2001년 한국정보처리학회 춘계 학술발표논문집 제8권 제1호

XML 문서 실체 뷰의 점진적 갱신의 성능 분석

임재국°, 문찬호, 강현철
중앙대학교 컴퓨터공학과
e-mail:{jklim, moonch, hckang}@dblab.cse.cau.ac.kr

Performance Analysis of Incremental Refresh of Materialized Views over XML Documents

JaeGuk Lim°, ChanHo Moon, Hyunchul Kang
Dept. of Computer Science and Engineering, Chung-Ang University

요약

웹 문서 표준어인 XML의 등장으로 앞으로 웹 상에 수많은 XML 문서가 존재할 것이며, 이들에 대한 효율적인 검색 기법이 요구된다. 그 중 하나로 웹 상에 산재된 XML 문서들을 여과 및 통합할 수 있는 뷰의 기능이 필요하다. 뷰의 구현 기법으로는 전통적인 질의 변경 기법과 실체 뷰 기법 등이 있다. 본 논문에서는 XML 문서를 대상으로 하는 실체 뷰에 관한 것으로, XML 문서가 변경되었을 경우에 XML 실체 뷰에 대해 점진적 갱신을 지원하는 XML 실체 뷰 관리 프레임워크(framework)에서, 실체 뷰를 통한 뷰 제공의 성능을 질의 변경 기법의 성능과 비교하여 분석한 후 실체 뷰 기법이 질의 변경 기법 보다 효율적일 수 있는 조건을 구한다.

1. 서 론

인테넷이 대중화되므로 인해서 현재 웹 상에는 많은 양의 데이타가 산재해 있고 대부분의 문서가 HTML로 작성되어 있다. HTML로 작성된 문서는 구조검색을 지원하지 못하므로 웹 상에 산재되어 있는 데이타에 대한 효율적인 검색을 지원하지 못한다. 이러한 문제점을 해결하기 위해서 문서의 내용뿐만 아니라 구조를 표현할 수 있는 XML(eXtensible Markup Language)[1]이 W3C(World Wide Web Consortium) 차세대 웹 문서의 표준으로 제안되었고, 현재 XML에 대한 연구가 활발히 진행되고 있다.

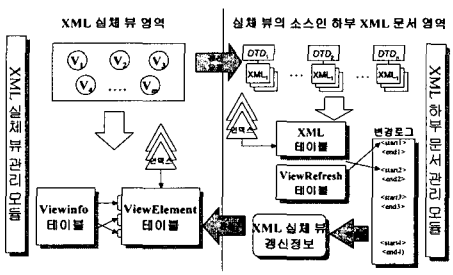
XML의 등장으로 앞으로 웹 상에 수많은 XML 문서가 존재할 것이며, 이들에 대한 효율적인 검색 기법이 요구된다. 이를 위해 웹 상에 산재된 XML 문서들을 여과 및 통합할 수 있는 뷰의 기능에 대한 연구가 필요하다.

기존의 뷰에 대한 연구는 관계 및 객체지향 데이타베이스 시스템에서 많이 수행되었다[2][3]. 뷰를 구현하는 기법에는 질의 변경(query modification)[4]과 실체 뷰(materialized view)가 있다. 질의 변경은 뷰의 정의만을 시스템 목록에 저장해 놓고, 뷰에 대한 요청이 있을 때마다 뷰의 정의에 따라 뷰 전체를 재생성하는 기법으로 뷰를 재생성하는 데 일반적으로 많은 시간이 걸릴 수 있다. 질의 변경과는 달리 실체 뷰 기법은 뷰의 내용을 실제로 저장하는 기법으로서, 사용자의 질의에 빠른 응답시간을 제공할 수 있다. 그러나 실체 뷰는 하부 데이타가 변경(update)되었을 경우에 일관성을 유지해야 하는 오버헤드가 있는데, 사용자로부터 뷰의 제공이 요청되었을 때 실체 뷰의 내용 중 변경된 부분만을 갱신하는 점진적 갱신(incremental refresh) 기법을 사용함으로써 뷰 갱신 오버헤드를 줄일 수 있다. 실체 뷰 기법에서는 뷰의 점진적 갱신을 지원하기 위해서 하부 데이타가 변경되었을 때, 데이타 변경에 대한 내용을 변경로그(update log)에 기록한다. 이때, 하부 데이타의 수정(modify)뿐만 아니라 삽입 및 삭제도 뷰에 영향을 줄 수 있기 때문에 변경로그에는 데이타 변경에 대한 모든 정보가 기록되어야 한다.

본 논문에서는 XML 문서를 대상으로 하는 실체 뷰에 관한 것으로, XML 문서가 변경되었을 경우에 XML 실체 뷰에 대해 점진적 갱신을 지원하는 XML 실체 뷰 관리 프레임워크(framework)에서, 실체 뷰를 통한 뷰 제공의 성능을 질의 변경 기법의 성능과 비교하여 분석한 후 실체 뷰 기법이 질의 변경 기법 보다 효율적일 수 있는 조건을 구한다.

본 논문의 구성은 다음과 같다. 2절에서는 XML 실체 뷰 관리 프레임워크에 대해 기술하고, 3절에서는 실체 뷰와 질의 변경 기법 간의 성능을 비교, 분석한다. 마지막으로 4절에서 결론을 맺는다.

2. XML 실체 뷰 관리 프레임워크



(그림 1) XML 실체 뷰 관리 프레임워크

XML 실체 뷰 관리 프레임워크의 구성요소들을 그림으로 나타내면 (그림 1)과 같다. XML 문서의 저장소는 XML 실체 뷰의 소스인 하부 XML 문서 영역(이하, 하부 영역)과 XML 실체 뷰 영역(이하, 뷰 영역)으로 구분되고, 이들 영역은 XML 하부 문서 관리 모듈과 XML 실체 뷰 관리 모듈에 의해 각각 관리된다. 하부 영역에는 다수의 DTD, XML 문서를 저장하는 XML 테이블과 인덱스, XML 실체 뷰의 점진적 갱신을 지원하

* 본 연구는 한국과학재단 목적 기초 연구(2000-1-30300-001-3) 지원으로 수행되었음.

기 위한 XML 문서 변경로그(update log), 각 XML 실체 뷰의 갱신을 지원하기 위한 뷰 갱신 테이블이 저장된다. 뷰 영역에는 각 XML 실체 뷰에 대한 정보를 저장한 뷰 정보 테이블, EID(element ID) 사상(mapping)정보와 실체 뷰를 저장한 뷰 엘리먼트 테이블과 인덱스가 저장된다.

XML 실체 뷰 관리 모듈은 사용자로부터 뷰가 요청되면 해당 실체 뷰의 점진적 갱신에 필요한 XML 실체 뷰 갱신정보를 XML 저장소 관리 모듈에게 요청한다. 실체 뷰 갱신정보 요청을 받은 XML 하부 문서 관리 모듈은 변경로그를 검색해서 해당 뷰에 대한 XML 실체 뷰 갱신정보를 생성하고, 이것을 XML 실체 뷰 관리 모듈에게 전송한다. 실체 뷰 갱신정보를 전송받은 XML 실체 뷰 관리 모듈은 이를 이용해서 해당 XML 실체 뷰에 대한 점진적 갱신을 수행한 후 사용자에게 뷰를 제공한다[5].

3 성능 분석

본 절에서는 XML 실체 뷰 기법의 성능을 질의 변경 기법의 성능과 비교 분석한다. 이를 위해 먼저 3.1절에서 성능 척도를 기술하고, 3.2절에서 성능 분석에 필요한 비용 모델 파라미터를 기술한 후, 3.3절에서 두 기법 간의 성능을 비교하고, 실체 뷰 기법이 질의 변경 기법보다 효율적일 수 있는 조건을 구한다.

3.1 성능 척도

두 기법 간의 성능 비교는 중앙집중 환경과 분산 환경으로 나누어서 생각할 수 있다. 중앙집중 환경의 경우, 하부 영역과 뷰 영역은 같은 사이트에 존재한다. 따라서 두 영역 간의 데이타 전송 시 통신비용이 존재하지 않기 때문에, 뷰를 제공하기 위해 발생하는 디스크 I/O 회수(변경로그의 검색 시 발생한 I/O 회수와 뷰 갱신 시 발생하는 I/O 회수를 합한 것)가 성능의 중요한 척도가 될 수 있다. 반면에 분산 환경에서는 두 영역이 서로 다른 사이트에 존재한다. 따라서 분산 환경일 경우에는 네트워크의 트래픽이 중요한 성능의 척도가 될 수 있으므로 요청된 뷰의 갱신정보를 제공하기 위해서 하부영역에서 뷰 영역으로 전송되는 메시지의 양으로 두 기법의 성능을 비교할 수 있다.

3.2 성능 파라미터

XML 뷰 관리 및 제공의 비용을 나타내기 위해 필요한 파라미터들은 (표 1)과 같다[5].

DE_{num}은 하부 영역의 XML 테이블에 저장된 하나의 XML 문서를 구성하는 평균 엘리먼트 개수, 즉 레코드의 개수를 나타낸다. DN_{xml}은 하부 영역에 저장된 XML 문서 중에서 같은 DTD를 참조하여 생성된 XML 문서의 개수를 나타낸다. (식 1)은 하부 영역에 저장된 XML 문서 중에서 같은 DTD를 참조하는 문서들의 레코드 수 T_{xml}을 구한 것이다.

$$T_{xml} = DE_{num} \times DN_{xml} \quad \text{(식 1)}$$

$R_{interval}$은 사용자에 의해 뷰가 재요청되는 간격을 초단위로 나타낸 것이다. U_{rate}는 단위 시간에 XML 문서에 대한 변경 발생 회수를 나타낸다. $R_{interval}$, U_{rate}는 사용자가 뷰를 재요청하는 사이에 XML 문서에 발생한 변경 회수, 즉 변경로그 레코드 개수 UL_{count}를 결정하는 파라미터가 되고, 이것은 (식 2)와 같다.

$$UL_{count} = R_{interval} \times U_{rate} \quad \text{(식 2)}$$

U_{sel}은 XML 문서에 변경이 발생했을 때, DE_{num}개의 엘리먼트 중에서 몇 개의 엘리먼트가 변경되었는 지의 비율을 나타낸다. DU_{ratio}는 전체 변경로그 레코드 중에서 같은 엘리먼트를 중복 수정한 변경로그 레코드의 비율을 나타낸다. DU_{avg_nt}는 중복 수정된 엘리먼트의 평균 중복 수정 회수를 나타낸다. UL_{count}와 DU_{ratio}, DU_{avg_nt}를 이용하여 중복 수정된 엘리먼트의 개수 DU_{ele_count}는 (식 3)과 같이 구할 수 있다.

(표 1) XML 뷰의 성능 파라미터

파라미터	설명
DE_{num}	하부 영역에 저장된 XML 문서의 평균 엘리먼트 개수
DN_{xml}	하부 영역에 저장된 XML 문서 중에서 같은 DTD를 참조하는 XML 문서 개수
$R_{interval}$	뷰 재요청 간격 (단위:초)
U_{rate}	단위 시간당 XML 문서 변경 발생 횟수 (회/초)
U_{sel}	XML 문서 변경 시 영향을 받는 엘리먼트 비율(단위:%)
DU_{ratio}	전체 변경로그 레코드 중에서 중복 수정 비율(단위:%)
DU_{avg_nt}	같은 엘리먼트의 평균 중복 수정 횟수
RU_{ratio}	하부 영역에 저장된 XML 문서의 구조 변경 발생 비율 (단위:%)
VR_{rate}	실체 뷰의 생성조건을 만족하는 XML 문서의 비율 (단위 :%)
VE_{num}	실체 뷰의 평균 엘리먼트 개수
BF_{ulog}	변경로그의 블록킹 팩터(blocking factor)
$BF_{element}$	XML 테이블의 블록킹 팩터
E_{avg_size}	엘리먼트의 평균 크기 (단위:byte)
UL_{avg_size}	변경로그 레코드의 평균 크기 (단위:byte)

$$DU_{ele\_count} = UL_{count} \times \frac{DU_{ratio}}{100} \times \frac{1}{DU_{avg\_nt}} \quad \text{(식 3)}$$

중복 수정된 엘리먼트를 고려할 때, 실제로 수정된 엘리먼트 개수는 (식 4)와 같다.

$$EN_{updated} = UL_{count} \times (1 - \frac{DU_{ratio}}{100}) + DU_{ele\_count} \quad \text{(식 4)}$$

RU_{ratio}는 XML 문서 변경 중에서 실체 뷰의 구조변경을 유발하는 비율을 나타낸다. 즉, XML 문서 변경 중에서 엘리먼트의 삽입 혹은 삭제 비율을 의미한다. VR_{rate}는 실체 뷰의 조건을 만족하는 XML 문서의 비율을 나타낸다. 또한, VR_{rate}는 전체 변경로그 레코드 중에서 특정 뷰에 영향을 주는 변경로그 레코드의 비율을 결정한다. VE_{num}은 실체 뷰를 구성하는 엘리먼트 개수를 나타낸다. BF_{ulog}는 변경로그 레코드의 블록킹 팩터를 나타낸다. $BF_{element}$는 XML 테이블의 블록킹 팩터를 나타낸다. 이것은 BF_{ulog}와 유사한 의미를 갖는다. E_{avg_size}는 엘리먼트의 평균 크기를 나타내며, UL_{avg_size}는 변경로그 레코드의 평균 크기를 나타낸다.

3.3 질의 변경과 점진적 갱신 기법의 성능비교

본 절에서는 질의 변경과 점진적 갱신 기법의 성능을 중앙집중 환경과 분산 환경에 대해 각각 비교한다.

3.3.1 중앙집중 환경

본 절에서는 중앙집중 환경에서 뷰 제공을 위해서 질의 변경 기법과 실체 뷰 기법을 사용할 때 발생하는 디스크 I/O 회수를 비교하고, 실체 뷰 기법이 질의 변경 기법보다 효율적일 수 있는 조건을 구한다. 이전 절의 식들과 비용 모델 파라미터를 이용하면 뷰 재생성과 점진적 갱신 시 발생하는 디스크 I/O 회수를 구할 수 있다.

사용자가 뷰를 요청하면 질의 변경 기법에서는 해당 뷰를 재생성한다. 이때, 해당 뷰의 뷰 정의를 만족하는 문서의 엘리먼트를 찾기 위해서 하부 영역의 XML 테이블에 저장된 모든 레코드들을 검색하게 된다. 질의 변경에서 뷰를 재생성하는 데 필요한 디스크 I/O 회수 $R_{QM_I/O}$는 (식 5)와 같다.

$$R_{QM\_I/O} = \left\lceil \frac{T_{xml}}{BF_{element}} \right\rceil \quad \text{(식 5)}$$

실체 뷰는 사용자로부터 뷰에 대한 요청이 발생하면 사용자에게 뷰를 제공하기 전에 뷰의 일관성 유지를 위해서 점진적 갱신을 수행한다. 이때 실체 뷰는 XML 테이블 전체를 검색하는

질의 변경과는 다르게 해당 뷰에 영향을 주는 하부 데이타 변경 사항만을 변경로그에서 검색하여 뷰의 일부분만을 갱신한 후에 사용자에 뷰를 제공한다. 또한 뷰 갱신 시 엘리먼트의 삽입 혹은 삭제가 발생하면 뷰의 구조변경이 유발되므로, 이에 따른 추가 비용이 요구된다. 그러므로 실체 뷰를 점진적으로 갱신하는 비용 $R_{MV_I/O}$은, 실체 뷰 갱신 정보를 생성하는 비용과 인덱스 접근 비용을 무시할 경우, (식 6)과 같이 정리된다.

$$R_{MV\_I/O} = \text{변경로그 검색 시 발생하는 디스크 } I/O \text{ 회수}(UL_{I/O}) + \text{실체 뷰 갱신 시 발생하는 디스크 } I/O \text{ 회수}(R_{I/O}) \quad (식\ 6)$$

(식 6)에서 변경로그 검색 시 발생하는 디스크 I/O 회수 $UL_{I/O}$는 (식 7)와 같다. 이 식에서 디스크 I/O 회수는 BF_{ulog}의 값이 커질수록 감소한다.

$$UL_{I/O} = \left\lceil \frac{UL_{count}}{BF_{ulog}} \right\rceil \quad (식\ 7)$$

실체 뷰 갱신 시 발생하는 디스크 I/O 회수 $R_{I/O}$는 (식 8)과 같이 엘리먼트 내용 수정 시 발생하는 디스크 I/O 회수와 뷰 구조 변경 시 발생하는 디스크 I/O 회수를 합한 것과 같다.

$$R_{I/O} = \text{엘리먼트 수정 시 발생하는 디스크 } I/O\text{횟수}(RE_{modified}) + \text{뷰 구조 변경 시 발생하는 디스크 } I/O\text{횟수}(V_{restruct}) \quad (식\ 8)$$

(식 8)에서 엘리먼트 수정 시 발생하는 디스크 I/O 횟수 $RE_{modified}$는 변경로그 레코드 중에서 실제 변경된 엘리먼트 개수 $EN_{updated}$와 VR_{rate}를 이용하여 구할 수 있고, 이것은 (식 9)와 같이 나타낼 수 있다. VR_{rate}는 전체 하부 영역의 XML 문서 변경 중에서 뷰에 영향을 주는 변경 비율을 결정한다. 그러므로 $RE_{modified}$는 VR_{rate}에 따라서 증가 혹은 감소한다.

$$RE_{modified} = EN_{updated} \times \frac{VR_{rate}}{100} \quad (식\ 9)$$

뷰의 구조변경 시 발생하는 디스크 I/O 회수는 엘리먼트의 삽입 혹은 삭제 수에 크게 영향을 받는다. 그러므로 엘리먼트의 삽입 혹은 삭제 수를 알아내는 것이 중요하다. (식 10)은 XML 문서 변경 시 엘리먼트의 삽입 혹은 삭제 수 $E_{I/D}$를 나타낸다. 이전 절에서 설명한 것과 같이 RU_{ratio}는 실체 뷰의 구조변경 발생 비율을 나타낸다. 그러므로 $E_{I/D}$는 RU_{ratio}에 따라서 증가 혹은 감소한다.

$$E_{I/D} = RE_{modified} \times \frac{RU_{ratio}}{100} \quad (식\ 10)$$

(식 10)과 같이 엘리먼트의 삽입 혹은 삭제 회수 $E_{I/D}$를 구하면, 구조변경 시 발생하는 디스크 I/O 회수 $V_{restruct}$는 (식 11)와 같이 나타낼 수 있다.

$$V_{restruct} = \frac{VE_{num}}{2} \times E_{I/D} \quad (식\ 11)$$

뷰의 구조변경은 실체 뷰에 저장된 엘리먼트의 EID를 수정하는 것이다. 만약 가장 마지막 엘리먼트 이후에 새로운 엘리먼트가 삽입되면 기존 EID의 수정은 발생하지 않지만, 루트 엘리먼트의 첫 번째 자식 엘리먼트 이전에 삽입되면 모든 EID의 수정을 유발하게 된다. 본 논문에서는 '$VE_{num}/2$'를 구조변경 시 변경해야 하는 평균 엘리먼트의 개수로 가정한다.

하부 영역에 저장된 변경로그 레코드의 개수를 'α', 중복 수정된 엘리먼트 개수를 'β', 뷰에 영향을 주는 하부 영역 XML의 문서 변경을 'γ', 뷰 구조변경 시 발생하는 평균 I/O 회수를 'δ'라 했을 경우에, 중앙집중 환경에서 두 기법 간의 성능 차이 $\Delta I/O$는 (식 12)와 같이 나타낼 수 있다.

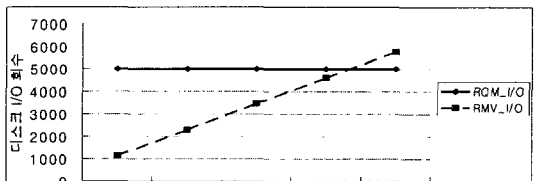
$$\begin{aligned}\Delta_{I/O} &= R_{QM\_I/O} - R_{MV\_I/O} \\ &= R_{QM\_I/O} - (UL_{I/O} + RE_{modified} + V_{restruct}) \\ &= R_{QM\_I/O} - \left(\frac{R_{interval} \times U_{rate}}{BF_{ulog}} + \left(\left(\alpha - \frac{\alpha \times DU_{ratio}}{100} + \beta\right) \times \frac{VR_{rate}}{100}\right) + \left(\delta \times \gamma \times \frac{RU_{ratio}}{100}\right)\right)\end{aligned} \quad (식\ 12)$$

(식 12)에서 $UL_{I/O}$는 변경로그 레코드 검색 시 발생하는 디스크 I/O 회수를 나타낸다. $UL_{I/O}$는 변경로그 레코드의 개수에 영향을 받으므로 $R_{interval}$과 U_{rate}가 커짐에 따라서 $UL_{I/O}$는 증가한다. 그러나 BF_{ulog}에 따라서 $UL_{I/O}$는 줄어들게 된다. $RE_{modified}$는 뷰의 엘리먼트 수정 시 발생하는 디스크 I/O 회수를 나타낸다. 이것은 중복 수정된 엘리먼트 비율 DU_{ratio}와 하부 영역의 XML 문서 수정 중에서 뷰에 영향을 미치는 비율 VR_{rate}에 영향을 받는다. 즉, DU_{ratio}에 반비례, VR_{rate}에 비례하게 증가한다. $V_{restruct}$는 뷰 구조변경 시 발생하는 I/O 회수를 나타낸다. 이것은 RU_{ratio}에 반비례하게 증가한다.

(식 12)를 이용하면 사용자에게 뷰를 제공할 때, 하부 영역에 저장된 XML 문서의 변경 상황에 따라서 질의 변경의 뷰 재생성과 실체 뷰의 점진적 갱신 기법 중에서 어느 기법이 더 효율적인 지를 알 수 있다. $\Delta I/O > 0$이면 실체 뷰가 질의 변경보다 우수한 성능을 내고, $\Delta I/O < 0$이면 질의 변경이 더 우수한 성능을 낸다. (식 12)를 구성하는 파라미터는 모두 하부 영역 내에서 관리되는 것들이므로 사용자가 뷰를 요청하는 시점에 이들 식을 계산하여 실체 뷰 기법과 질의 변경 기법 중 더 효율적인 것을 선택할 수 있다. 실체 뷰의 점진적 갱신 비용에 가장 영향을 많이 주는 것이 뷰의 구조변경이다. 그러므로 하부 데이타의 변경 중에서 엘리먼트 값의 수정이 많더라도 엘리먼트의 삽입 혹은 삭제가 거의 없다면 실체 뷰가 더 우수한 성능을 낼 수 있다. 다음은 구조변경이 없을 경우, 단위 시간당 XML 문서 변경 발생 회수 U_{rate}에 따른 질의 변경 기법과 실체 뷰 기법의 성능 비교 실험 예로서, 실험을 위해서 설정된 파라미터는 다음과 같다.

DE_{num}	DN_{xml}	$R_{interval}$	U_{rate}	DU_{ratio}	DU_{avg_nt}	VR_{rate}	BF_{ulog}	$BF_{element}$
100	100	60	2	50	2	30	10	2

(그림 2)에 나타난 것과 같이 U_{rate}가 증가함에 따라서 실체 뷰 기법에서 발생하는 디스크 I/O 회수가 증가하는 것을 알 수 있다.

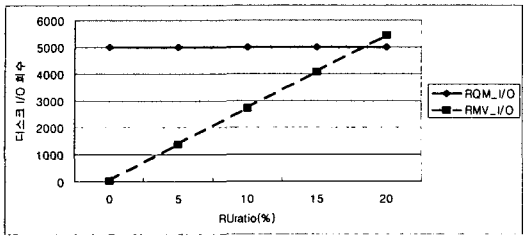


(그림 2) U_{rate}에 따른 디스크 I/O 회수

다음은 구조변경이 있을 경우, 하부 영역에 저장된 XML 문서의 구조 변경 발생 비율 RU_{ratio}에 따른 질의 변경 기법과 실체 뷰 기법의 성능 비교 실험 예로서, 설정된 파라미터는 다음과 같다.

DE_{num}	DN_{xml}	$R_{interval}$	DU_{ratio}	DU_{avg_nt}	RU_{ratio}	VR_{rate}	BF_{ulog}	$BF_{element}$
100	100	60	10	2	0	30	10	2

(그림 3)에 나타난 것과 같이 RU_{ratio}가 증가함에 따라서 실체 뷰 기법에서 발생하는 디스크 I/O 회수가 증가하는 것을 알 수 있다. 또한 구조변경의 양이 많지 않더라도 실체 뷰 기법의 성능에 크게 영향을 미치는 것을 알 수 있다.



(그림 3) RU_{ratio}에 따른 디스크 I/O 회수

(그림 2)와 (그림 3)에서 나타난 것처럼 실체 뷰의 점진적 갱신 비용에 주로 영향을 미치는 것은 단위 시간당 XML 문서 변경 발생 회수와 뷰의 구조변경이며, 그 중에서도 구조변경이 실체 뷰 기법의 성능에 더 많은 영향을 주게된다. 그러므로 하부 XML 문서의 변경 중에서 엘리먼트 값의 수정이 많더라도 엘리먼트의 삽입 혹은 삭제가 거의 없다면 실체 뷰가 더 우수한 성능을 낼 수 있다.

3.3.2 분산 환경

본 절에서는 분산 환경에서 뷰 제공을 위해서 질의 변경 기법과 실체 뷰 기법을 사용할 때 뷰 갱신을 위해서 전송되는 메시지의 양을 비교하고, 실체 뷰 기법이 질의 변경 기법보다 효율적일 수 있는 조건을 구한다. 질의 변경에서 뷰 재생성 시 전송되는 메시지의 양 M_{QM}은 (식 13)과 같다.

$$M_{QM} = VE_{num} \times E_{size} \quad \text{(식 13)}$$

사용자가 뷰를 요청했을 때 질의 변경 기법의 경우 재생성된 뷰 전체를 제공한다. 그러므로 전송되는 메시지의 양은 뷰를 구성하는 레코드들의 크기와 같다. 이와는 다르게 실체 뷰에서 뷰 갱신 시 전송되는 메시지의 양 M_{MV}는 (식 14)와 같다.

$$M_{MV} = EN_{updated} \times UL_{size} \quad \text{(식 14)}$$

실체 뷰는 사용자에게 뷰에 대한 요청이 있을 때, 뷰를 제공하기 전에 일관성 유지를 위해서 점진적 갱신을 수행한다. 이를 위해서 XML 실체 뷰 갱신정보, 즉 하부 데이타 변경 중에서 해당 뷰에 영향을 미치는 변경 정보만을 전송받는다. 그러므로 실체 뷰의 점진적 갱신 시에 전송되는 메시지의 양은 $RE_{modified}$를 구성하는 레코드의 양과 같다.

변경로그 레코드 개수를 'α', 중복 수정된 엘리먼트 개수를 'β'라 하고, 분산 환경에서 E_{size}와 UL_{size}가 같다고 하면, 두 기법 간의 성능 차이 ΔM은 (식 15)와 같이 나타낼 수 있다.

$$\Delta_M = VE_{num} - RE_{modified} = VE_{num} - ((\alpha - \frac{\alpha \times DU_{ratio}}{100} + \beta) \times \frac{VR_{rate}}{100}) \quad \text{(식 15)}$$

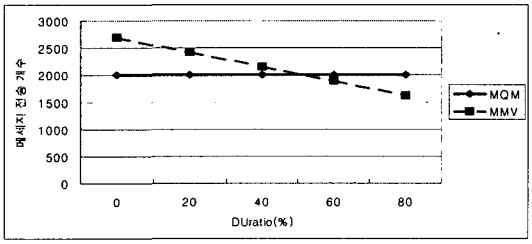
(식 15)에서 $RE_{modified}$는 실체 뷰에 영향을 미치는 변경로그 레코드 수, 즉 점진적 갱신을 위해 전송되는 XML 실체 뷰 갱신 정보의 양과 같다. $RE_{modified}$는 엘리먼트 중복 수정 비율 DU_{ratio}와 하부 데이타 수정 중에서 뷰에 영향을 미치는 비율 VR_{rate}에 영향을 받는다. 즉, DU_{ratio}에 반비례, VR_{rate}에 비례하여 증가한다.

(식 15)를 이용하면 사용자에게 뷰를 제공할 때, 하부 데이타의 변경 상황에 따라서 질의 변경의 뷰 재생성과 실체 뷰의 점진적 갱신 기법 중에서 어느 기법이 더 효율적인 지를 알 수 있다. $\Delta M > 0$이면 실체 뷰가 질의 변경보다 적은 양의 메시지를 전송하므로 실체 뷰 기법을 선택하는 것이 효율적이다. 반면에 $\Delta M < 0$이면 실체 뷰를 폐기하고 질의 변경 기법을 선택하는 것이 효율적이다. (식 15)를 구성하는 파라미터도 (식 12)와 마찬가지로 모두 하부 영역 내에서 관리되는 것들이므로 사용자가 뷰를 요청하는 시점에 이들 식을 계산하여 실체 뷰 기법과 질의 변경 기법 중 더 효율적인 것을 선택할 수 있다.

다음은 분산 환경에서 중복 수정 비율 DU_{ratio}따른 질의 변경 기법과 실체 뷰 기법의 성능 비교 실험 예로서, 설정된 파라미터는 다음과 같다.

DE_{num}	DN_{xml}	$R_{interval}$	U_{rate}	DU_{avg_nt}	RU_{ratio}	VR_{rate}	VE_{num}
100	100	60	150	2	0	30	2000

이 실험에서 엘리먼트의 평균 크기 E_{size}와 변경로그 레코드의 평균 크기 UL_{size}가 같다고 가정하면, 분산 환경에서 질의 변경 기법과 실체 뷰 기법의 성능은 전송되는 메시지의 개수로 비교할 수 있다.



(그림 4) DU_{ratio}에 따른 메시지 전송 개수

(그림 4)에 나타난 것과 같이 DU_{ratio}가 증가함에 따라서 실체 뷰 기법에서 전송되는 메시지의 개수가 감소하는 것을 알 수 있다. 그러므로 같은 엘리먼트에 대한 중복 수정이 빈번한 곳에서는 실체 뷰 기법이 더 좋은 성능을 낼 수 있다.

4. 결 론

XML은 문서의 구조정보를 나타낼 수 있으므로 기존의 관계 및 객체지향 데이타베이스 등에서 연구되었던 실체 뷰 관리 기법과는 다른 XML 실체 뷰 관리 기법이 요구된다. 본 논문에서는 XML 문서 변경 시에, XML 실체 뷰에 대해 점진적 갱신을 지원하는 XML 실체 뷰 관리 프레임워크에서 실체 뷰 기법의 성능을 질의 변경 기법의 성능과 비교하여 분석하였다. 디스크 I/O 회수는 중앙집중 환경에서 성능평가의 척도로서, 중복 수정 비율이 높고 뷰의 구조 변경이 적을 경우에 실체 뷰가 더 좋은 성능을 나타냈다. 전송되는 메시지 양은 분산 환경에서 성능평가의 척도로서, 중복 수정 비율이 높을수록 실체 뷰가 더 좋은 성능을 나타냈다.

참고문헌

[1] T. Bray et al., "Extensible Markup Language(XML) 1.0," http://www.w3.org/TR/1998/REC-xml-19980210, 1998.

[2] B. Lindsay et al., "A Snapshot Differential Refresh Algorithm," Proc. SIGMOD, Jun. 1986, pp. 53-60.

[3] N. Roussopoulos, "An Incremental Access Method for ViewCache: Concept, Algorithms, and Cost Analysis," Trans. on Database Systems, Vol. 16, No. 3, Sep. 1991, pp. 535-563.

[4] M. Stonebraker., "Implementation of Integrity Constraints and Views by Query Modification," Proc. SIGMOD, 1975, pp. 65-78.

[5] 임재국, "XML 문서의 실체 뷰를 위한 점진적 갱신," 석사 학위 논문, 중앙대학교 컴퓨터공학과, 2001. 2.