Bio-XML 관리를 위한 DTD 의존적 객체 데이터베이스 스키마 설계기법

DTD-dependent object database schema design methods for efficiently managing Bio-XML

김태경, 이경희*, 조완섭**
충북대학교 정보산업공학과, 충북대학교 전자계산학과*, 충북대학교 경영정보학과**

Kim Tae-Kyung, Lee Kyung-Hee*, Cho Wan-Sup**
Dept. of Information Industrial Engineering,
Dept. of Computer Science*,
Dept. of Management Information System
Chungbuk National University**

요약

본 논문에서는 Bio-XML 문서를 효율적으로 객체 데이터베이스에 저장하고, XML 질의에서 주로 사용되는 경로식을 효과적으로 처리할 수 있는 DTD 의존적인 객체 데이터베이스 스키마 설계기법을 제안한다. XML DTD와 객체데이터베이스의 스키마는 구조적으로 비슷하고, 객체 데이터베이스의 고유 특성인 객체 참조와 집합값 속성은 XML 데이터를 저장하는데 유리하다. 본 논문에서는 객체 데이터베이스의 고유 특성을 충분히 반영하여 두 가지의 스키마 설계기법인 기본적 방법과 인라인 방법을 제안한다. 뿐만 아니라, 각각 설계 기법에 대하여 시스템 성능 평가를 수행하였으며, 설계 기법에 따른 공간 효율과 시간 효율을 비교 및 분석하였다.

Abstract

In this paper, we present DTD-dependent object database schema design methods to efficiently store XML data and process path expression. The similarity between DTD graph model and the object database model, and the characteristics of object database, object references and set-valued attributes, are very profitable to store XML documents into object databases. We propose two kinds of schema design methods. We then compare and anayze space and time complexity for the methods.

I. 서론[1)]

XML은 자기 기술성(self-describing)을 가지는 언어로써, 정보의 표현 및 교환의 표준 포맷으로 정의된 언어이다[2]. 이는 현재 전자상거래, 디지털 도서관, 생명정보학 등 다양한 분야에서 컨텐츠를 관리하기 위해 사용되고 있으며, 또한 그 응용분야가 더욱 넓어지고 그 양이 더욱 증가될 것으로 전망된다. 이러한 환경에서 XML문서를 기존의 데이터베이스를 이용하여 효율적으로 관리하고자 하는 연구가 활발히 진행되고 있다. 현재 Oracle 또는 MS-SQL과 같은 관계형 데이터베이스 시스템의 벤더들과, 객체 데이터베이스 시스템을 이용한 연구[8,20]에서는 XML 문서를 저장, 검색 및 관리를 위한 방안을 제시하고 있다. 기존의 관계형 데이터베이스를 사용하는 경우[1,7,8,14] 트리 기반의 XML 문서를 정형화된 테이블로 변환하는 모델 변환과 그에 따른 질의 변환이 문제점으로 지적되고 있다.[5,14], 객체 데이터베이스의 경우 트리기반의 XML 문서를 복합객체 형태로 저장할 수 있고, 또한 셋값 관련 데이터 타입과 메소드를 지원하므로 XML문서를 저장 및 질의하는데 유연성하다는 장점은 있으나 그 연구가 미흡한 실정이다. 본 논문에서는 Unisql[13] 객체데이터베이스에 XML 문서를 저장하고 XML 질의에 주로 사용되는 경

1) 본 연구는 한국 과학재단의 특정 기초연구 사업의 지원을 받았음(R01-2003-000-11723-0)

로식 처리를 위한 DTD 의존적인 스키마 설계기법을 제안한다. 본 논문에서 제안하는 스키마 설계 기법은 다음과 같은 공헌을 가진다.

- 본 논문에서 제안하는 XML 저장 스키마 구조는 객체 데이터베이스의 고유 특성인 객체참조와 셋 값 속성을 충분히 활용한다. 이것은 XML 문서를 객체데이터베이스의 복합객체의 형태로 저장하는 데 유리하다.
- 기본적(basic) 스키마 생성 기법과 인라인(inlining)을 이용한 스키마 생성기법에 대하여 공간 복잡도를 비교하고 경로식 질의처리 시간을 분석하므로써, 질의 처리 시간과 경로식 길이의 상관관계를 분석한다.

본 논문의 구성은 다음과 같다. 2장에서는 XML 저장 시스템에 관한 관련 연구를 소개하고, 3장에서는 본 논문에서 제안한 객체 데이터베이스 스키마 설계 기법을 소개한다. 4장에서는 스키마 설계에 따른 성능 분석을 제시하고 마지막으로 5장에서는 본 논문의 결론을 맺는다.

Ⅱ. 관련 연구

XML 문서는 계층 구조를 가지는 반 구조적 언어로써 이 문서를 저장하고자 하는 연구는 주로 다음과 같이 3가지로 분류할 수 있다.

1. 관계형 데이터 베이스

관계형 데이터베이스를 이용하는 경우 크게 DTD를 기반으로 데이터베이스 스키마를 생성하여 XML문서를 저장하는 DTD 종속적인(DTD-dependent)방법[7], DTD와 상관없이 스키마를 생성하고 DTD 독립적인(DTD-independent)[3,6,9] 방법이 있다. 관계형 데이터베이스에서는 그래프 기반의 DTD를 단순한 테이블 형태로 변환해야 하고[5], 셋값 속성을 지원하지 않으므로 XML문서에 자주 사용되는 다중 엘리먼트를 표현하기 위해서 테이블을 정규화해야 한다[5,8]. 이로 인하여 테이블 수가 많아지고, XML 질의에서 주로 사용되는 경로식은 고비용의 조인 연산으로 변환되어 성능이 저하되는 문제점이 발생한다. DTD 독립적인 방법의 경우, 제한된 테이블 내에 XML문서를 저장하므로 테이블의 수가 줄어드는 장점은 있지만, DTD와 데이터베이스 스키마와의 모델 불일치로 인하여 XML 질의를 SQL로 변환시 많은 부담을 가지게 된다. 관련 연구[9]에서는 관계 데이터베이스의 기능을 확장하지 않고서 XML 데이터 검색의 성능을 향상시키는 것은 어려운 일이라고 주장한다.

2. XML 전용 저장 시스템

XML 문서를 저장하기 위하여 기존의 저장시스템을 이용하는 것이 아닌 eXcelon[10], Tamino[12]와 같은 전혀 새로운 시스템의 개발이다. 이와같은 시스템은 XML 문서 전체를 파일시스템으로 저장하면서, 문서의 구조 정보와 내용 정보를 이용하여 인덱스를 구축하고 질의할 때 이 인덱스를 이용한다. 이러한 XML 전용 데이터베이스의 경우 XML문서의 저장 공간보다 인덱스 공간이 더 많아지는 단점이 존재하고 대량의 문서를 저장할 경우 질의 처리 성능이 저하되는 단점이 있다[2,6,10].

3. 객체 데이터베이스

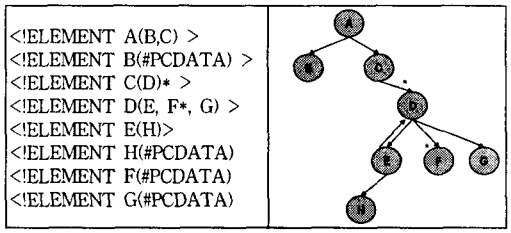
객체 데이터베이스는 관계형 데이터베이스보다 데이터 모델에서 XML과 유사하다. 그러므로 DTD로부터 객체 데이터베이스 스키마를 생성하는 문제가 간단하다. 또한 다중값 속성이 지원되며, sequence와 같은 데이터 타입을 지원하기 때문에 XML의 순서정보가 자연스럽게 표현된다[8]. 또한, XML 질의에 자주 사용되는 경로식은 조인이 아니라 객체 참조 형태로 변환·처리되므로 질의 변환이 용이하고, 성능도 향상된다. 본 논문에서는 객체 참조와 셋값 속성과 같은 객체데이터베이스 특성을 충분히 고려하여 XML을 저장하기 위한 스키마 설계 기법을 제안한다.

Ⅲ. XML 문서 저장 스키마 설계

1. 용어 정의

본 논문에서 사용될 DTD와 인라인 규칙을 설명하기

위한 용어를 정의한다.



▶▶ 그림 1. 예제 DTD 와 DTD 그래프

se는 DTD상에 존재하는 엘리먼트의 하위엘리먼트를 의미하며, S는 엘리먼트 e의 하위 엘리먼트 집합을 뜻한다. 그리고 A는 엘리먼트 e의 애트리뷰트 집합이며, sequenceType은 다중값 속성을 의미하는 ?, *, + 연산자를 가지는 엘리먼트들의 집합이다. #은 엘리먼트의 개수를 의미하고, R(a,b) 는 엘리먼트 a, b 사이에 참조 관계 여부를 알려주는 함수이다.

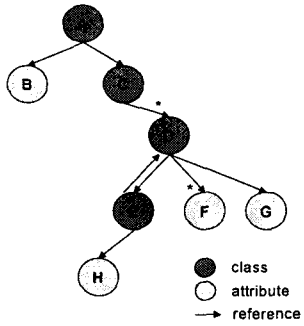
2. 기본 방법

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

For each e,#(S)≧1 OR #(A)≧1⟶define_Class(e)
$s_e \in S \longrightarrow$ Add_attributes_of_Class(e)
$s_e \in$ SequenceType ⟶ Define_multivalued_att(s_e,e)

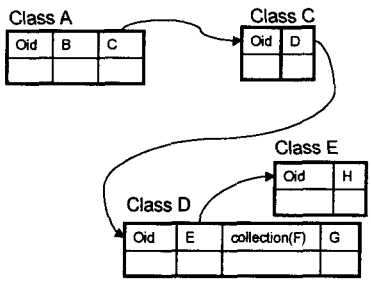
▶▶ 그림 2. 기본 규칙

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 객체데이터베이스 스키마 그래프가 생성된다.



▶▶ 그림 3. 객체데이터베이스 스키마 그래프

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 3의 스키마 그래프를 객체데이터베이스 스키마로 변환하면 그림 4와 같다.



▶▶ 그림 4. 객체데이터베이스 스키마

기본 규칙을 적용하는 경우 DTD의 모델과 객체 데이터베이스 스키마 간의 모델이 일치하므로 XML 질의를 객체 데이터베이스 질의로 변환시 용이하다.

3. 인라인 기법

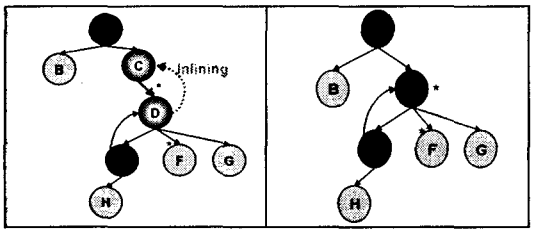
인라인 규칙은 기본 규칙을 이용했을 때, 불필요한 경로를 최대한 줄이기 위한 방법이다. 그림 3의 객체 데이터베이스 스키마에서 클래스 C는 단순히 클래스 A와 D사이에서 경로를 제공해주는 역할만 한다. 이러한 클래스들의 속성을 상위 클래스의 속성으로 인라인 시켜서 클래스의 수를 줄인다. 또한, 경로식의 길이를 줄임

으로써 질의처리 성능을 높이는 것이 주요 아이디어이다. 다음 그림 5은 인라인 기법의 변환 규칙이다.

For each e , #(S) = 1 and se ∈ SequenceType
→ Add_Multi-Valued_attribute_of_Parent-Class(e)

▶▶ 그림 5. 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 엘리먼트 C는 오직 하나의 하위 엘리먼트 D를 가지고, 이 엘리먼트 D는 다중값 속성을 가지므로 그림 5의 규칙에 의해 그림 6의 그림과 같이 상위 엘리먼트의 속성으로 인라인 된다.



▶▶ 그림 6. 인라인 규칙 적용한 스키마 그래프

그림 6에서와 같이 인라인 규칙을 적용하면 기본 규칙을 적용한 데이터베이스 스키마 구조에 비하여 클래스 수가 줄어듦을 볼 수 있다.

Ⅳ. 실험 및 평가

본 논문에서 제안된 DTD를 객체데이터베이스 스키마로 변환하기 위한 두 가지 기법에 대하여 저장 비용 및 질의 비용을 평가한다. 실험에 사용될 DTD 및 XML데이터는 GenBank[10]에서 제공되는 DTD와 GBSeq XML데이터이다.

1. 실험 환경

[표 1] 시스템 사양

구분	사양 및 기종
OS	Window 2000 Server
RAM	512MB
CPU	Pentium4 2.4GHz
DBMS	UniSQL

2. 공간 복잡도

실험에 사용된 GBSeq DTD와 100MB, 200MB, 300MB의 GBSeq XML 문서이다. 기본적으로 Basic 기법과 Inlining 기법에 대하여 표 2과 같이 각각 클래스의 수와 객체의 수를 비교하였다.

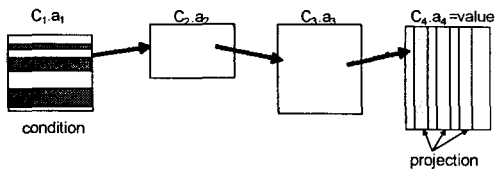
[표 2] 객체 데이터베이스와 관계 데이터베이스의 공간복잡도 비교

규칙 / 데이터용량	기본 방법		인라인 방법	
	class	ⓐobjects	class	ⓑobjects
100MB	17	528104	9	424532
200MB	17	1056208	9	849064
300MB	17	1584312	9	1273596

<표 2>에서 볼 수 있듯이, 기본 방법을 적용했을 때보다 인라인 방법을 적용했을 때, 생성된 클래스의 수가 많이 줄어들었다. 이는 동일한 경로 질의에 대하여 더 짧은 경로를 제공해준다. 또한 저장되는 객체의 수에 있어서도, 인라인 규칙을 적용했을 때 기본 규칙을 적용한 것보다 80% 가량 줄어듦을 알 수 있다. 이는 인라인 기법이 저장 공간의 측면에 있어서도 효율적임을 보여준다.

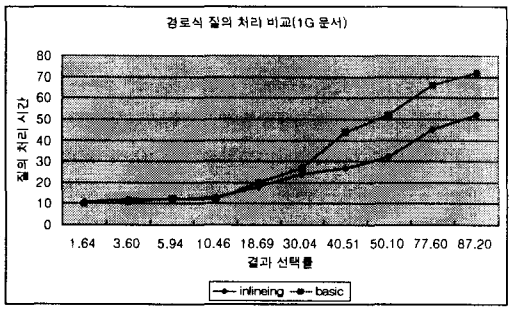
3. 시간 복잡도

두 가지 스키마 변환기법에 대하여 동일한 경로질의에 대한 질의처리 시간을 분석한다. 실험에 이용된 질의는 그림 7과 같다.



▶▶ 그림 7. 경로질의(순방향)

즉 경로의 시작 노드에 조건을 주고 끝노드에 있는 값을 가져오는 순방향 질의 이다. 경로식의 대한 질의 처리시간에 대한 결과는 다음 그림 8과 같다.



▶▶ 그림 8. 질의 처리 시간 비교

결과선택률이 10% 이전에는 기본 방식과 인라인 방식이 비슷한 성능을 보였으나 그 이상의 결과 선택률에 있어서는 인라인 방식이 좀더 낮은 성능을 보였다. 이것은 결과 선택률이 많아질수록 기본 방식보다 인라인 방식에서 경로 탐색비용이 줄어들기 때문이다.

V. 결론 및 향후 연구

본 논문에서는 XML 문서를 객체 데이터베이스에 저장하기 위하여 DTD를 객체 데이터베이스 스키마로 변환하는 기법을 제안하였다. DTD에 의존적인 스키마를 생성하면서 기본 방법과, 인라인 방법을 소개하였다. 뿐만 아니라, 각각의 기법에 대하여 공간복잡도와 시간복잡도를 비교하였다. 기본 방식은 DTD와 객체데이터베이스 스키마 사이에 모델이 유사하므로 XML 질의를 객체데이터베이스 질의로 변환하는데 유리하다. 하지만, 필요없는 정보로 인하여 클래스 수와 객체수에 있어서 인라인 방식보다 데이터베이스 공간 복잡도와 시간복잡도에서 불리하다는 것을 실험을 통해 확인할 수 있었다. 향후 연구로는, XML 질의의 특성을 분석하고, 이러한 질의를 객체데이터베이스 스키마에 적합하게 하는 매핑하는 기법을 추가해 나갈 예정이다.

▌참고문헌▐

[1] S. Abiteboul, P. Buneman. and D. Suciu, Data on the web: from relations to semistructured data and XML, Morgan Kaufmann Publisher, Los Altos, CA114022, USA, 1999.

[2] T. Bay, J Paoli, C.M Sperberg-McQueen, and E. Maler, "Extensible Markup Language(XML) 1.0 (second edition)," W3C Recommendation, http://www.w3.org/TR/REC-xml, 2000.

[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] J. McHugh and J. Widom.,"Query Optimization for XML," In Proc. Intl. Conf. on the 25th VLDB, 1999.

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

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

[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] 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.

[9] P. M. Tolani and J.R. Haritsa, "XGrind: A Query-friendly XML compressor," In proceedings of the 18th International Conference on Database Engineering, 2002.

[10] eXcelon Inc., http://www.exceloncorp.com/

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

[12] Tamino database, http://www.softwareag.com/tamino/

[13] UniSQL, http://www.unisql.com/

[14] XML-DBMS, http://www.rpbourret.com/xmldbms/

[15] 정태선, 박상원, 한상영, 김형주, "XML 데이터를 위한 객체지향 데이터베이스 스키마 및 질의처리," 한국정보과학회 논문지 : 데이터베이스, 29(2), 2002.