을 $W_1,\ldots,W_l$이라 하자.

그리고 c(**y**,**z**)=$y_0'r_0'z_0'$, ...,y_{k-1}' $r_{k-1}'z_{k-1}'$이라 하고, 각 질의 조각을 2^*-index 방법으로 r_i에 매칭되는 객체의 짝으로 이루어지는 릴레이션을 $Q_0,\ldots,Q_{k-1}$이라 하면 2^*-index-join 알고리즘은 $\pi_{\mathbf{u}}(W_1 \bowtie \ldots \bowtie W_l \bowtie Q_0 \ldots \bowtie Q_{k-1})$를 리턴 한다.

5.4 평가

여기서는 앞서 언급한 다중 정규 경로식의 처리에 있어서 일반적으로 생각할 수 있는 세 가지 방법 즉, 1. 순방향 탐색 방법 2. 역방향 탐색 방법 3. 복합 방법 중에서 순방향 탐색 방법과 2^*-index-join 방법을 비용 모델을 통하여 비교한다[5].

질의 처리를 위한 주요 인자로는 객체 탐색 수를 사용한다. 왜냐하면 비정형 데이타 모델에서는 객체간의 클러스터링 정보를 시스템이 미리 제공할 수 없기 때문이다. 즉, 한 객체의 탐색이 한 페이지 탐색에 대응된다고 가정한다.

비용 함수의 인자는 [20]과 같은데 OIDJoin이 추가되었다.

- Fanout(x,l): 변수 x에 바인드되는 객체의 집합에서 레이블 l로 도달되는 평균 객체 수이다. 예를 들어, 그림 1에서 객체 2,3이 변수 z에 바인딩되었다고 가정하면, Fanout(z,b)=1이다.
- |x|: x에 바인드되는 객체 수이다. 위의 예에서 |z|=2이다.
- cost(x,l,y): x에 바인드되는 객체의 집합에서 레이블 l로 도달되는 객체 y의 집합을 얻기 위한 객체 탐색 수로 |x|*Fanout(x,l)로 계산된다. 위의 예에서, 단일 경로식(simple path expression)[6] z b w를 계산하기 위한 비용은 cost(z,b,w)=|z|*Fanout(z,b)=2*1=2이다.
- OIDJoin: 2^*-index-join 방법에서 각 조각의 결과 집합에 대한 죠인 비용이다.

여기서는 식 (3)에서 w(**x**,**y**)= ϕ[7]이고 c(**y**,**z**)가 식(2)의 형태 일 때 순방향 탐색 방법과 2^*-index-join의 비용을 비교한다. 그리고 r_0=a 이고 r_1=b라고 가정한다.[8] 다음은 순방향 탐색 방법과 2^*-index-join 방법의 비용을 계산한 식이다. 여기서 $|p_i^1|$을 근원 데이타베이스(source database)에서 변수 p_i에 바인딩된 객체 수이고, $|p_i^2|$는 I^2 그래프에서 바인딩된 객체 수이다. 마찬가지로 $Fanout^1(p_i, l)$은 근원 데이타베이스에서 변수 p_i에 바인딩된 객체로부터 레이블 l로 도달되는 평균 객체 수이고, $Fanout^2(p_i, l)$는 I^2 그래프에서의 그것이다.

$$\begin{aligned}
&\text{Cost}_{\text{순방향 탐색 방법}} \\
&= |p_0^1| \, Fanout^1(p_0, a) + |p_1^1| \, Fanout^1(p_1, b) \\
&= |p_0^1| \, Fanout^1(p_0, a) \\
&\quad + |p_0^1| \, Fanout^1(p_0, a) \, Fanout^1(p_1, b) \\
&\text{Cost}_{2^*\text{-index-join}} \\
&= |p_0^2| \, Fanout^2(p_0, a) + |p_1^2| \, Fanout^2(p_1, b) \\
&\quad + OIDJoin
\end{aligned}$$

I^2 그래프에서 $|p_0^2|=|p_1^2|$이고, 이것은 루트의 개수로 I^2 그래프에서 루트 노드는 [(x,x)] 형태의 동치 클래스(equivalent class)이다. 대부분의 경우 이 루트의 개수는 매우 작다. 특히, 사이클이 없는 데이타베이스의 경우에는 I^2는 단일 루트를 가진다.

또한 $Fanout^1(p_i, l) \gg Fanout^2(p_i, l)$이다. 그리고 OIDJoin의 비용은 객체 식별자간의 죠인 비용이므로 매우 작은 값이다. 따라서 Cost(순방향 탐색 방법) ≫ Cost (2^*-index-join)이다.

6. 결 론

본 논문에서는 XML 데이타가 매핑될 그래프 기반의 비정형 데이타 모델 하에서 다중 정규식을 가지는 질의에 대한 효율적인 처리 기법을 제안하였다. 우리의 알고리즘은 1. 질의 변환 단계와 2. 질의 처리 단계로 이루어진다. 질의 변환 단계에서는 뷰의 몸체(body)에서 질의의 몸체로의 심벌 매핑을 질의의 구조적인 정보를 이용하여 그 복잡도를 떨어뜨리는 기법을 제안하고, 그 알고리즘의 안전성(soundness)을 보였다. 질의 처리 단계에서는 각 질의 조각에 대하여 특정 정규식을 만족하는 두 노드의 짝을 쉽게 찾는 2-index 방법을 확장한 2^*-index 방법을 이용한다. 2^*-index 방법은 XML의 DTD 정보를 이용하여 _* 연산이 들어간 질의를 효율

5) 데이타베이스의 통계정보에 따라 2,3의 방법이 우수할 수 있지만 일반적인 경우 1,2,3 방법은 비슷한 성능을 가진다고 가정한다.

6) [13]의 정의 3.1에서 정의된 표현으로 데이타베이스에서 단일 스텝의 이동을 의미한다.

7) 이 경우에 상대적으로 순방향 탐색 방법이 2^*-index-join 방법에 비하여 유리한 경우이다.

8) 정규식이 복잡할 수록 2^*-index-join이 유리하므로 이 경우도 순방향 탐색 방법이 유리한 경우이다.

적으로 처리한다. 우리의 기법이 기존의 일반적 질의 처리 기법에 비하여 우수함을 보였다.

참 고 문 헌

[1] Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Moshe Y. Vardi, "Rewriting of regular expressions and regular path queries," Proceedings of the 18th ACM SIGACT SIGMOD SIGART Sym. on Prin- ciples of Database Systems, 1999.

[2] Giosta Grahne and Alex Thomo, "An optimization technique for answering regular path queries," International Workshop on the Web and Databases, 2000.

[3] Roy Goldman and Jennifer Widom, "DataGuides: enabling query formulation and optimization in semistructured databases," Proceedings of the Conference on Very Large Data Bases, 1997.

[4] Alon Y. Levy, Alberto O. Mendelzon, Yehoshua Sagiv, and Divesh Srivastava, "Answering queries using views," Proceedings of ACM Symposium on Principles of Database Systems, 1995.

[5] Bruce G. Lindsay, Laura M. Haas, C. Mohan, Hamid Pirahesh, and Paul F. Wilms, "Efficiently updating materialized views," Proceedings of the ACM SIGMOD International Conference on the Management of Data, 1986.

[6] P. Larson and H. Yang, "Computing queries from derived relations," Proceedings of the Conference on Very Large Data Bases, 1985.

[7] Surajit Chaudhuri, Ravi Krishnamurthy, Spyros Potamianos, and Kyuseok Shim, "Optimizing queries with materialized views," Proceedings of International Conference on Data Engineering, 1995.

[8] Elke A. Rundensteiner, "Multiview: a metho- dology for supporting multiple views in object- oriented databases," Proceedings of the Confe- rence on Very Large Data Bases, 1992.

[9] Serge Abiteboul and Anthony J. Bonner, "Objects and views," Proceedings of the ACM SIGMOD International Conference on the Management of Data, 1991.

[10] Dana Florescu, Alon Levy, and Dan Suciu, "Query containment for conjunctive queries with regular expressions," Proceedings of ACM Symposium on Principles of Database Systems, 1998.

[11] Yannis Papakonstantinou, and Vasilis A. Vassalos, "Query rewriting using semistru- ctured views," Proceedings of the ACM SIGMOD International Conference on the Management of Data, 1999.

[12] J. McHugh and J. Widom, "Optimizing bran- ching path expressions," Technical report, Stanford University Database Group, 1999.

[13] J. McHugh and J. Widom, "Query optimization for XML," Proceedings of the Conference on Very Large Data Bases, 1999.

[14] Mary Fernandez and Dan Suciu, "Optimizing regular path expressions using graph schemas," Proceedings of International Conference on Data Engineering, 1998.

[15] Tova Milo and Dan Suciu, "Index structures for path expressions," Proceedings of the International Conference on Database Theory, 1999.

[16] Peter Buneman, Susan Davidson, Gerd Hille- brand, and Dan Suciu, "A query language and optimization techniques for unstructured data," Proceedings of the ACM SIGMOD Interna- tional Conference on the Management of Data, 1996.

[17] S. Abiteboul, Dallan Quass, Jason McHugh, Jennifer Widom, Janet Wiener, "The lorel query language for semistructured data," International Journal on Digital Libraries, 1996.

[18] A. Deutsch, M. Fernandez, D. Florescu, A. Levy, and D. Suciu, "Query language for XML," Proceedings of Eighth International World Wide Web Conference, 1999.

[19] L.J. Stockmeyer and A.R. Meyer, "Word problems requiring exponential time," ACM Symposium on Theory of Computing, 1973.

[20] S. Abiteboul, J. McHugh, M. Rys, V. Vassalos, and J. Weiner, "Incremental maintenance for materialized views over semistructured data," Proceedings of the Conference on Very Large Data Bases, 1998.

정 태 선
정보과학회논문지 : 데이타베이스
제 28 권 제 2 호 참조

김 형 주
정보과학회논문지 : 데이타베이스
제 28 권 제 1 호 참조