고성능 XML 질의 처리를 위한 XML 뷰 인덱스의 생성 및 실체화 기법*

박대성° 김영현 강현철
중앙대학교 컴퓨터공학부
{dspark, yhkim}@dblab.cse.cau.ac.kr hckang@cau.ac.kr

Creation and Materialization of XML View Indices for High Performance XML Query Processing

Daesung Park° Younghyun Kim Hyunchul Kang
School of Computer Science and Engineering, Chung-Ang University

요 약

웹에서 XML 데이터의 양이 많아짐에 따라 XML 질의 처리를 신속하게 해주는 기술이 필요하게 되었다. 이를 가능하게 해주는 것이 XML 질의 캐싱이다. 자주 제기되는 질의에 대하여 질의 결과를 캐쉬한 후, 동일 질의에 재사용함으로써 빠른 응답속도를 보장할 수 있다. 본 논문은 XML 질의 캐싱 기법 중 캐쉬되는 데이터의 공간 부담을 줄일 수 있는 XML 뷰 인덱싱 기법에 관한 것으로, 뷰 인덱스의 생성 및 실체화 기법을 제안하고 구현 및 실험을 통한 성능 평가 결과를 기술한다.

1. 서 론

웹의 등장으로 말미암아 정보통신 분야를 포함한 모든 분야에서 정보의 양이 폭발적으로 증가하였다. 또한 XML이 웹 상에서의 데이터 교환의 표준으로 부각되어 웹에서의 XML 문서가 차지하는 비중이 점점 커지고 있다. 이에 따라서 웹상의 방대한 XML 소스에 대한 질의 처리를 신속하게 해주는 기술이 필요하게 되었다.

e-Commerce와 같은 XML 데이터베이스 기반의 웹 응용(XML database-backed web application)은 웹 상에 산재해 있는 XML 소스로부터 XML 질의 결과를 얻어온다. 이러한 작업을 동일한 질의가 제기될 때마다 매번 수행하는 것은 매우 비효율적일 수 있다. 이러한 문제를 해결하기 위한 대표적 기술이 XML 질의 캐싱이다. 자주 제기되는 질의에 대하여 질의 결과를 캐쉬한 후, 동일 질의의 처리에 이를 재사용함으로써 빠른 응답속도를 보장할 수 있다. 이 기술의 핵심 요소 기술 중의 하나는 질의의 하부 소스 데이터에 발생한 변경을 캐쉬에 점진적으로 반영하여 갱신(incrementally refresh)함으로써 캐쉬의 일관성을 유지하는 것이다. 이러한 기법들은 90년대 후반부터 반구조적(semistructured) 데이터 및 XML 데이터에 대해서 연구되고 있다[1][2][3].

XML 질의 캐싱은 XML 실체뷰나 XML 뷰 인덱싱 기법으로 구현될 수 있다. 이때, 뷰란 XML 질의어로 정의된 질의의 결과를 뜻한다. XML 실체뷰의 경우 질의 결과를 바로 반환할 수 있지만 뷰의 크기가 클 수 있기 때문에 많은 수의 뷰를 유지할 경우 공간 비용 문제가 발생한다. 한편 XML 뷰 인덱싱 기법은 질의 결과를 뷰 인덱스 형태로 캐쉬한 후 동일한 질의의 결과를 반환할 때 이를 사용한다. 이때, 뷰 인덱스란 질의 결과를 구성하는 데이터들을 가리키는 포인터들 및 뷰의 하부 소스 데이터의 변경을 반영하여 뷰 인덱스를 점진적으로 갱신할 때 필요한 보조정보로 이루어진 자료구조를 말한다. 때문에 해당 질의 결과를 반환하기 위해서는 뷰 인덱스의 포인터들을 실제 데이터로 변환하는 실체화 과정을 거쳐야 한다. XML 뷰 인덱싱 기법은 뷰 인덱스만을 캐쉬하므로 XML 실체뷰 기법에 비하여 훨씬 많은 수의 뷰를 유지할 수 있다.

뷰 인덱싱의 요소 기술로는 뷰 인덱스의 생성, 실체화, 갱신 기술이 있다. 본 논문에서 제안하는 뷰 인덱싱 기술은 질의 결과에 해당하는 정보만을 뷰 인덱스에 유지시켜 뷰 인덱스의 공간 부담을 줄였다. 본 논문에서는 XML 뷰 인덱싱 기법의 요소 기술 중 뷰 인덱스 생성 및 실체화 기법을 제안한다.

본 논문의 구성은 다음과 같다. 2절에서는 본 논문의 뷰 인덱스 자료구조 및 생성 방법에 대해 기술한다. 3절에서는 뷰 인덱스의 실체화 방법을 기술한다. 4절에서는 이들의 구현과 성능 평가 결과를 기술한다. 마지막으로 5절에서 결론을 맺고 향후 연구 내용을 기술한다.

2. 뷰 인덱스의 자료구조 및 생성 방법

본 논문에서는 세가지 XML 뷰 인덱싱 기법을 제안한다. 본

* 본 논문은 정보통신부의 정보통신기초기술연구지원사업(정보통신 연구진흥원)으로 수행한 연구결과입니다 (과제번호: 04-기초-082).

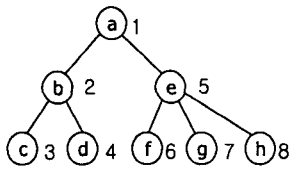

그림 1. XML 문서

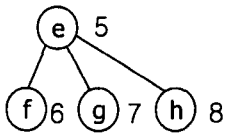

그림 2. 뷰 V=/a/e

절에서는 제안하는 각 기법의 뷰 인덱스 자료구조와 그것의 생성 방법을 기술한다. 그림 1은 XML 문서를 트리 구조로 나타낸 것으로서 DID(Document ID)값이 1이며, 각 EID(Element ID)는 XML 트리의 Preorder 탐색 순서로 할당되었다. 본 논문의 뷰 정의는 XPath[4] 구문으로 정의된다고 가정한다. 그림 2는 뷰 정의 V=/a/e의 결과 XML 트리이다.

2.1 TE/DE(Target Element's DID&EID) 기법

기법 TE/DE의 뷰 인덱스 자료구조는 뷰 정의를 만족하는 목적 엘리먼트 (즉, V의 경우 그림 1의 e)의 DID와 EID의 쌍의 집합으로 이루어진다. 즉, {(d,e)|d=DID, e=Target Element's EID}이다. 그림 1과 같은 XML 문서에 대하여 뷰 V가 정의될 경우, 뷰 인덱스는 V의 목적 엘리먼트인 e 엘리먼트의 DID와 EID의 쌍인 (1,5)로 구성된 집합 {(1,5)}이다.

2.2 TE/DER(Target Element's DID&EID&RmdEID) 기법

기법 TE/DER의 뷰 인덱스 자료구조는 뷰 정의를 만족하는 목적 엘리먼트의 DID와 EID 그리고 RmdEID(Rightmost Descendant Element ID)의 세 쌍의 집합으로 이루어진다. 즉, {(d,e,r)|d=DID, e=Target Element's EID, r=Target Element's RmdEID}이다. RmdEID란 가장 하위의 자손 중 가장 오른쪽에 존재하는 엘리먼트의 EID를 말한다.그림 1과 같은 XML 문서에 대하여 뷰 V가 정의될 경우, e의 RmdEID는 8이므로 기법 TE/DER의 뷰 인덱스는 집합 {(1,5,8)}이다.

2.3 AE/DE(All Element's DID&EID) 기법

기법 AE/DE의 뷰 인덱스 자료구조는 뷰 정의를 만족하는 목적 엘리먼트 및 그 하위 엘리먼트들의 DID와 EID의 쌍으로 이루어진다. 즉, {(d,e)|d=DID, e=Target or its Child Element's EID}이다. 그림 1과 같은 XML 문서에 대하여 뷰 V가 정의될 경우, 기법 AE/DE의 뷰 인덱스는 집합 {(1,5), (1,6), (1,7), (1,8)}이다. 표 1은 이상의 내용을 정리한 것이다.

표 1. 뷰 인덱스 예

XML 뷰 인덱싱 기법	뷰 인덱스
TE/DE	{(1,5)}
TE/DER	{(1,5,8)}
AE/DE	{(1,5),(1,6),(1,7),(1,8)}

3. 뷰 인덱스의 실체화 방법

본 절에서는 본 논문이 제시하는 세가지 XML 뷰 인덱싱 기법의 실체화 방법을 기술한다. 각 방법마다 그림 2의 뷰 V를 예로 들어 설명한다.

3.1 TE/DE(Target Element's DID&EID) 기법

기법 TE/DE의 뷰 인덱스는 목적 엘리먼트의 (DID,EID)쌍의 집합으로 이루어진다. 실체화 방법은 우선, XML 소스에서 목적 엘리먼트의 DID와 EID가 같은 엘리먼트 (즉, V의 경우 그림 1의 e)를 찾는다. 둘째로 찾아낸 엘리먼트의 RmdEID (즉, 8)를 알아낸다. 끝으로 XML 소스에서 목적 엘리먼트의 DID (즉, 1)와 같으며, EID가 목적 엘리먼트의 EID (즉, 5)와 RmdEID (즉, 8) 그리고 그사이에 존재하는 엘리먼트 (즉, e,f,g,h)들을 찾아 질의 결과를 구성한다.

3.2 TE/DER(Target Element's DID&EID&RmdEID) 기법

기법 TE/DER의 뷰 인덱스는 목적 엘리먼트의 (DID,EID,RmdEID) 세 쌍의 집합으로 이루어진다. 기법 TE/DER는 기법 TE/DE와는 달리 RmdEID 정보를 저장해두기 때문에 실체화 과정에서 RmdEID를 찾는 비용이 들지 않는다. 실체화 방법은 XML 소스에서 목적 엘리먼트의 DID (즉 1)와 같으며, EID가 목적 엘리먼트의 EID (즉, 5)와 RmdEID (즉, 8) 그리고 그사이에 존재하는 엘리먼트 (즉, e,f,g,h)들을 찾아 질의 결과를 구성한다.

3.3 AE/DE(All Element's DID&EID) 기법

기법 AE/DE의 뷰 인덱스는 목적 엘리먼트의 하위 엘리먼트들의 (DID,EID) 쌍의 집합으로 이루어진다. 실체화 방법은 XML 소스에서 하위 엘리먼트들의 DID (즉, 1)와 EID (즉, 5,6,7,8)가 같은 엘리먼트 (즉, e,f,g,h)들을 찾아 질의 결과를 구성한다.

4. 성능평가

본 논문에서 제시한 XML 뷰 인덱싱 기법을 관계 DBMS 시스템을 기반으로 구현하였다. 실험에 사용된 서버는 Windows 2000 Server이며 Pentium IV 1.3GHz 듀얼 CPU가 탑재된 것이다. 메모리는 2304MB이며 DBMS는 Oracle 9i를 사용하였다. 실험에 사용된 데이터는 세익스피어 희곡 XML 문서[5]이다 (표 2 참조). 표 3은 실험에 사용된 뷰 정의들이다. 본 실험에서는 XML 뷰 인덱싱 기법과 질의 재수행 기법 간의 질의 응답 시간을 비교하였다. 질의 재수행이란 캐싱 기법을 사용하지 않고 질의를 처리하는 방법을 말한다.
실험에서 사용된 질의 재수행 기법은 본 연구실에서 개발한 시스템으로서 XRel[6]에 XML 변경 기능이 부가된 시스템의

XML 질의 처리 알고리즘에 의한 것이다. 실험 결과의 수치는 10회 실험 결과의 평균을 나타낸 것이다. 실험 결과를 요약하면 다음과 같다. 그림 3은 본 논문에서 제안한 XML 뷰 인덱싱 기법의 응답 시간이 질의 재수행 기법의 그것보다 훨씬 빠르다는 것을 보여준다. 그림 4는 XML 뷰 인덱싱의 각 기법 간의 응답 시간의 차이를 보여준다. 그림 5는 뷰 인덱스의 생성 시간을 나타낸 것이다. 뷰의 목적 엘리먼트가 많은 하위 엘리먼트를 가질수록 기법 AE/DE가 나머지 두 기법보다 뷰 생성 시간이 많이 걸리는 것으로 나타났다. 뷰 인덱스 생성 비용은 뷰 생성시 한번만 소요되므로 질의 처리의 성능에는 영향을 미치지 않는다.

표 2. 실험 데이터[세익스피어 희곡 XML 문서]

상세 정보

#of documents	37
Total size (MB)	7.65
Average size/document (KB)	206.71
#of element nodes	179,689
#of attribute nodes	0
#of text nodes	147,442
#of simple paths	57

표 3. 뷰 정의

뷰 정의
Q1 /PLAY/ACT/SCENE/SPEECH/LINE/STAGEDIR
Q2 //SCENE/TITLE
Q3 //ACT//TITLE
Q4 /PLAY/ACT/SCENE/SPEECH[SPEAKER='CURIO']
Q5 /PLAY/ACT/SCENE[//SPEAKER='Steward']/TITLE

5. 결론 및 향후연구

본 논문에서는 고성능 XML 질의 처리를 위한 XML 뷰 인덱싱의 세가지 기법 TE/DE, TE/DER, 그리고 AE/DE의 뷰 인덱스 생성 및 실체화 기법을 제안하고 구현한 후 실험을 통하여 그 성능을 비교, 평가 하였다. 향후 연구 과제는 각 기법 별로 XML 소스의 변경에 대한 뷰 인덱스의 점진적 갱신 기법을 연구하는 것이다.

참고문헌

[1] L. Chen and E. Rundensteiner, " Aggregate Path Index for Incremental Web View Maintenance," Proc. 2nd Int'l Workshop on Advanced Issues of E-Commerce and Web-based Information Systems, 2000.

[2] L. Quan et al., " Argos: Efficient Refresh in an XQL-Based Web Caching System, " Proc. Workshop on the Web and Databases, 2000, pp. 23-28.

[3] S. Abiteboul et al., " Incremental Maintenance for Materialized Views over Semistructured Data," Proc. Int'l Conf. on VLDB, 1998, pp. 38-49.

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

[5] http://metalab.unc.edu/bosak/xml/eg/shaks200.zip

[6] M. Yoshikawa et al., " XRel: A path-based approach to storage and retrieval of XML documents," ACM Transactions on Internet Technology, 2001, pp. 110-141.

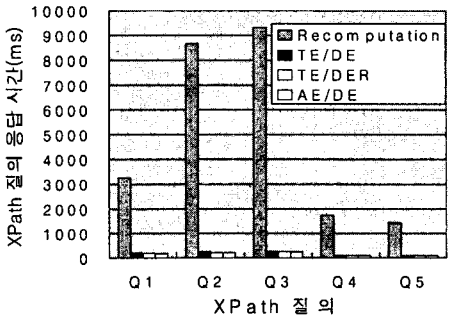

그림 3. XML 질의 응답 시간

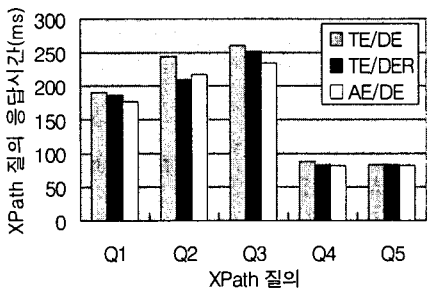

그림 4. XML 질의 응답 시간

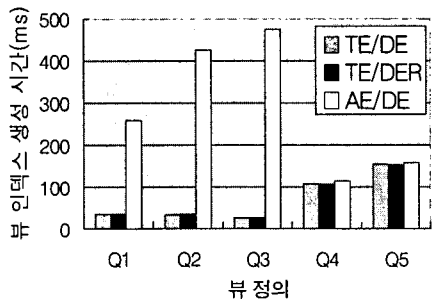

그림 5. 뷰 인덱스 생성 시간