객체 데이터베이스를 이용한 바이오 XML 저장시스템

김태경*, 이경희**, 임정곤***, 정태성*, 조완섭***
*충북대학교 정보산업공학과, **충북대학교 전자계산학과, ***충북대학교 경영정보학과
misoh049@hanmail.net, {khlee, tinnom}@cbnu.ac.kr, mispro97@naver.com, wscho@cbnu.ac.kr

The Bio-XML Storage System Using Object Database Systems

Taekyung Kim* ,Kyunghee Lee**, JoungKon Lim***, Taesung Jung*, Wansup Cho***
*Dept. of Information Industrial Engineering,
**Dept. of Computer Science
***Dept. of Management Information System
Chungbuk National University

요 약

본 논문은 객체 데이터베이스 속성을 적용하여 데이터베이스 스키마를 생성하고 XML 문서를 저장하는 기법을 제안한다. 기존의 관계형 데이터베이스는 트리 기반의 XML 문서를 플랫한 테이블에 저장하므로 모델 불일치 문제가 발생한다. 또한, 문서를 검색할 때 고비용의 조인 연산이 필요하다. 하지만 객체 데이터베이스의 집합값 속성과 객체참조 속성은 트리 기반의 XML 문서를 저장할 때 모델 측면에서 자연스럽다. 집합값 속성과 객체참조 속성은 XML 질의에 자주 사용되는 경로질의 및 순서를 이용하는 질의를 처리할 때에도 유리하다. 본 논문에서는 객체 데이터베이스의 집합값 속성과 객체참조 속성을 이용하여 XML 문서를 저장하기 위한 2가지의 DTD-의존적 스키마 설계 기법인 i) 기본 규칙, ii) 인라인 규칙을 제시한다. 다양한 XML 문서에 대해 각각의 규칙에 따른 클래스 수, 저장 공간, 그리고 질의처리 시간을 비교 분석하였다.

1. 서론

XML은 정보의 표현 및 교환의 표준 포맷으로 W3C에서 제안하였다. 현재 전자상거래 분야의 ebXML, 공간 정보 관리 분야의 GML, Semantic Web의 RDF, 그리고 생명정보학의 SBML과 같은 다양한 응용 분야에서 XML 기반의 데이터 표현 명세서들을 제안하고 있다. 이러한 문서에 대한 검색을 지원하기 위하여 XPath[2], XQuery[10]와 같은 XML 질의어 들이 있다.

일반적으로 XML을 저장하는 방법은 XML 질의 수행에 가장 큰 영향을 준다. 그러므로 현재 XML 데이터를 효과적으로 저장함으로써 검색 성능을 향상하는 것이 주된 화두이다. 대부분의 연구는 관계형 데이터베이스, XML 전용 데이터베이스, 그리고 객체 데이터베이스에 XML 데이터를 저장하는 것이다.

관계형 데이터베이스의 경우 뛰어난 성능과 안정된 관리 시스템을 XML 저장에 사용할 수 있으므로 다수의 관련연구가 제시되었다. XML 전용 데이터베이스는 XML 데이터를 저장하는데 있어서 모델의 불일치 문제가 없기 때문에 이론적으로 바람직한 방법이다. 객체 데이터베이스는 XML과 모델 측면에서 유사도가 높으므로 저장이 쉽고 자연스럽다. 그러나 연구[8]에서는 객체 데이터베이스의 고유 성질을 충분히 반영하지 못하였다.

본 논문에서는 객체 데이터베이스의 성격을 충분히 활용하여 DTD-의존적 XML 저장 스키마를 생성하고 저장하는 기법을 제안한다. 다중 값 속성과 객체 참조 속성은 객체 데이터베이스의 고유 특징으로 트리 기반의 XML 문서를 저장하는데 유리한 조건이다. 또한 저장공간 측면과 XML 질의에 자주 사용하는 경로질의 및 순서를 이용한 질의 처리에도 뛰어나다.

본 연구는 한국과학재단 특정기초 연구사업(R01-2003-000-11723-0)으로부터 지원을 받았음

본 논문의 구성은 다음과 같다. 2절에서는 XML 저장시스템에 대한 관련 연구를 살펴보고, 3절에서는 XML 데이터 저장을 위한 스키마 설계 기법을 살펴본다. 4절에서는 각각의 저장 기법에 대한 실험 및 평가를 하고, 5절에서는 본 논문의 결론을 맺고자 한다.

2. 관련 연구

기존의 XML 저장 관리 시스템은 기반 시스템의 특징에 따라 분류할 수가 있는데, 관계형 데이터베이스 기반 시스템, 객체 데이터베이스 기반 시스템, XML 전용 데이터베이스가 있다.

2.1 관계형 데이터베이스

관계형 데이터베이스를 이용하여 XML 문서를 저장하는 방법에는 크게 DTD-의존적인 방식과 DTD-독립적인 방식이 있다. DTD-의존적인 방식은 그래프 형태의 DTD를 기반으로 하여 관계형 데이터베이스 스키마를 생성하고, 이 DTD를 만족하는 XML 문서를 파싱하여 생성한 관계형 데이터베이스의 테이블에 저장한다. 연구[7]에서는 DTD를 기반으로 하는 공유 인라인 방식과 복합 인라인방식을 제안하였다. DTD-독립적 방식은 각 문서의 DTD와 별개의 관계 데이터베이스 스키마를 생성하고 XML 문서를 저장하는 방식으로 간선(edge)을 기반으로 하는 Edge방식[3], Monet 방식[1]이 있다. 관계형 데이터베이스의 경우 자체의 뛰어난 성능을 이용할 수 있다. 또한 동시성 제어, 보안, 다양한 데이터베이스 관리시스템을 그대로 이용할 수 있다는 장점이 있다. 하지만 트리 기반의 XML 문서를 정형화된 테이블로 저장하는데 생기는 모델간의 불일치로 인한 문제점과 대용량을 다루는 XML 데이터에 대한 질의처리시 조인의 수의 증가로 성능이 저하되는 문제점이 있다[7].

2.2 XML 전용 데이터베이스

XML 전용 데이터베이스는 반 구조적(semi-structured)인 데이터를 저장하기 위해서 개발된 시스템이다. 대표적인 예로 Timber[4], Tamino[5]가 있다. 이러한 시스템들은 XML 문서 전체를 파일 시템으로 저장하면서 문서에 대한 구조 정보와 내용 정보에 대한 인덱스를 생성하고, 질의시 이 인덱스를 이용한다. 이론적으로는 XML 전용 관리 시스템을 이용하여 XML 데이터를 저장, 관리하는 것이 가장 효율적이다. 그러나 기존의 구조적 데이터와의 통합문제, 다중 사용자 지원 및 대용량 데이터 처리에 대한 검증이 미흡하다.

2.3 객체 데이터베이스

객체 데이터베이스는 모델 측면에서 트리 구조의 XML과 유사하다. 그러므로 DTD로부터 객체 데이터베이스 스키마를 생성하는 문제가 간단하다. 다중 값 속성이 지원되며, sequence와 같은 타입을 지원하므로 XML의 엘리먼트 순서 정보를 자연스럽게 유지한다. 또한, XML 질의에 사용되는 경로식은 조인이 아닌 객체의 포인터를 따라 검색하므로 질의 처리 성능이 뛰어나다. 본 논문에서 제안하는 XML 저장 스키마 설계기법의 기본 아이디어는 연구[7]와 유사하지만 객체 데이터베이스의 다중 값 속성과 객체 참조 속성을 이용한 것이 차이점이다. 또한 기존의 객체 데이터베이스를 이용한 연구[8]는 다중 값 속성과 객체 참조 속성을 반영하지 않았다.

3. XML 저장을 위한 객체 스키마 생성 기법

이 절에서는 제안하는 객체 데이터베이스의 속성을 이용한 XML 저장 구조의 생성방법을 설명한다. 본 논문에서 제안하는 방식은 기본 규칙과 인라인 규칙을 설명한다.

3.1 용어 정의 및 예제 DTD

제안하는 스키마 생성 기법을 설명하는데 사용할 예제 DTD와 필요한 용어를 정의한다.

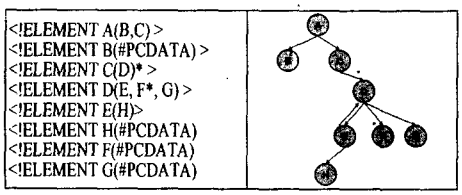

그림 1. 예제 DTD와 DTD 그래프

그림 1은 예제 DTD와 DTD의 구성요소인 엘리먼트를 노드로 표현하고, 노드사이의 관계를 간선으로 매핑하여 표현한 것이다. 제안하는 스키마 생성은 주어진 DTD로부터 이 구조를 표현할 수 있는 객체 데이터베이스 스키마를 생성하는 것이다.

e는 DTD 상에 존재하는 엘리먼트를 의미하며, E는 엘리먼트의 집합을 나타낸다. 그리고, s_e는 엘리먼트 e의 하위엘리먼트를 의미하며, S는 엘리먼트 e의 하위 엘리먼트 집합을 뜻한다. 그리고 A는 엘리먼트 e의 애트리뷰트 집합이며, *sequenceType*은 다중값 속성을 의미하는 ?, *, + 연산자를 가지는 엘리먼트들의 집합이다.

3.2 기본 규칙

기본 규칙은 DTD에 가장 충실한 객체 데이터베이스 스키마 생성 방법으로써, 그림 2와 같은 변환 규칙을 가진다.

For each $e \in E, \#(S) \geq 1$ OR $\#(A) \geq 1 \rightarrow$ define_Class(e)
For each $s_e \in S \rightarrow$ Add_attributes_of_Class(e)
$s_e \in$ SequenceType $\rightarrow$ Define_multivalued_att(s_e,e)

그림2 기본 규칙

각각의 DTD 엘리먼트에 대하여 하위 엘리먼트가 하나라도 존재하거나 또는 애트리뷰트가 적어도 한 개 이상 존재한다면 클래스로 정의한다. 또한 하위 엘리먼트 또는 애트리뷰트가 '?', '+', '*'의 다중값 속성을 가지면, 셋 값 애트리뷰트로 정의한다. 그림 1에 있는 DTD에 기본 규칙을 적용하면 그림 3과 같은 객체 데이터베이스 스키마 그래프가 생성된다.

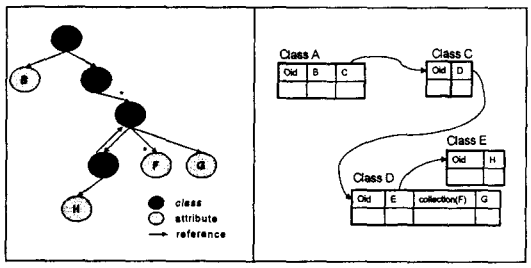

그림 3. 객체데이터베이스 스키마 그래프(a) 및 스키마 구조(b)

엘리먼트 A, C, D, E는 하위 엘리먼트를 가지므로 클래스로 변환되고 클래스에 해당하는 엘리먼트의 하위 엘리먼트들은 클래스의 애트리뷰트로 표현된다. 또한 엘리먼트간의 상하관계를 표현하기 위하여 속성-도메인 관계로 설정한다. 관계형 데이터베이스의 경우 관계를 설정하기 위해 주키-외래키를 사용한다. 그림 3(a)의 스키마 그래프를 객체데이터베이스 스키마로 변환하면 그림3(b)와 같다. 기본 규칙을 적용하는 경우 DTD의 모델과 객체 데이터베이스 스키마 간의 모델이 일치하므로 XML 질의를 객체 데이터베이스 질의로 변환시 용이하다.

3.인라인 규칙

인라인 규칙은 질의 성능 향상을 목적으로 생략 가능한 클래스를 제거함으로써 스키마를 간략화 하는 규칙이다. 그림 3의 객체 데이터베이스 스키마에서 클래스 C는 단순히 클래스 A와 D사이에서 경로를 제공해 주는 역할만 한다. 이러한 클래스들의 속성을 상위 클래스의 속성으로 인라인 시켜서 클래스의 수를 줄인다. 또한, 경로식의 길이를 줄임으로써 질의처리 성능을 높이는 것이 주요 아이디어이다. 다음 그림 4는 인라인 기법의 변환 규칙이다.

For each e , #(S) = 1 and se $\in$ SequenceType
$\rightarrow$ Add_Multi-Valued_attribute_of_Parent-Class(e)

그림 4. 인라인 규칙

이 규칙에 의하면, 각 엘리먼트에 대해서, 한 개의 하위 엘리먼트를 가지고, 그 하위 엘리먼트가 다중값 속성일 때, 이 속성을 상위 클래스의 속성으로 인라인 한다. 그림 1의 DTD 그래프에서 엘리

먼트 C는 오직 하나의 하위 엘리먼트 D를 가지고, 이 엘리먼트 D는 다중값 속성을 가지므로 그림 4의 규칙에 의해 그림 5의 그림과 같이 상위 엘리먼트의 속성으로 인라인 한다.

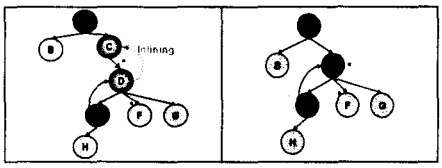

그림 5. 인라인 규칙 적용한 스키마 그래프

그림 5에서와 같이 인라인 규칙을 적용하면 기본 규칙을 적용한 데이터베이스 스키마 구조에 비하여 SequenceType의 개수만큼의 클래스가 줄어드는 효과를 얻는다.

4. 실험 및 평가

본 논문에 제안된 DTD를 객체 데이터베이스 스키마로 변환하기 위한 두 가지 기법에 대하여 저장 비용 및 질의 비용을 평가 한다. 실험에 사용된 DTD 및 XML 데이터는 GBSeqXML[11], MBench XML data[12], SigmodRecord[13] 이다.

4.1 공간 복잡도

<표 1>은 세 가지의 XML 문서에 대하여 문서의 실제 크기, 생성된 클래스 수, 그리고 저장된 뒤의 데이터베이스 크기 및 레코드 개수를 보여준다.

<표 1> XML 문서에 대한 공간복잡도

	size (MB)	기본 규칙			인라인 규칙		
		클래스 개수	DB size(MB)	레코드 개수	클래스 개수	DB size(MB)	레코드 개수
DBLP	0.4	6	0.78	6891	4	0.63	5309
MBench	42.8	2	48.3	67696	2	48.3	67696
GBSeq	80	22	71.7	528104	10	61.7	424532

<표 1>은 동일한 문서에 대하여 기본 규칙과 인라인 규칙을 각각 적용할 때 클래스 수, 데이터베이스 사이즈, 그리고 레코드 개수에서 인라인 방식이 더욱 효율적임을 보여준다.

4.2 시간 복잡도

기본 규칙과 인라인 규칙으로 생성한 데이터베이스에 대하여 질의 처리 시간을 비교한다. 실험에 사용한 질의는 XML 질의에 자주 사용되는 경로질의이다. 이 실험은 GBSeq XML 1GB의 문서를 대상으로 한 경로 질의의 수행시간을 결과 선택율에 따라 비교하는 것이다.

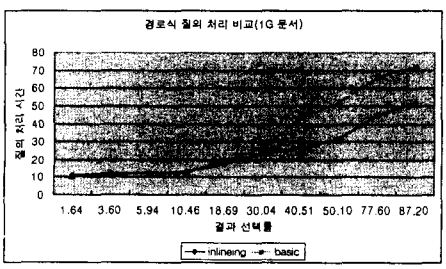

그림 6. 기본 방식과 인라인 방식에 대한 질의 처리 시간

그림 6에서 보는 바와 같이 결과 선택률이 증가할 수 록 인라인 규칙을 적용한 스키마에 따라 저장한 경우의 성능이 기본 규칙을 적용한 스키마의 경우보다 질의처리 시간이 빠른 것을 알 수 있다. 이것은 인라인 규칙을 적용한 스키마는 기본 규칙을 적용한 스키마 보다 클래스의 개수가 더 작기 때문에 경로탐색에 소요되는 비용이 줄어들기 때문이다.

5. 결론

본 논문은 객체 데이터베이스의 고유 속성인 객체 참조와 다중값 속성을 이용하여 XML 데이터를 저장하기 위한 스키마 생성 방법인 기본 규칙과 인라인 규칙을 제안하였다. 인라인 규칙은 가장 DTD에 충실한 스키마 생성 방법이고, 인라인 방법은 기본 규칙을 통해 생성된 스키마에서 의미 손상 없이 경로를 좀 더 줄이기 위해 제안한 방법이다. 실험 결과는 인라인 방법이 기본 방법보다 공간 복잡도나 질의 처리에서 더욱 효율적임을 보여준다. 현재 관계형 데이터베이스 벤더들이 객체 데이터베이스의 속성들을 도입하고 있다. 이 시점에서 본 논문에서 제안한 방법은 데이터베이스 벤더들이 XML 데이터를 테이블에 한정시켜 저장하는 상황에서 더욱 확장할 수 있는 방안을 제시해 준다. 향후 연구로는 XQuery를 객체 데이터베이스 질의로 변환을 위한 질의 변환 처리기의 구현과 질의 결과를 XML 변환하는 연구가 필요하다.

6. 참고문헌

[1] A. Schmidt, M. L.. Kersten, M. Windhouwer, and F. Waas, "Efficient relational storage and retrieval of XML documents," In proceedings of WebDB, 2000

[2] Clark and S. DeRose, "XML path language(XPath) 1.0," W3C Recommendation, http://www.w3c.org/TR/xpath, 1999.

[3] D. Florescu and D. Kossmann. Storing and Querying XML Data using an RDBMS. Bulletin of the IEEE Computer Society Technical Committee on Data Engineering. 22(3):27-34, September 1999.

[4] H. V. Jagadish, Shurug Al-Khalifa, Adriane Chapman, Laks V. S. Lakshmanan, Andrew Nierman, Stelios Paparizos, Jignesh M, Patel, Divesh Srivastava, Nuwee, "TIMBER: A Native XML Database," ACM SIGMODE Cof. 1988

[5] H. Schoning, "Tamino- a DBMS Designed for XML," in Proceedings of IEEE ICDE, 2001.

[6] I. Tatarinov and S. D. Viglas, "Storing and Querying Ordered XML Using a Relational Database System", In *Proc. Intl. Conf. on Management of Data, ACM SIGMOD*, 2002.

[7] J. Shanmugasundaram, K. Tufte, G. He, C. Zhang, D. DeWitt, and J. Naughton. "Relational databases for querying xml documents: Limitations and opportunities," In *Proc. Intl. Conf. on 25th VLDB*, 1999.

[8] K. Runapongsa and J. M. Patel, "Storing and Querying XML Data in Object-Relational DBMSs," *ACM SIGMOD Record*, **31**(1), 2002.

[9] Shimura, Yoshikawa and Uemura. "Storage and Retrieval of XML Documents Using Object-Relational Databases", pg.206--217, in DEXA, 1999.

[10] S. Boag, et al., "XQuery 1.0: An XML Query Language," W3C Working Draft, http://www.w3.org/TR/xquery/ 2003.

[11] NCBI GenBank, http://www.ncbi.nlm.nih.gov/

[12] MBench, http://www.eecs.umich.edu/db/mbench/

[13] SigmodRecordhttp://www.dia.uniroma3.it/Araneus/Sigmod