웹에서 캐쉬를 이용한 XML 질의 처리: 구현 및 성능 평가*

박정기° 강현철

중앙대학교 컴퓨터공학부

jkpark@dblab.cse.cau.ac.kr hckang@cau.ac.kr

Cache-Answerability of XML Queries on the Web: Implementation and Performance Evaluation

Jungkee Park° Hyunchul Kang

School of Computer Science and Engineering

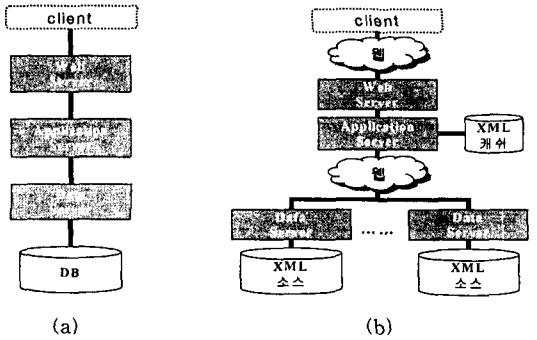
Chung-Ang University

요 약

데이터베이스 기반의 웹 응용을 위한 캐싱 기법이 최근 많이 연구되고 있다. 자주 제기되는 질의의 결과를 캐쉬해두면 반복 질의를 위한 재사용은 물론 관련 질의의 처리에 이용될 수 있다. 웹 상에서 데이터 교환의 표준으로 XML이 등장한 이래 현재 웹 응용들은 네트워크 상의 원격 XML 소스로부터 데이터 검색을 수행하는 경우가 많아졌는데 이의 효율적인 지원을 위해 검색 결과를 캐쉬하는 것은 유용하다. 본 논문은 웹에서 XML 질의를 관련 XML 캐쉬를 이용하여 처리하는 기법의 구현 및 성능 평가에 관한 것이다. XML 질의로 XQuery, XPath, XQL 등과 같은 모든 XML 질의어의 핵심 요소인 경로 표현식을 대상으로 하였고, XML 캐쉬는 XML 실체뷰를 고려하였고, 캐쉬를 이용한 XML 질의 변환 알고리즘은 [12]에 제시된 것을 대상으로 하였다. [12]의 질의 변환 알고리즘을 지원하는 프로토타입 XML 저장 시스템이 관계 DBMS를 이용하여 구현되어 실제 웹에서의 성능 실험에 이용되었다. 성능 실험 결과 웹에서 캐쉬를 이용한 XML 질의 처리의 효율성을 확인하였다.

1. 서론

자주 제기되는 질의의 결과를 캐쉬해 두었다가 후속 질의에 사용하는 것은 중요한 질의 최적화 기법 중의 하나이다. 소스 데이터에 대한 변경을 반영하여 캐쉬를 갱신(refresh)하는 것이나 캐쉬 교체(replacement)와 같은 캐쉬 관리가 적절히 이루어진다고 할 때, 결과를 캐쉬해 둔 질의와 동일한 질의가 반복 제기되면 그 결과는 즉각 주어질 수 있고, 완전히 동일하지는 않지만 관련된 질의가 제기되면 해당 캐쉬에 대한 질의로 변환(rewrite)되어 캐쉬를 이용하여 처리될 수 있다. 이와 같은 캐쉬를 이용한 질의 변환



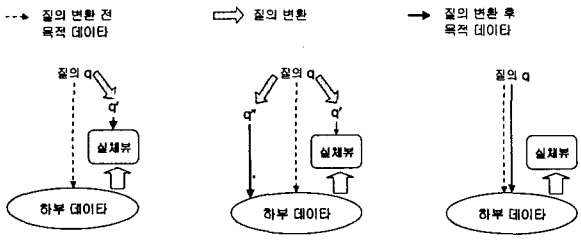
(그림 1) XML 데이타베이스 기반 웹 응용을 지원하는 다계층 구조

* 본 논문은 정보통신부의 정보통신기초기술연구지원사업(정보통신 연구진흥원)으로 수행한 연구결과입니다.

및 처리 기법은 관계 데이터베이스 시스템에서 많이 연구되었지만 [1], 90년대 후반부터 웹 환경에서 데이터베이스 기반의 웹 응용을 위한 질의 처리 기법을 위해 반구조적 데이터나 XML 데이터에 대해 많이 연구되고 있다 [2-8].

(그림 1)(a)는 널리 사용되고 있는 데이터베이스 기반의 웹 응용(database-backed web application)을 지원하는 웹 서버, 응용 서버, 데이터 서버의 다계층(multi-tier) 구조를 나타낸 것이다. 이 구조에서 웹 서버는 웹 페이지에 포함된 질의를 미들웨어 층인 응용 서버를 통해 데이터 서버로 보내게 된다. 질의 결과는 데이터 서버에 캐쉬되어 그 소스와 함께 존재할 수도 있고 응용 서버에 캐쉬되어 있을 수도 있다.

본 논문은 (그림 1)(b)와 같이 웹에서 응용서버의 XML 캐쉬를 이용하여 주어진 XML 질의를 변환, 처리하는 기법의 구현 및 성능 평가에 관한 것이다. 본 논문에서는 XML 질의로 XQuery [9], XPath [10], XQL [11] 등과 같은 모든 XML 질의어의 핵심 요소인 경로 표현식(path expression)을 대상으로 하였고, XML 캐쉬는 XML 실체뷰를 고려하였고, 캐쉬를 이용한 XML 질의 변환 알고리즘은 [12]에 제시된 것을 대상으로 하였다. (그림 2)는 XML 질의 처리에 실체뷰를 이용하는 두가지 경우와 그렇지 못한 경우를 나타내고 있다. (그림 2)(a)는 주어진 질의의 결과를 모두 관련 실체뷰에서 얻을 수 있는 경우를 나타낸 것으로 질의 q가 실체뷰에 대한 질의 q′으로 변환되어 처리된다. 이와 같은 질의 변환 유형을 MV(Materialized View Only)라고 부른다. (그림 2)(b)는 질의 q의 결과 일부는 실체뷰로부터, 나머지는 하부 XML 소스로부터 얻는 경우를 나타낸 것으로, q는 실체뷰에 대한 질의 q′과 하부 데이터에 대한 질의 q″로 변환되어 처리되고 q′과 q″의 결과는 통합되어 최종 결과를 얻는다. 이와 같은 질의 변환 유형을 BD+MV(Both Base Data and Materialized View)라고 부른다. (그림 2)(c)는 질의 처리에 관련 실체뷰를 이용할 수 없거나 이용



(그림 2) 실체뷰를 이용한 질의 처리의 유형

하지 않는 경우를 나타낸 것으로 질의 변환은 일어나지 않는데 BD(Base Data Only) 유형이라 부른다.

본 논문의 구성은 다음과 같다. 2절에서는 관계 DBMS를 XML저장소로 이용하여 수행된 웹에서 캐쉬를 이용한 XML 질의 처리 시스템의 구현에 대해 기술하고, 3절에서는 구현된 시스템을 이용하여 실제 웹상에서 수행한 초기 실험 결과를 기술한다. 4절에서는 결론을 맺고 향후 연구 과제를 제시한다.

2. 캐쉬를 이용한 XML 질의 처리 시스템의 구현

본 시스템의 구현을 위해 고려되어야 할 주요 항목으로는 (1) 뷰의 하부 XML 소스 및 실체뷰 등을 저장하는 테이블 스키마, (2) 경로 표현식으로 표현된 XML 질의의 SQL로의 변환, 그리고 (3) SQL 결과 셋의 XML 태깅을 들 수 있다.

2.1 테이블 스키마

XML 저장소는 크게 하부 데이타 영역과 XML 실체뷰 영역으로 나누어진다. 하부 데이타 영역은 DTD 정보를 기록하는 DTD_Info 테이블과 소스 XML 문서들을 엘리먼트 단위로 분할하여 XML 문서의 구조 정보, 내용 정보, 그리고 경로 정보를 각각 따로 저장한 Element_Info 테이블, Element_Content 테이블, Path_Info 테이블, 그리고 XML 문서의 기타 정보를 저장한 Doc_Info 테이블로 구성된다. XML 질의 결과인 XML 실체뷰는 XML 문서로 모델링하였다. 따라서 XML 실체뷰 영역 또한 하부 데이터 영역과 비슷하게 XML 실체뷰들을 엘리먼트 단위로 분할하여 뷰의 구조 정보와 내용 정보, 경로 정보를 각각 따로 저장한 View_Info 테이블과 View_Content 테이블, View_Path_Info 테이블, 그리고 뷰 정의를 저장하는 View_Definition 테이블로 구성된다.

2.2 XML 경로 표현식의 SQL로의 변환

XML 소스 및 XML 실체뷰에 대한 XML 경로 표현식은 이들이 관계 DBMS의 테이블에 저장되어 있기 때문에 SQL로 변환되어 처리된다. MV 유형의 경우, 변환된 XML 경로 표현식은 View_Info 테이블, View_Content 테이블, 그리고 View_Path_Info 테이블에 대한 SQL 구문으로 변환되고, BD+MV 유형의 경우, XML 소스 및 실체뷰에 대한 XML 경로 표현식은 각각 Element_Info 테이블, Element_Content 테이블, Path_Info 테이블로 구성된 하부 데이타에 대한 SQL 구문과 View_Info 테이블, View_Content 테이블, View_Path_Info 테이블로 구성된 실체뷰 데이타에 대한 SQL 구문으로 변환된다.

2.3 XML 태깅

XML 경로 표현식으로부터 변환된 SQL 문의 수행 결과는 튜플 셋이다. 이 결과를 가지고 질의의 최종 결과를 XML 형식으로 구성하기 위해서는 적절한 태깅이 이루어져야 한다. 본 구현에서는 SQL 문의 결과를 구성하는 튜플이 반환될 때마다 즉각 그것을 태깅 프로세스로 넘겨 질의 결과를 XML로 생성하도록 하였다. 즉, SQL 문의 수행 결과인 튜플 셋을 모두 한 곳에 버퍼링한 후에 비로소 태깅 작업에 들어가는 것이 아니라, SQL 문의 결과 튜플이 하나씩 나올 때마다 태깅 프로세스에 바로 입력되어 최종 결과 XML 문서를 생성하는 파이프라인(pipeline) 방식의 태깅이 수행된다.

3. 성능 평가

본 절에서는 구현된 프로토타입 XML 저장 시스템을 대상으로 웹 상에서 실체뷰를 이용한 XML 질의 처리의 성능을 실제 실험을 통해 평가한 기초 결과를 기술한다. 본 실험은, XML 데이터베이스 기반 웹 응용을 지원하는 웹 서버, 응용 서버, 데이타 서버의 다계층 구조를 대상으로, XML 소스는 웹 상의 원격 사이트에 존재하고 XML 실체뷰는 [5-7] 등의 연구에서와 같이 응용 서버에 캐쉬되는 환경을 대상으로 하였다.

3.1 개요

본 실험의 성능 척도는 웹 응용에서 XML 질의 처리를 수행할 때의 응답 시간이다. 질의 처리 시 응용 서버에 캐쉬된 XML 실체뷰를 이용할 수도 있고 그렇지 않을 수도 있는데 질의가 응용 서버에 전달되기까지의 시간과 질의 결과가 응용 서버로부터 사용자에게 전달되는 시간은 양자 모두 동일하므로 측정에서 제외하였다. 즉, 응용 서버에 XML 질의가 제기된 시점부터 질의 결과(관계 DBMS로부터 검색된 결과 셋을 태깅 처리한 XML 문서)가 응용 서버에 확보되기까지 걸린 시간으로, MV, BD+MV, 그리고 BD 유형의 XML 질의가 주어졌을 때, 이들 간에 처리 시간을 비교하였다. 실험 환경 및 데이터는 <표 1>과 같다.

<표 1> 실험 환경 및 데이타

항 목		내 용
데이터 서버	CPU	Dual PentiumIII 1.13GHz
	메모리	2304MB
	OS	Windows 2000 Server
	DBMS	Oracle 9i
응용 서버	CPU	Dual PentiumIII 550MHz
	메모리	1024MB
	OS	Windows 2000 Server
	DBMS	Oracle 8i
실험 데이터		"영화" XML 문서 문서당 평균 엘리먼트 개수 22개 문서당 평균 크기 2KB
웹의 네트워크 환경		100Mbps LAN

<표 2> 실험에서 사용된 경로 표현식

실험결과	구분	경로 표현식
(그림 3)	q	/영화/평가/평/내용
	MV-I [23%]	/영화/평가
	MV-II [12%]	/영화/평가/평/내용
(그림 4)	q	/영화/평가
	MV	/영화/평가
	MV+BD(92%)	/영화/평가/평
	MV+BD(51%)	/영화/평가/평/내용
	MV+BD(28%)	/영화/평가/평/평론가

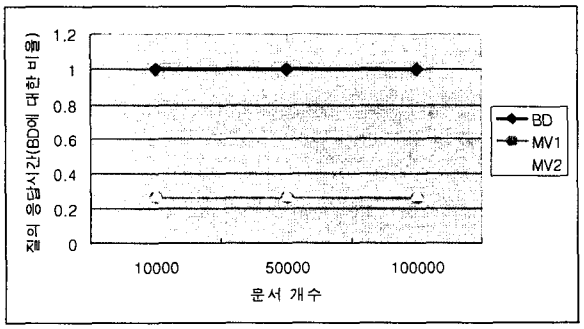
<표 2>는 본 실험에서 사용된 질의 q, 실체뷰 정의 v를 정리한 것이다. (그림 3)의 실험에서는 두개의 서로 다른 뷰 정의로 구성된 실체뷰를 대상으로 MV 유형의 성능을 측정했기 때문에 MV-I과 MV-II로 나누어 표시하였는데 [ ] 안에 표시된 % 값은, MV-I의 실체뷰는 뷰 정의에 의해 XML 소스의 23%에 해당되는 뷰를 실체화한 것이고, MV-II의 실체뷰는 12%에 해당되는 뷰를 실체화한 것임을 나타낸다. (그림 4)에서는 BD+MV 유형의 질의 처리 시 세개의 서로 다른 뷰 정의로 구성된 실체뷰를 대상으로

실험하였는데 ( ) 안에 표시된 % 값은, 질의 q의 결과 엘리먼트들 중 실체뷰로부터 검색된 것의 비율 (즉, 실체뷰에 대한 질의 결과 엘리먼트가 q의 최종 결과에서 차지하는 비율)를 나타낸다. 예를 들어, 질의 q의 결과 엘리먼트를 XML 소스와 실체뷰에서 각각 반씩 얻었다면, 50%, 결과 전체를 XML 소스로부터 얻었다면 0%, 그리고 결과 전체를 실체뷰로부터 얻었다면 100%가 된다.

3.2 실험 결과

3.2.1. MV와 BD의 비교

(그림 3)은 XML 소스의 12%에 해당되는 데이타를 검색하는 질의 처리 실험 결과이다. 곡선 MV-I는 XML 소스의 23%에 해당되는 실체뷰를 이용하여 MV 유형의 질의 처리를 수행했을 때의 성능을, MV-II는 XML 소스의 12%에 해당되는 실체뷰 (즉, 실험 질의의 답과 완전 일치(exact match)하는 경우에 해당)를 이용한 MV 유형 질의 처리의 성능을 BD의 성능을 기준으로 정규화하여 나타낸 것이다. (그림 3)에 나타난 바와 같이 BD보다 검색비용과 통신비용에서의 이득으로 MV의 성능이 우수하다. 본 실험에서는 100Mbps의 LAN환경이기 때문에 통신비용이 미비하지만 느린 웹 환경에서는 통신비용이 대폭 증가하여 XML 소스 문서의 개수 증가에 따른 BD 대비 MV의 성능 향상이 더욱 클 것으로 기대된다.



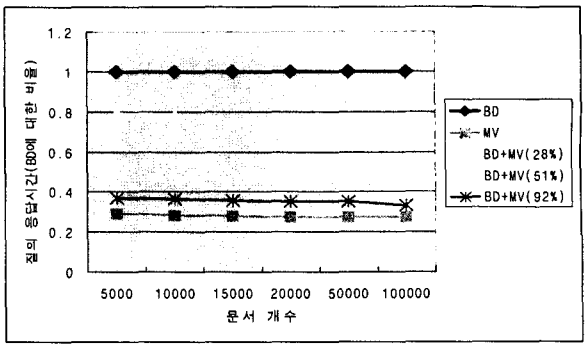
(그림 3) XML 소스 문서 개수의 변화에 따른 BD, MV 비교

3.2.2. BD+MV와 BD의 비교

(그림 4)는 XML 소스의 23%에 해당되는 데이타를 검색하는 질의 처리 실험 결과이다. XML 소스 문서의 개수가 5000개, 10000개, 15000개, 20000개, 50000개, 100000개일 때 각각 BD+MV와 BD의 성능을 정규화 하여 비교한 결과인데 BD+MV가 BD보다 우수하게 나타났으며 실체뷰로부터의 검색 비율이 증가함에 따라 BD+MV의 성능이 계속 향상되는 것으로 나타났다. 이는 BD+MV의 경우 XML 소스에 대한 접근과 실체뷰에 대한 접근이 웹 상의 서로 다른 사이트 상에서 병렬로 처리되어 검색비용이 적고, BD에 비해 통신 비용도 적기 때문이다.

4. 결론

본 논문은 웹에서 XML 질의를 관련 XML 캐쉬를 이용하여 처리하는 기법의 구현 및 성능 평가에 관한 것이다. 캐쉬를 이용한 XML 질의 처리 기능을 지원하는 XML 저장 시스템 프로토타입이 관계 DBMS를 기반으로 구현되어 성능 실험에 이용되었다. XML 질의로는 모든 XML 질의어의 핵심 요소인 경로 표현식을 대상으로 하였고, XML 캐쉬는 XML 실체뷰를 고려하였다. 캐쉬를 이용한 질의 처리 유형 결정 및 변환 알고리즘은 [12]에 제시된 것을 이용하였다. 구현의 주요 이슈로는 XML 소스와 실체뷰를 저장하는 테이블 스키마, XML 질의의 SQL로의 변환, 그리고 SQL 결과 셋에 대한 XML 태깅을 들 수 있다. 구현된 시스템으로 데이터베이스 기반 웹 응용을 위한 웹 서버, 응용 서버, 데이터 서버의 다계층 구조 하에서 응용 서버에 캐쉬를 둘 때 원격 XML 소스에 대한 XML 질의 처리의 성능을 평가하였다. 실험



(그림 4) XML 소스 문서 개수의 변화에 따른 BD,MV+BD 비교

결과 실체뷰를 사용하지 않는 BD 유형의 처리에 비해, 질의의 답을 실체뷰로부터 모두 얻는 MV 유형이나 질의의 부분 답을 실체뷰와 하부 XML 소스로부터 각각 얻어 통합하는 BD+MV 유형의 성능이 모두 우수하게 나왔다.

향후 연구 과제로는, XML 소스 문서 및 실체뷰를 분할 저장하기 위한 테이블 스키마로 본 논문의 구현에서 사용된 것 이외의 것을 채택했을 때 캐쉬를 이용한 XML 질의 처리 성능에의 영향 파악, XML 경로 표현식에 정규 경로가 허용될 때 실체뷰를 이용한 질의 변환 알고리즘의 개발, 웹 환경에서 다수 사용자에 의한 질의 처리 시 캐쉬 이용 여부에 따른 확장성(scalablilty) 비교 등이 있다.

참고문헌

[1] A. Levy et al., "Answering Queries Using Views," Proc. of : ACM Int'l Symp. on PODS, 1995.

[2] Y. Papakonstantinou and V. Vassalos, "Query Rewriting for Semistructured Data," SIGMOD Proc. Int'lConf. on Management of Data, pp. 455-466, 1999.

[3] D. Calvanese et al., "Answering Regular Path Queries Using Views," Proc. Int'l Conf. on Data Eng., pp. 389-398, 2000.

[4] D. Florescu et al., "Query Containment for Conjunctive Queries with Regular Expressions," Proc. Int'l Symp. on PODS, 1998, pp. 139-148.

[5] V. Hristidis and M. Petropoulos, "Semantic Caching of XML Databases," Proc. Workshop on the Web and Databases, 2002.

[6] L. Chen and E. Rundensteiner, "ACE-XQ: A CachE-aware XQuery Answering System," Proc. Workshop on the Web and Databases, 2002.

[7] L. Chen et al., "XCache - A Semantic Caching System for XML Queries," Proc. ACM SIGMOD Int'l Conf. on Management of Data, 2002.

[8] P. Marron and G. Lausen, "Efficient Cache Answerability for XPath Queries," Proc. the 2nd Int'l Workshop on Data Integration over the Web, 2002, pp. 35-45.

[9] S. Boag et al., "XQuery 1.0: An XML Query Language," http://www.w3.org/TR/xquery/, 2002.

[10] A. Berglund et al., "XML Path Language (XPath) 2.0," http://www.w3.org/TR/xpath20/, Nov. 2002.

[11] J. Robie et al., "XML Query Language (XQL)," http://www.w3.org/TandS/QL/ QL98/pp/xql.html, 1998.

[12] C. Moon et al., "Processing XML Path Expressions Using XML Materialized Views," Proc. BNCOD, Jul. 2003, pp. 19-37.