무선 방송을 위한 효과적인 XML 스트리밍
(Effective Streaming of XML Data for Wireless Broadcasting)

박 준 표 † 박 창 섭 †† 정 연 돈 †††
(Jun Pyo Park) (Chang-Sup Park) (Yon Dohn Chung)

요 약 방송 기법을 통한 데이타의 전달은 대역폭 활용의 이점과 에너지 효율성, 확장성으로 인해 무선 모바일 환경에서 효과적인 방법으로 알려져 있다. 본 논문에서는 무선 방송 환경에서 트리 기반의 색인 구조를 사용하기 때문에 이동 사용자의 접근 시간이 증가하는 "질의 처리의 지연 문제"를 다루고 있다. 본 논문에서는 "질의 처리의 지연 문제"를 해결하기 위한 분산 색인 구조와 함께 XML 데이타의 에너지 및 접근 시간 효율적인 방송을 위한 클러스터링 방법을 제안한다. 먼저 분산 색인 구조를 구현하기 위해 엘리먼트의 태그 이름과 애트리뷰트, 그리고 텍스트와 색인 정보를 포함하고 있는 DIX 노드 구조를 제안한다. 모바일 사용자는 DIX 노드에 포함되어 있는 색인 정보를 통해 무선 XML 스트림에서 보다 짧은 지연 시간만으로 원하는 정보에 접근할 수 있다. 또한, 질의 처리를 위한 탐색 범위를 한정시킴으로써 질의 처리에 소요되는 접근 시간과 튜닝 시간을 단축시킬 수 있는 클러스터링 정책을 제안한다. 성능 평가 실험을 통해 제안 방법이 기존의 XML 데이타 방송 기법들에 비해 우수함을 확인할 수 있다.

키워드 : 무선방송, XML 스트리밍, 분산 색인

Abstract In wireless and mobile environments, data broadcasting is recognized as an effective way for data dissemination due to its benefits to bandwidth efficiency, energy-efficiency, and scalability. In this paper, we address the problem of delayed query processing raised by tree-based index structures in wireless broadcast environments, which increases the access time of the mobile clients. We propose a novel distributed index structure and a clustering strategy for streaming XML data which enable energy and latency-efficient broadcast of XML data. We first define the DIX node structure to implement a fully distributed index structure which contains tag name, attributes, and text content of an element as well as its corresponding indices. By exploiting the index information in the DIX node stream, a mobile client can access the wireless stream in a shorter latency. We also suggest a method of clustering DIX nodes in the stream, which can further enhance the performance of query processing over the stream in the mobile clients. Through extensive performance experiments, we demonstrate that our approach is effective for wireless broadcasting of XML data and outperforms the previous methods.

Key words : wireless broadcast, XML streaming, distributed indexing

· 본 연구는 2단계 BK21 사업의 지원을 받았음
· 본 연구는 경기도지역협력연구센터사업[GGA0801-45700, U-city보안감시기술협력센터]의 지원을 받았음.
· 이 논문은 Korean DataBase Conference 2008에서 '에너지 및 접근 시간 효율적인 무선 XML 스트림 색인 기법'의 제목으로 발표된 논문을 확장한 것임

† 학생회원 : 고려대학교 컴퓨터학과
jp_park@korea.ac.kr
†† 정 회 원 : 수원대학교 인터넷정보공학과
cspark0@gmail.com
††† 종신회원 : 고려대학교 컴퓨터학과 교수
ydchung@korea.ac.kr
(Corresponding author임)

논문접수 : 2008년 6월 23일
심사완료 : 2008년 11월 28일

정보과학회논문지: 데이타베이스 제36권 제1호(2009.2)

1. 서 론

무선 통신 기술의 발달에 따라, 커뮤니케이션 패러다임은 점차 사용자가 이동하면서 자신의 휴대용 단말기(laptop, PDA, cellular phone) 등을 통해 데이타를 주고받는 무선 모바일 컴퓨팅으로 전환되고 있다. 한편, 방송 기법을 통한 데이타의 전달은 대역폭 활용의 이점과 에너지 효율성, 확장성으로 인해 무선 모바일 환경에서 효과적인 방법으로 알려져 있다[1,2]. 서버는 방송 채널을 통해 데이타를 방송하고, 이동 사용자들은 방송 채널을 통해 데이타를 수신할 수 있다. 서버 입장에서 방송 기법을 통한 데이타의 전달은 한정된 네트워크 대역폭을 효율적으로 활용하며, 이동 사용자의 수가 늘더라도 추가적인 비용이 들지 않는다. 한편, 이동 사용자의 입장에서는 에너지 소모가 많은 "송신" 작업 없이 "수신"만으로 데이타에 접근할 수 있는 이점이 있다.

무선 방송 환경에서 이동 사용자들은 한정된 배터리 자원의 휴대용 단말기를 사용하므로, 에너지 보존을 위해 데이타의 선택적인 수신이 요구된다. 동시에 이동 사용자에게 빠른 응답을 주기 위해 전체 질의 처리 시간 또한 최소화되어야 한다. 전자를 에너지 효율성, 후자를 접근 시간 효율성이라 일컫는다[3]. 방송 기법을 통한 데이타 전달에 있어, 이동 사용자의 에너지 효율성과 접근 시간 효율성을 측정하기 위해 각각 튜닝 시간(tuning time)과 접근 시간(access time)이 성능 측정의 척도로 활용된다[2]. 튜닝 시간은 이동 사용자가 방송 채널을 수신하며 데이타를 읽은 시간들의 합이다. 이동 사용자는 방송 스트림을 수신할 때, 많은 양의 에너지를 소모한다. 따라서 튜닝 시간은 에너지 효율성의 기준으로 활용되고 있다. 접근 시간은 이동 사용자의 질의가 시작한 시점부터, 모든 필요한 데이타를 수신 받은 시점까지의 소요 시간이다.

본 논문에서는, 무선 방송 환경에서 XML 데이타의 효율적인 스트리밍 및 접근 방법에 대해 다루고 있다. XML(eXtensible Markup Language)[4]는 데이타의 표현 및 전달을 위한 표준으로서 다양한 분야에서 활용되고 있는 언어이다. 그림 1은 간단한 XML 문서의 예(a)와 이를 트리 형식으로 표현한 그림(b)을 보여주고 있다. 이 XML 문서는 본 논문 전체에 걸쳐 예제로 활용된다.

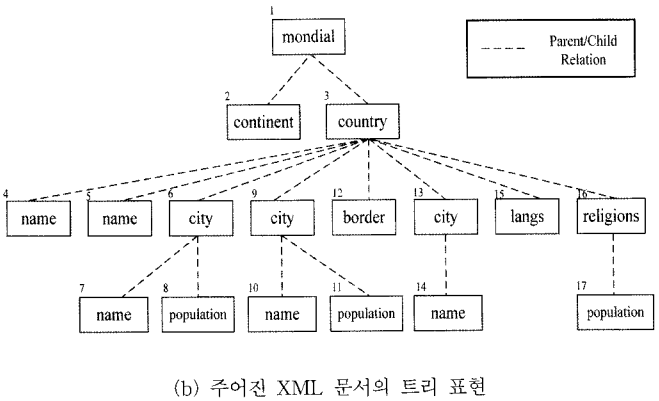
지금까지 XML 데이타의 스트리밍 및 XML 데이타 스트림의 방송 기법에 대해 여러 연구들이 이루어져 왔다[5-7]. 그러나, 기존의 연구들은 에너지 효율성 문제를 다루지 않았거나[5], 트리 기반의 색인 구조를 사용하기 때문에 생기는 질의 처리의 지연 문제가 있다[6,7].

본 논문에서는, 에너지 및 접근 시간 효율적인 XML 데이타 스트리밍 방법을 제안한다. 이를 위해, 분산 색인 구조를 위한 XML 데이타 스트림의 기본 구성 요소, 즉 DIX 노드를 정의한다. DIX 노드는 위치 경로 정보를 포함하고 있으므로 이동 사용자가 스트림의 임의의 위치에서 질의 처리를 시작할 수 있으며, 모든 질의 처리는 하나의 방송 주기 이내에 완료된다. 또한 질의 처리 성능을 더욱 향상시키기 위한 클러스터링 구조를 제안한다. 같은 깊이를 가진 XML 데이타들을 클러스터링함으로써 이동 사용자는 질의의 탐색 범위를 한정할 수 있다. 따라서 질의 처리의 에너지 및 접근 시간 효율성이 향상될 수 있다.

이 논문의 구성은 다음과 같다. 관련 연구 및 연구 동기는 2장에 기술되어 있다. 3장에서는 분산 색인 구조를 포함한 XML 데이타 스트림의 구조를 정의하고 스트림의 생성 방법을 설명한다. 4장에서는 제안 기법에 의해 생성된 XML 데이타 스트림에 대한 질의 처리 알고리즘에 대해 설명한다. 5장에서는 XML 엘리먼트의 깊이

```
01:<mondial>
02:   <continent ID="f0_119" NAME="Europe" />
03:   <country ID="f0_136" NAME="Albania" >
04:     <name>Albania</name>
05:     <name>keshilli</name>
06:     <city ID="f0_1461" COUNTRY="f0_136" >
07:         <name>Tirana</name>
08:         <population YEAR="87">172010</population>
09:     </city>
10:     <city ID="f0_36499" COUNTRY="f0_137" >
11:         <name>Shkoder</name>
12:         <population YEAR="87">192000</population>
13:     </city>
14    <border> ... </border>
15:     <city>Shkoder
16:         <name>...</name>
17:     </city>
18:     <langs>Eng</langs>
19:     <religions>
20:         <population>120130</population>
21:     </religions>
22:   </country>
23:   ...
24:</mondial>
```

(a) 간단한 XML 문서의 예



(b) 주어진 XML 문서의 트리 표현

그림 1 간단한 XML 문서의 예와 이에 대한 트리 표현

에 기반한 클러스터링 정책에 대해 설명한다. 클러스터된 스트림의 생성과 이에 따른 질의 처리 방법을 위해 수정된 알고리즘들 또한 이 장에 기술되어 있다. 다양한 실험을 통한 성능 측정 결과는 6장에 기술되어 있다. 마지막으로 7장에서 논문의 결론을 도출한다.

2. 관련 연구 및 연구 동기

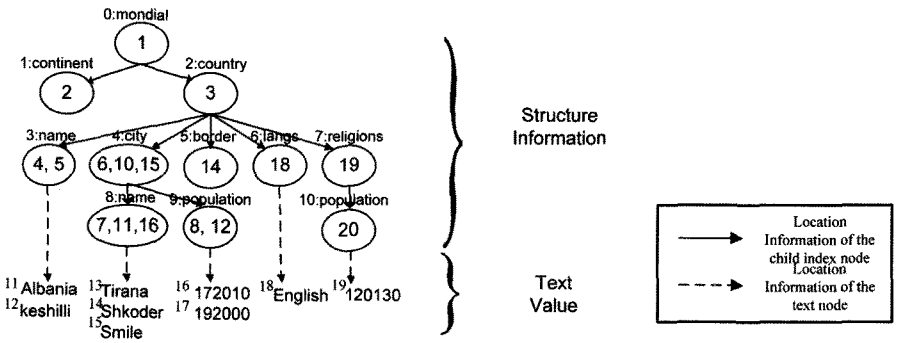
2.1 관련 연구

지금까지 무선 환경에서 XML 데이타의 스트리밍과 질의 처리 및 압축 방법에 대해 여러 기법들이 제안되었다[5,8,9]. 특히, Xstream[2]은 주어진 XML 문서의 의미적, 구조적(semantic, structural) 특징들에 기초한 단편화(fragmentation)와 패킷화(packetizing) 기법을 제안하고 있는 미들웨어 시스템이다. 그러나 이러한 연구들은 에너지 효율성을 고려하지 않았으며, 무선 방송 환경을 지원하지 않고 있다.

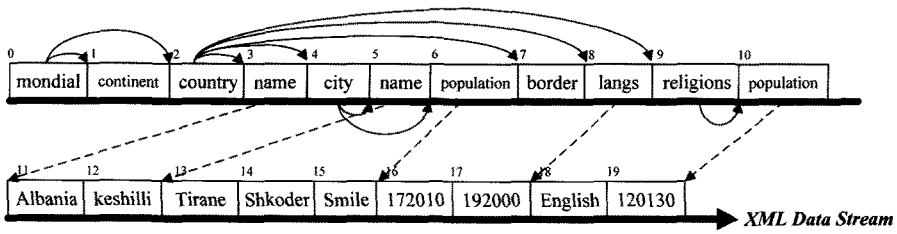
무선 방송 환경에서 이동 사용자의 에너지 효율성 문제와 관련하여, [7]은 무선 방송 환경에서 경로 요약(path summary) 기법을 활용하여 에너지 효율적인 XML 스트림의 질의 처리 방법을 제안하였다. 이 방법은 XML 문서의 구조적 정보와 텍스트를 분리하여 색인정보로 활용하며, 중복된 레이블 경로(label path)를 생략함으로써 전체 스트림의 크기를 줄였다. 그림 2는 이 방법에 의해 생성된 XML 데이타 트리(a)와 XML 데이타 스트림의 구조(b)를 보여주고 있다. 그림 2와 같이 모든 중복된 레이블 경로는 요약되었으며, 각 인덱스 노드는 자손 노드들과 텍스트에 대한 위치 정보를 포함하고 있다. 그러나 이 기법은 엘리먼트의 순서를 고려하지 않고, 후손 축(descendant axe)이나 프레디케이트(predicate condition) 등의 다양한 XPath 특징을 지원하지 않고 있다.

[6]에서는 S-node라고 불리우는 XML 데이타 방송을 위한 스트리밍 단위의 정의를 통해 에너지 효율적인 XML 스트리밍 방법들과 질의 처리 알고리즘들을 제안하고 있다. 이 방법은 XML 문서의 구조적 특징을 이용하여 각각의 S-node가 두 개의 다른 형제 노드 주소, 즉 같은 태그 이름을 가진 형제 노드(same-tag sibling link)와 다른 태그 이름을 가진 형제 노드(different-tag sibling link)에 대한 주소를 가진 트리 형식의 색인 구조를 구성한다. 이 색인 정보를 통해 이동 사용자는 불필요한 데이타의 수신을 생략하고 원하는 데이타만 선택적으로 수신한다. 그림 3은 S-node의 트리 표현(a)과 XML 데이타 스트림에 대한 질의 처리 예(b)를 보여주고 있다.

예를 들어, 이동 사용자가 "/mondial/country/city/name"과 같은 XPath 질의를 하였다고 가정하자. 질의의 결과 값을 찾기 위해서 이동 사용자는 문서의 루트부터 찾는 데이타까지 위치 경로(location path)에 따라 탐색한다.

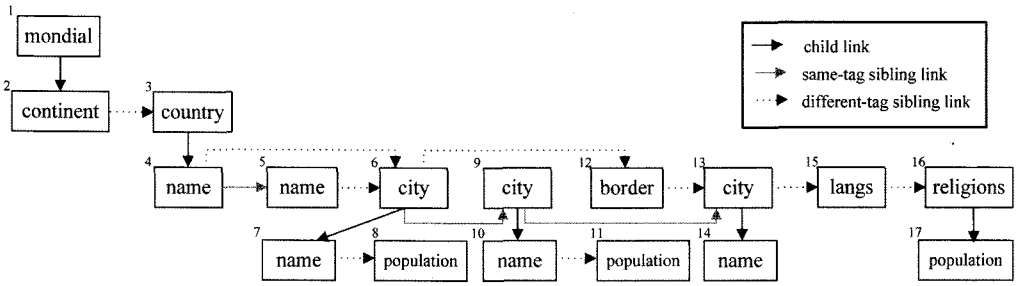


(a) XML 데이타 트리

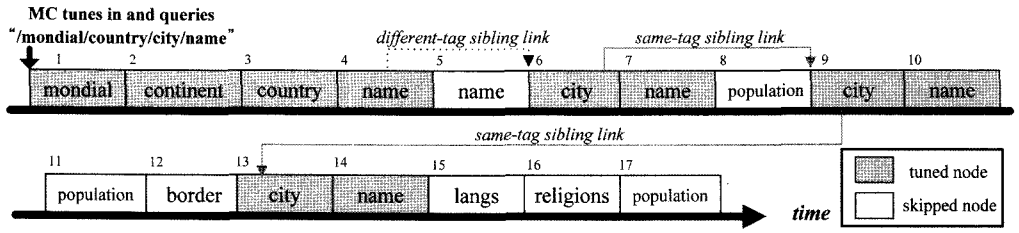


(b) XML 데이타 스트림 구조

그림 2 Path Summary 방법에 의해 생성된 XML 데이타 트리와 데이타 스트림의 구조



(a) S-node의 트리 표현



(b) XML 데이타 스트림에 대한 질의 처리

그림 3 S-node에 의해 생성된 데이타 트리와 스트림에 대한 질의 처리

즉, "mondial," "country," "city," "name"노드 순으로 탐색한다. 먼저, 이동 사용자는 1번 "mondial"부터 4번 "name"노드까지 수신한다. 이때, 4번 노드를 수신한 후에 이 노드의 태그 이름으로부터 이 노드가 질의의 위치 경로와 다르므로 different-tag sibling link를 이용하여 같은 태그 이름을 가진 5번 "name" 노드의 수신을 생략한다. 또한 질의의 위치 경로에 해당하는 6번 "city" 노드의 same-tag sibling link를 이용하여 8번 "population" 노드의 수신을 생략한다. 결론적으로, 이 예제에서 이동 사용자는 5, 8, 11, 12, 15, 16, 17번 노드의 수신을 생략한다.

2.2 연구 동기

서론에서 기술하였듯이 기존의 XML 스트리밍 기법들은 트리 기반의 색인 구조를 사용한다. 그러나 트리 기반의 색인 기법은 이동 사용자가 방송 채널에 들어오자마자 질의 처리를 시작할 수 없으므로, 무선 방송 환경에서 비효율적이다. 본 논문에서는 이를 "질의 처리의 지연 문제"라 부르기로 한다.

질의 처리의 지연 문제. 트리 기반의 색인 구조에서 질의 처리는 문서의 루트부터 찾는 데이타까지 위치 경로(location path)를 따라 이루어진다. 그러나 데이타는 시간에 따라 정해진 순서대로 방송되기 때문에 이동 사용자는 이미 방송된 데이타에 대해 역방향으로 탐색할 수 없다. 따라서, 문서의 루트가 방송된 이후에 방송 채널에 들어온 이동 사용자들은 질의 처리를 위해 다음 방송 주기까지 대기해야 한다. 이때의 대기 시간은 접근 시간을 크게 증가시키는 원인이 된다.

예제 1. 그림 4는 "질의 처리의 지연 문제"의 예를 나타내고 있다. 이동 사용자 'A'는 문서의 루트가 방송되기 이전에 방송 채널에 들어온 반면 이동 사용자 'B'와 'C'는 방송 중간에 방송 채널에 들어왔다. 이때, 이동 사용자 'A'는 즉시 질의 처리를 시작할 수 있는 반면,

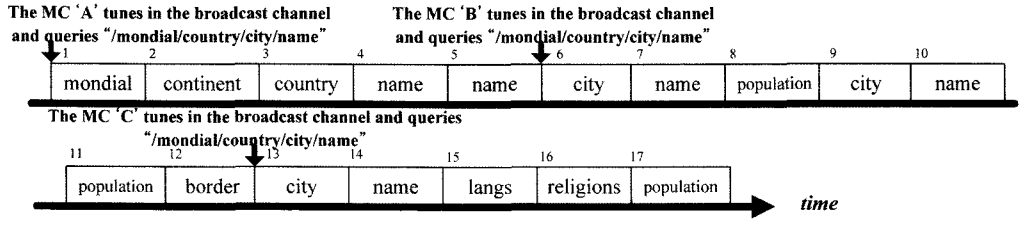


그림 4 질의 처리의 지연 문제

이동 사용자 'B'와 'C'는 다음 방송 주기까지 기다려야 한다. □

무선 방송 환경에서 이동 사용자가 질의 처리를 바로 시작하기 위해서는 루트로부터 시작하는 위치 경로 단계를 따르지 않고 질의 처리할 수 있는 방법이 필요하다. 따라서 본 논문에서는 이러한 목적을 위해 색인 정보가 XML 데이타 스트림 전체에 분포되어 있는 분산 색인 구조를 제안한다.

3. 제안하는 XML 스트리밍 방법

이 장에서는 XML 데이타 스트림의 효율적인 질의 처리를 위한 색인 구조와 스트림의 생성 방법에 대해 설명한다. 먼저 무선 XML 방송을 위한 분산 색인 구조를 구현하기 위해 DIX(Distributed Index for XML broadcast) 노드 구조를 정의한다. DIX 노드는 XML 엘리먼트의 단순화된 정보(데이타 세그먼트)와 이에 대한 색인 정보(인덱스 세그먼트)로 구성된 XML 데이타 스트림의 기본 구성 단위이다. XML 문서의 구조적인 과부하(overhead)를 줄이기 위해 SAX 파서[10]를 이용하여 태그 이름, 애트리뷰트, 그리고 텍스트만 추출하여 DIX로 이루어진 스트림을 생성한다.

3.1 DIX 노드 구조

분산 색인 구조를 구현하기 위해 XML 데이타 스트림의 기본 구성 단위가 되는 DIX 노드를 정의한다. 각 DIX 노드는 데이타 세그먼트와 인덱스 세그먼트의 두 개의 부분으로 이루어져 있다.

정의 1. DIX 노드는 다음과 같이 표현된다. DIX_i=[DS_i, IS_i], 여기서 DIX_i는 원본 XML 문서의 i번째 엘리먼트 e_i이고, DS_i와 IS_i는 각각 DIX_i의 데이타 세그먼트와 인덱스 세그먼트이다. DS_i는 e_i의 태그 이름, 애트리뷰트, 텍스트로 구성되어 있으며, IS_i는 e_i의 위치 경로 정보(Location Path Information: LPI), 유사 노드 링크(Clone node Link: CL), 외래 노드 링크(Foreign node Link: FL)로 구성된다. □

그림 5는 그림 1(a)의 예제 XML 문서에서 "name" 엘리먼트(4번째 줄)로부터 생성된 DIX 노드의 구조를 보여주고 있다. DIX 노드는 태그 이름과 애트리뷰트, 텍스트와 함께 색인 정보를 포함하고 있으므로 이동 사용자는 색인 정보와 데이타에 따로 접근할 필요가 없다. 특히, 이동 사용자는 태그이름과 LPI로부터 이 노드의 위치 경로를 파악할 수 있으므로 루트부터 질의 처리를 시작할 필요가 없다.

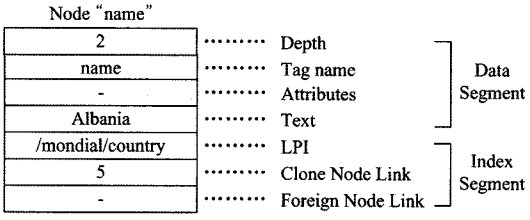


그림 5 DIX 노드 구조

3.2 위치 경로 정보

정의 2. 위치 경로 정보(Location Path Information: LPI)는 조상 엘리먼트들(즉, 루트 엘리먼트부터 부모 엘리먼트까지)의 태그 이름으로 구성된 문자열로 다음과 같이 표현될 수 있다. LPI_i="/T_0/T_1/../T_{n-1}," 여기서 T_0~T_{N-1}은 루트 엘리먼트(T_0)부터 부모 엘리먼트(T_{n-1})까지의 조상 엘리먼트들의 태그 이름이다. 또한, XML에서 경로는 자신의 태그 이름을 포함한 레이블들로 구성되므로 이 엘리먼트의 경로 $Path_i$는 다음과 같이 표현될 수 있다. $Path_i$="/T_0/T_1/../T_{n-1}/T_n." □

한편 LPI를 문자열로 전송하게 되면 XML 스트림의 크기를 매우 커지게 된다. 이 논문에서는 해시 함수를 이용하여 위치 경로 정보를 4 바이트 길이의 비트열(bit string)으로 부호화하여 전송한다. 해쉬 함수는 다양한 길이의 문자열을 고정된 짧은 길이의 출력으로 변환하므로 스트림의 크기를 줄이는 데에 효과적이다[11]. 이동 사용자는 질의의 LPI 해쉬 값과 DIX 노드의 해쉬 값을 비교하여 위치 경로에 따라 질의 처리를 하지 않더라도 현재 노드의 위치 경로를 파악할 수 있다.

3.3 유사 노드 링크와 외래 노드 링크

유사 노드 링크와 외래 노드 링크는 XML 스트림 상에서 선택적인 수신을 위해 사용되는 색인 정보이다.

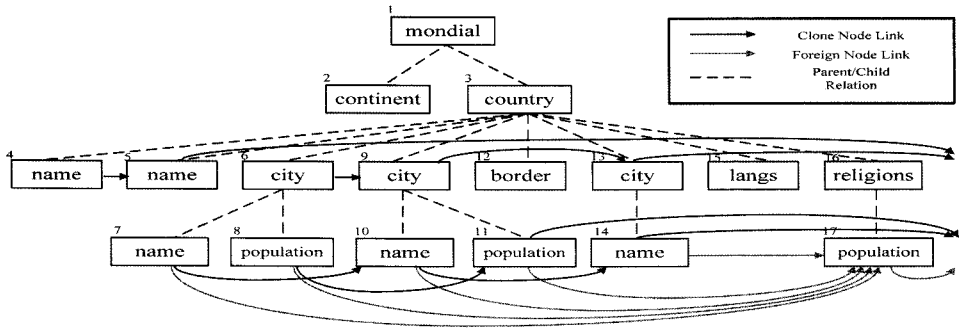
정의 3. 유사 노드 링크(Clone node Link: CL)란 현재 노드와 같은 깊이의 노드들 중에서 같은 위치 경로 정보와 같은 태그 이름을 가진 가장 가까운 다음 노드의 주소 정보이다.

정의 4. 외래 노드 링크(Foreign node Link: FL)란 현재 노드와 같은 깊이의 노드들 중에서 위치 경로 정보가 다른 가장 가까운 다음 노드의 주소 정보이다. □

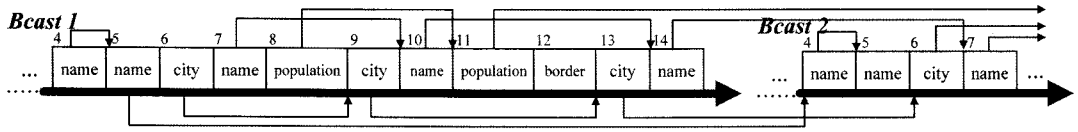
그림 6은 XML 트리에서의 CL과 FL 표현(a)과 XML 데이타 스트림에서 유사 노드 링크(b)와 외래 노드 링크(c)를 보여주고 있다. 질의 처리 단계에서 유사 노드 링크는 질의를 만족하는 노드를 찾은 후 질의를 만족하는 다음 노드를 빨리 찾기 위해 사용되며, 외래 노드 링크는 질의와 다른 경로를 가진 노드들의 탐색을 생략하는 데에 사용된다.

3.4 XML 데이타 스트림 생성 알고리즘

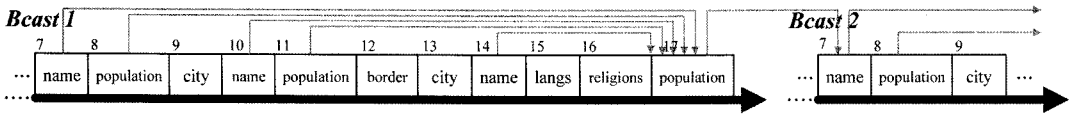
이 논문에서는 SAX[10]에서 제공하는 이벤트 핸들러인 ContentHandler를 이용해 XML 데이타 스트림을 생성하였다. 스트림의 생성은 두 개의 단계로 구분된다. 첫 번째 단계에서는, 데이타 세그먼트와 DIX 노드의



(a) XML 트리에서의 CL과 FL



(b) XML 데이타 스트림에서의 CL



(c) XML 데이타 스트림에서의 FL

그림 6 유사 노드 링크(CL)과 외래 노드 링크(FL)

LPI를 구성한다. 이 단계에서 스트림 생성기는 엘리먼트의 태그 이름, 애트리뷰트, 텍스트를 추출하여 데이타 세그먼트를 구성하고, 엘리먼트의 LPI를 부호화한다. 두 번째 단계에서는, 각 DIX 노드의 CL가 FL을 계산하여 문서에 나타난 엘리먼트 순서대로 DIX 노드를 배열한다.

그림 7은 XML 데이타 스트림 생성 알고리즘을 기술하고 있다. 먼저 스트림 생성기는 XML 문서의 파싱을 시작하면 초기화한다(4~9번째 줄). 스트림 생성기는 문서를 파싱하는 동안 엘리먼트의 시작 태그를 만날 때마다 *startElement()* 이벤트 핸들러를 실행하여 태그 이름과 애트리뷰트를 추출한다(14번째 줄). 또한 경로 스택에 저장된 태그 이름들과 미리 정의된 해쉬 함수를 통해 4바이트 길이의 LPI를 계산하여 큐에 저장한다(15~17번째 줄). 만약 엘리먼트가 텍스트를 포함하고 있다면, 스트림 생성기는 *character()* 이벤트 핸들러를 호출하여 큐에 저장된 DIX 노드에 텍스트 정보를 추가한다(26~31번째 줄). 마지막으로 XML 문서의 끝을 만나면 큐에 저장된 DIX 노드들의 CL과 FL을 계산하고 큐에 저장된 모든 DIX 노드들을 XML 데이타 스트림으로 출력한다(35~36번째 줄).

4. XML 데이타 스트림에 대한 질의 처리

이 장에서는 제안 기법에 의해 생성된 XML 데이타 스트림에서 이동 사용자가 자신의 질의를 처리하는 방법에 대해 설명한다. 그림 8은 질의 처리 알고리즘을 나타내고 있다.

질의가 주어지면 이동 사용자는 방송 채널로부터 질의와 같은 깊이의 노드를 만날 때까지 DIX 노드를 수신한다(6번째 줄). 질의와 같은 깊이의 노드를 만나면 DIX 노드의 LPI와 태그 이름을 질의와 비교하여 일치할 경우 결과에 포함시킨다. 이때 만약 질의가 프레디케이트를 포함하고 있다면 프레디케이트를 만족하는 경우에만 결과에 포함시킨다(8~11번째 줄). 질의를 만족하는 DIX 노드를 찾은 후에는 이 노드의 CL를 이용하여 같은 LPI와 태그 이름을 가진 노드만 탐색한다(13~15번째 줄). 한편, DIX 노드가 질의와 위치 경로 정보가 다르다면, 이 노드와 같은 위치 경로 정보를 갖는 모든 노드는 질의를 만족하지 않으므로 이 노드의 FL를 이용하여 상관 없는 데이타의 수신을 생략한다(30~32번째 줄). 즉, 이동 사용자는 질의와 같은 깊이의 노드를 수신한 후에 CL과 FL를 사용하여 질의와 상관 없는 데이타의 수신을 효과적으로 생략할 수 있다.

```
01: Algorithm Stream_Generator
02: Input: A well-formed XML document D
03: Output: XML Data Stream DS
04: ContentHandler.startDocument()
05: begin
06:     Path Stack PS = Ø              // initialization
07:     Node Queue Q = Ø
08:     int depth = - 1
09: end
10:
11: ContentHandler.startElement()
12: begin
13:     depth = depth + 1
14:     Construct a DIX node DN with Depth, Tag name, and Attribute List
15:     LPI of DN = encoding(PS)
16:     Push Tag name into PS
17:     Push DN into Q
18: end
19:
20: ContentHandler.endElement()
21: begin
22:     depth = depth - 1
23:     Pop an top entry from PS
24: end
25:
26: ContentHandler.characters()
27: begin
28:     Get the top entry DN in Q
29:     Initialize Text Data of DN
30:     Increase Node Size by the length of Text Data
31: end
32:
33: ContentHandler.endDocument()
34: begin
35:     Generate Clone node Link and Foreign node Link of all the DN in Q
36:     Flush all the DN in Q into the XML data stream DS
37: end
```

그림 7 XML 데이타 스트림 생성 알고리즘

예제 2. 그림 9는 이동 사용자가 "mondial/country/city/name"라는 질의를 처리하기 위해 5번 노드가 방송될 때 방송 채널에 들어와서 질의 처리를 수행하는 단계를 보여주고 있다. 5번 노드의 깊이가 질의가 다르므로 이동 사용자는 계속해서 데이타를 수신한다. 7번 노드를 수신한 후에 이동 사용자는 이 노드의 LPI와 태그 이름을 질의와 비교하여 질의를 만족한다는 것을 알 수 있다. 따라서 7번 노드를 결과에 포함시킨다. 또한 이동 사용자는 7번 노드의 CL로부터 현재 노드와 LPI와 태그 이름이 같은 데이타(10번 노드)가 언제 방송되는지 알 수 있다. 따라서 10번 노드가 방송될 때까지 대기 모드로 들어가서 전원 소모를 줄인다. 이동 사용자는 10번 노드가 방송될 때 대기 모드에서 깨어나 데이타를 수신하고 같은 방식으로 14번 노드만 선택적으로 수신한다. 이 예제에서 이동 사용자는 5, 6, 7, 10, 14번 노드를 수신하는 것만으로 질의 처리를 완료할 수 있다.

예제 3. 이동 사용자가 "mondial/country/religions/population"이라는 질의를 처리하기 위해 5번 노드가 방송될 때 방송 채널에 들어왔다고 가정하자(그림 10). 예

```
01: Algorithm Query_Processing_Algorithm_over_XML_Stream
02: Input a XML stream S, XML path query Q
03: Output result set R satisfying Q
04: begin
05:     while(access latency of client <= the length of the stream)
06:         Read currentNode from the Stream S
07:         if(depth of query == the depth of currentNode) then
08:             if(LPI of currentNode == LPI of the query) then
09:                 if(tagName of currentNode == tagName of the query) then
10:                     if(currentNode satisfies Predicate of the query) then
11:                         Insert currentNode into result set R
12                      endif
13:                     if(CL of currentNode is not NULL) then
14:                         if(access latency including the CL node <= the length of the stream) then
15:                             Wait in doze mode until the CL node arrives on the air
16:                         else
17:                             return R
18:                         endif
19:                     else
20:                         return R
21:                     endif
22:                 else
23:                     if(access latency of client < the length of the stream) then
24:                         Wait until the next node arrives on the air
25:                     else
26:                         return R
27:                     endif
28:                 endif
29:             else
30:                 if(FL of currentNode is not NULL) then
31:                     if(access latency including the FL node <= the length of the stream) then
32:                         Wait in doze mode until the FL node arrives on the air
33:                     else
34:                         return R
35:                     endif
36:                 else
37:                     return R
38:                 endif
39:             endif
40:         else
41:             if(access latency of client < the length of the stream) then
42:                 Wait until the next node arrives on the air
43:             else
44:                 return R
45:             endif
46:         endif
47:     endwhile
48: end
```

그림 8 질의 처리 알고리즘

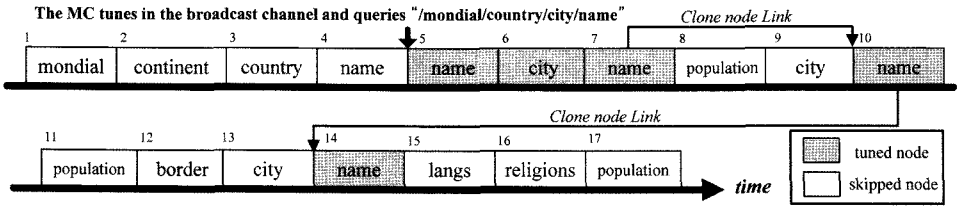


그림 9 CL을 이용한 선택적인 수신

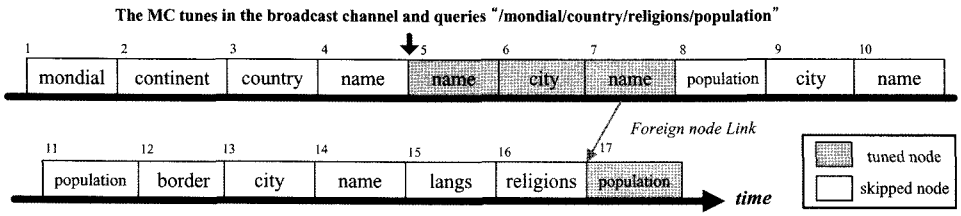


그림 10 FL을 이용한 선택적인 수신

제 2와 마찬가지로 이동 사용자는 같은 깊이의 노드(7번 노드)가 도착할 때까지 데이타를 수신한다. 7번 노드 수신 후에는 이 노드의 LPI가 질의와 다르다는 것을 알 수 있다. 따라서 이동 사용자는 7번 노드와 같은 LPI를 가진 노드들(8, 10, 11, 14번 노드)은 질의를 만족하지 않으므로 7번 노드의 FL을 이용하여 7번 노드와 다른 LPI를 가진 노드(17번 노드)가 방송 될 때까지 수신을 생략한다. 17번 노드의 LPI와 태그 이름을 비교하여 17번 노드가 질의를 만족한다는 것을 알 수 있다. 따라서 17번 노드를 결과에 포함시킨 후 17번 노드가 CL을 가지고 있지 않으므로 질의 처리를 종료한다. 이 예제에서 이동 사용자는 8, 9, 10, 11, 12, 13, 14, 15, 16번 노드의 수신을 생략할 수 있다. □

5. XML 데이타 스트림 클러스터링

5.1 클러스터링의 기본 개념

앞에서 설명한 DIX 노드를 이용한 분산 색인 구조는 질의 처리를 이동 사용자가 방송 채널에 들어온 순간부터 시작할 수 있으므로 접근 시간 성능을 크게 향상 시킨다. 그러나 XML 문서의 순서대로 생성된 스트림에서는 같은 위치 경로를 가진 노드들이 스트림 전체에 걸쳐 분포되어 있으며 같은 태그 이름을 가진 엘리먼트가 문서 전체에 걸쳐 나타날 수 있기 때문에 이동 사용자가 모든 결과 값을 반환 받기 위해서는 스트림의 대부분을 탐색해야 한다. 한편, 같은 특성을 가진 노드들을 클러스터링한다면 이동 사용자는 질의와 같은 특성의 노드들이 클러스터되어 있는 스트림의 일부만 탐색함으로써 모든 결과 값을 찾을 수 있으므로 접근 시간과 튜닝 시간을 향상 시킬 수 있다. 이러한 관찰로부터, 본 논문에서는 XML 엘리먼트들의 깊이에 기반한 클러스터링 정책을 제안한다.

정의 5. 클러스터링 영역(Clustering Region: CR)이란 노드의 깊이에 따라 나누어진 스트림의 논리적인 영역이다. 깊이 i의 클러스터링 영역 CR은 다음과 같이 표현될 수 있다. CR_i = {DIXk|DIXk is the DIX node for an element in dept i}. □

5.2 클러스터된 스트림을 위한 확장 노드 구조

클러스터된 스트림에서의 XML 엘리먼트의 순서는 원본 XML 문서의 순서와 다르기 때문에 본 논문에서는 듀이 순서(Dewey order)[12]를 사용하여 각 DIX 노드에 원래의 순서를 표기하였다. 스트림 생성기는 원본 XML 문서를 파싱하면서 듀이 순서를 계산하여 DIX 노드의 Order of element 필드에 저장한다.

한편, 4장에서 질의 처리 알고리즘에서 설명했듯이 이동 사용자는 DIX 노드의 유사 노드 링크와 외래 노드 링크를 사용하여 효율적으로 원하는 데이타를 탐색한다. 그러나 이와 같은 탐색 방법은 질의와 같은 깊이를 가진 노드가 나타날 때까지 계속해서 데이타를 수신해야 하는 문제점이 있다. 따라서 클러스터된 스트림 상에서 질의와 깊이가 다른 노드들의 검색을 생략하기 위해 다음 깊이의 노드들이 클러스터된 영역(the next CR)의 주소를 가리키는 새로운 색인 정보(Next Depth Pointer)를 추가한다.

정의 6. 다음 깊이 주소(Next Depth Pointer: NDP)란 각 DIX 노드에 포함되는 색인 정보로 XML 데이타 스트림에서 다음 깊이의 노드들이 클러스터된 영역의 시작 주소를 가리킨다. 이동 사용자는 방송 채널에 들어왔을 때 처음 수신한 노드의 깊이가 질의의 깊이와 다

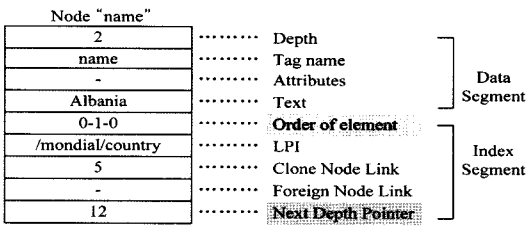


그림 11 확장된 DIX 노드

를 경우 NDP를 사용하여 불필요한 스트림의 수신을 생략한다. □

그림 11은 Order of element 필드와 Next Depth Pointer 필드가 추가된 확장된 DIX 노드 구조를 보여주고 있다.

한편, DIX 노드들을 깊이에 따라 클러스터링하기 위해서 3.4절에서 문서의 순서대로 스트림을 생성하는 알고리즘(그림 7)의 *endDocument()* 이벤트 핸들러(33~37번째 줄)를 다음과 같이 수정하여야 한다.

5.3 클러스터된 스트림에 대한 질의 처리

각 DIX 노드들이 NDP를 가지고 있기 때문에 이동 사용자는 질의와 같은 깊이의 노드들이 클러스터된 영역(target CR)을 찾을 수 있다. 질의 처리 단계에서 NDP를 활용하기 위해 그림 8의 질의 처리 알고리즘의 일부(6~7번째 줄)를 다음과 같이 수정한다. 이동 사용자는 질의와 같은 깊이의 노드들이 클러스터된 영역을 탐색한 후에 바로 질의 처리를 종료할 수 있다.

예제 4. 그림 12는 클러스터된 스트림에 대한 질의 처리 단계를 보여주고 있다. 이동 사용자가 "/mondial/country/city/name"이라는 질의를 처리하기 위해 5번 노드가 방송될 때 방송 채널에 들어왔다고 가정하자. 이 때, 질의의 깊이는 3이기 때문에 CR_2의 노드들(즉, 깊이가 2인 노드들)은 질의의 결과값이 될 수 없다. 따라서 5번 노드의 NDP를 사용하여 6번 노드부터 11번 노드까지의 수신을 생략하고 12번 노드를 수신한다. 12번 노드의 LPI와 태그 이름으로부터 이동 사용자는 12번 노드가 질의를 만족한다는 것을 알 수 있다. 따라서 12번 노드를 결과에 포함시키고 12번 노드의 CL을 사용하여 질의를 만족하는 다음 노드(14번 노드)를 선택적으로 수신한다. 마지막으로 14번 노드의 CL을 이용하여 16번 노드를 선택적으로 수신하고 질의 처리를 종료한다. □

6. 성능 평가

이 장에서는 제안 기법의 성능을 평가 한다. 실험은 mondial과 SigmodRecord 두 가지의 XML 데이타 집합[13]에 대해 [7]에서 제안하는 Path Summary 방법과 [6]에서 제안하는 S-node와 비교하였다. 경로 표현 질의로 XPath 질의[14]를 사용하였으며, 실험에 사용된 XPath 질의는 표 1과 같다. Path Summary 방법에서는 XML 엘리먼트의 순서가 무시되었고, 후손 축(descendant axe)나 프레디케이트(predicate condition) 처리가 지원되지 않으므로 단순 XPath 질의에 대한 성능 비교만 기술하였다.

실험은 표 1의 질의들에 대해 버킷의 크기에 따른 성능 측정과 클러스터링을 적용하지 않았을 때의 성능과 적용했을 때의 성능을 각각 측정하였다. 버킷의 크기에 따른 영향을 측정하기 위해, 128 바이트, 256 바이트, 그리고 512 바이트의 크기로 실험하였다. 그러나 버킷의 크기에 대한 성능 편차가 거의 없기 때문에 본 논문에서는 256 바이트 크기의 버킷에 대한 실험 결과만을 기술한다. 클러스터링을 적용하지 않았을 때의 성능 측정

표 1 실험에서 사용된 XPath 질의들

Dataset	Test Query	
mondial (ver. 3.0)	Q1	/mondial/country/name[text()="Finland"]
	Q2	/mondial/country/province[@name="Tyrol"]
	Q3	/mondial/country//city
	Q4	/mondial/country/*/name
	Q5	/mondial/country/province/city/name
Sigmod Record	Q6	/SigmodRecord/issue/articles/article
	Q7	/SigmodRecord/issue/articles/article/title
	Q8	/SigmodRecord/issue/articles/article/title[num="0"]

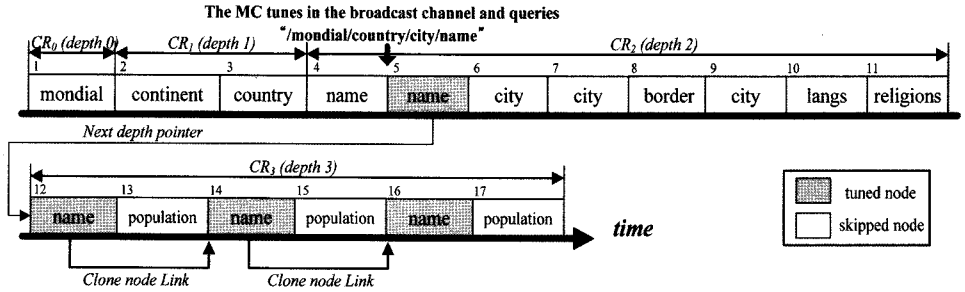


그림 12 클러스터된 스트림에 대한 질의 처리

결과와 클러스터을 적용했을 때의 성능 측정 결과는 각각 6.1절과 6.2절에 기술한다.

6.1 클러스터링을 적용하지 않았을 때의 성능 평가

이 절에서는 클러스터링을 적용하지 않았을 때의 성능 측정 결과를 기술하였다. 그림 13은 원본 XML 문서와, Path Summary, SL, SD, SP, DIX의 다섯 가지 방법에 의해 생성된 스트림의 크기를 보여주고 있다. 여기서 SL, SD, SP는 S-node에서 제안한 방법이며, 마지막으로 DIX는 본 논문에서 제안하는 분산 색인 구조를 적용한 방법이다. Path Summary 방법은 XML 엘리먼트의 중복된 태그 이름을 생략하였기 때문에 스트림의 크기가 가장 작으며, DIX는 색인 정보를 가장 많이 가지고 있기 때문에 스트림의 크기가 다른 기법들에 비해 비교적 크다. 하지만, 종료 태그(end tag)를 제거하여 구조적인 과부하(structural overhead)를 줄였기 때문에 원본 문서의 크기보다 작다.

그림 14는 접근 시간을 측정한 결과를 보여주고 있다. 제안 기법은 S-node의 모든 방법보다 우수한 접근 시간 성능을 보여준다. 평균적으로 SL방법에 비해 약 32% 정도 단축된 접근 시간을 보여주며, SD와 SP 방법에 비해 약 20% 정도 단축된 접근 시간을 보여주었다. S-node 방법과 Path Summary 방법은 질의 처리가 위치 경로를 따라 수행되므로 방송 중간에 방송 채널에 들어온 이동 사용자들은 다음 방송 주기까지 기다려야 한다. 이러한 대기 시간은 전체 방송 주기의 1/2에 해당하는 접근 시간을 증가시킨다. 한편, 제안 방법은 각 DIX 노드가 위치 경로 정보(LPI)를 포함하고 있으므로 이동 사용자가 방송 채널에 들어온 순간부터 질의 처리를 수행할 수 있다. 따라서, S-node보다 큰 스트림의 크기에도 불구하고 접근 시간은 오히려 단축되었다. Path Summary 방법의 경우, 단순 XPath 질의들(Q5~Q7)만에 대해서만 질의 처리가 가능했으며, 이때 Path Summary 방법에 의해 생성된 스트림의 크기가 가장 작기 때문에 다른 방법들에 비해 접근 시간 성능이 우수하였다.

그림 15는 튜닝 시간을 측정한 결과이다. 제안 방법의 튜닝 시간은 SL, SD 방법보다 단축되었지만, SP보다는 약간 증가되었다. 이는 제안 방법에서 이동 사용자가 질의와 같은 깊이의 노드를 수신할 때까지 연속해서 데이타 스트림을 수신하기 때문이다. Q6의 경우에는 질의에서 탐색하는 "article" 엘리먼트가 텍스트를 포함하고 있지 않기 때문에 Path Summary 방법 경우의 튜닝시간이 거의 없었다.

6.2 클러스터링을 적용했을 때의 성능 평가

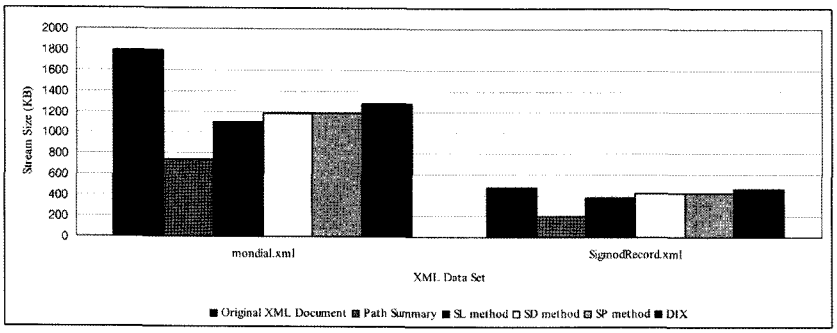


그림 13 스트림의 크기 비교

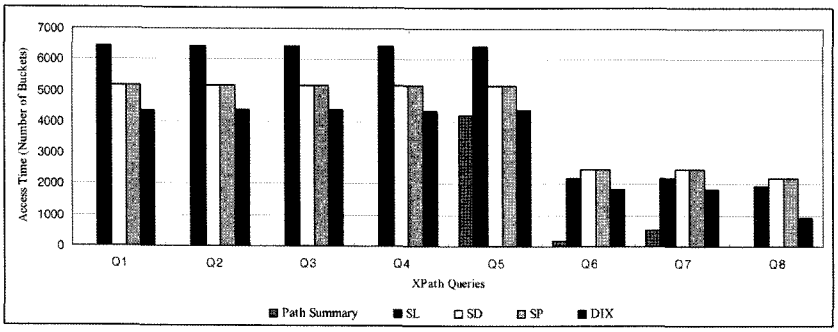


그림 14 접근 시간 측정

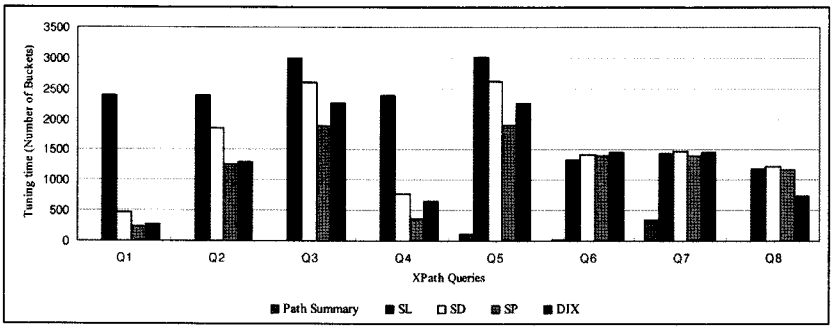


그림 15 튜닝 시간 측정

이 절에서는 분산 색인 구조와 함께 클러스터링을 적용했을 때의 성능을 측정하였다. 그림 16은 원본 XML 문서와 실험에서 사용한 다섯 가지 방법에 의해 생성된 스트림의 크기를 보여주고 있다. 여기서 clustered DIX는 분산 색인 구조와 클러스터링을 적용한 방법이다. clustered DIX는 기존의 DIX 노드 구조에 두 가지의 색인 정보 즉, *Order of Element*와 *Next Depth Pointer*가 추가되었기 때문에 스트림의 크기가 약간 더 증가되었다.

그림 17은 접근 시간을 측정한 결과를 보여주고 있다. 본 논문에서 제안하는 clustered DIX 방법은 SL, SD, SP 방법에 비해 각각 46%, 37%, 37%의 접근 시간을 단축시켰다. 접근 시간이 단축되는 이유는 다음의 두 가지 이유로 설명된다. 1) S-node나 Path Summary의 질의 처리 방법은 위치 경로를 따라 수행되기 때문에 항상 스트림의 시작 부분(즉, 문서의 루트가 방송되는 위치)부터 시작되어야 한다. 따라서 방송 중간에 방송 채널에 들어온 이동 사용자는 다음 방송 주기를 기다려야 하므로 평균적으로 방송 주기의 1/2에 해당하는 지연시간이 생긴다. 2) S-node는 질의를 만족하는 모든 노드들을 반환하기 위해서 거의 전체 스트림을 탐색하여야 한다. 한편, 제안 방법에서는 같은 깊이를 가진 노

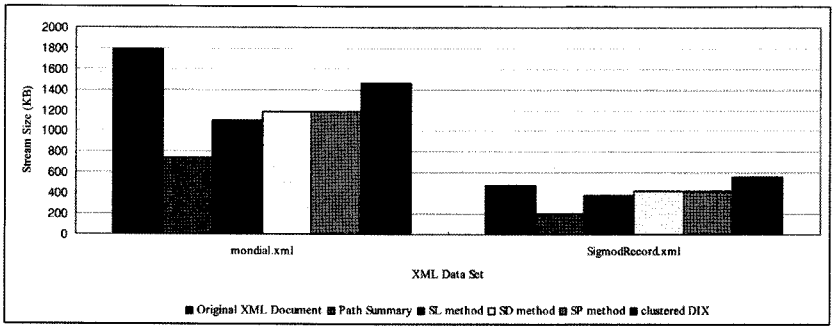


그림 16 스트림 크기 비교

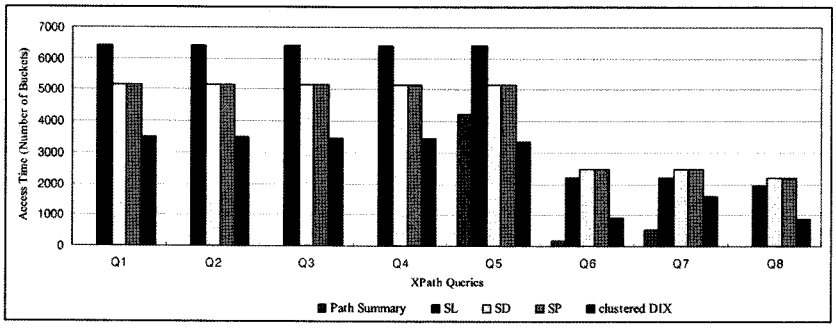


그림 17 접근 시간 측정

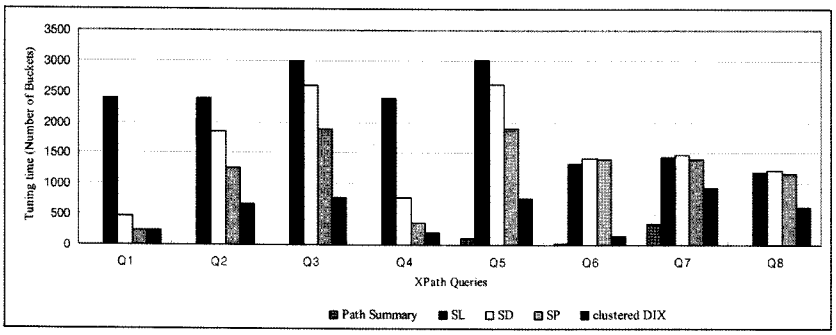


그림 18 튜닝 시간 측정

드들이 클러스터되어 있으므로 이동 사용자가 찾는 데이타가 포함되어 있는 클러스터링 영역(CR)이 방송되기 전에 들어올 경우 접근 시간이 크게 단축된다.

그림 18은 튜닝 시간을 측정한 결과이다. 제안 방법인 clustered DIX는 SL, SD, SP 방법에 비해 각각 75%, 66%, 55%의 튜닝 시간이 단축되었다. 제안 방법에서 튜닝 시간이 단축되는 이유는 다음과 같다. 1) 클러스터된 스트림의 각 DIX 노드는 NDP를 가지고 있으므로 질의의 결과값이 될 수 있는 노드들이 모여있는 클러스터링 영역을 파악하는데 소요되는 튜닝 시간이 최소화된다. 2) clustered DIX에서는 위치 경로를 탐색하는데 소요되는 튜닝 시간이 없다. 3) 같은 깊이의 노드들만 클러스터되어 있으므로 각각의 버킷에 포함되는 결과값들이 많다. 즉, 버킷 수신이 최소화 된다.

한편, Q1에 대해서는 clustered DIX 방법과 SP 방법의 성능이 비슷한데, 이는 Q1의 결과 값이 스트림의 앞부분에 위치하므로 SP 방법의 질의 처리가 빨리 끝나기 때문이다.

7. 결 론

본 논문에서는 무선 방송 환경에서 에너지 및 접근 시간 효율적인 XML 데이타의 스트리밍 방법에 대해 연구하였다. 트리 기반의 색인 구조에서 발생하는 "질의 처리의 지연 문제"를 해결하기 위해 분산 색인 구조를 제안하였다. 본 논문에서 정의한 DIX 노드 구조는 XML 엘리먼트의 데이타와 효율적인 질의 처리를 위한 색인 정보를 모두 포함하고 있으며, 특히, 해쉬된 위치 경로 정보(LPI)를 포함하고 있기 때문에 이동 사용자가 다음 방송 주기를 기다릴 필요 없이 방송 채널에 들어온 순간부터 질의 처리를 시작할 수 있다. 또한 유사 노드 링크(CL)와 외래 노드 링크(FL)를 사용하여 보다 효율적으로 질의 처리를 할 수 있다. 여기에 본 논문에서 제안하는 클러스터링을 적용하여 탐색 범위를 한정시킴으로써 이동 사용자는 스트림의 일부만 탐색함으로써 모든 결과값을 찾을 수 있다. 다양한 특성의 XPath 질의를 사용한 실험을 통해 본 논문에서 제안하는 DIX 노드를 통한 분산 색인 구조와 클러스터링을 통해 기존의 방법에 비해 접근 시간과 튜닝 시간을 효과적으로 단축시킴을 증명하였다. 향후 연구에서는 질의 처리를 보다 효율적으로 처리할 수 있도록 비슷한 구조적 특성을 지닌 노드들로 구성된 클러스터링 영역 내에서의 필터링 기법에 대한 연구가 고려되어야 할 것이다.

참 고 문 헌

[1] Acharya, S., Alonso, S., Franklin, M. J., and Zdonik, S. B., "Broadcast disks: Data management for asymmetric communication," In Proceedings of ACM SIGMOD Conference on Management of Data, pp. 199-210, 1995.

[2] Imielinski, T., Viswanathan, S., and Badrinath, B. R., "Data on air: Organization and access," IEEE Transactions on Knowledge and Data Engineering, Vol.9, pp. 353-372, 1997.

[3] Imielinski, T., Viswanathan, S., and Badrinath, B. R., "Energy Efficient Indexing on Air," In Proceedings of the ACM SIGMOD Conference, pp. 25-36, 1994.

[4] Extensible Markup Language, http://www.w3.org/XML.

[5] Wong, E. Y. C., Chan, A., and Leong, H., "X-stream: A middleware for streaming xml contents over wireless environments," IEEE Transactions on Software Engineering, Vol.30, pp. 918-935, 2004.

[6] Park, C. -S., Kim, C. S., and Chung, Y. D., "Efficient stream organization for wireless broadcasting of xml data," In Proceedings of Asian Computing Science Conference, pp. 223-235, 2005.

[7] Park, S. H., Choi, J. H., and Lee, S., "An Effective, Efficient XML Data Broadcasting Method in Mobile Wireless Network," In the Proceedings of DEXA Conference, pp. 358-367, 2006.

[8] Lam, W. Y., Ng, W., Wood, P. T., and Levene, M., "XCQ: XML Compression and Querying

System," In Proceedings of the International WWW Conference, 2003.
[9] Liefke, H. and Suciu, D., "XMill: An Efficient Compressor for XML Data," In Proceedings of the ACM SIGMOD Conference, pp. 153-164, 2000.
[10] Simple API for XML, http://www.saxproje ct.org/, 2004.
[11] Revest, Ronald L., The MD5 Message Digest Algorithm, RFC 1321, 1992.
[12] Online Computer Library Center. Introduction to the Dewey Decimal Classification. http://www.oclc.org/oclc/fp/about/about_the_ddc.htm
[13] XML data repository, http://www.cs.washington.edu/research/xmldatasets.
[14] World Wide Web Consortium. XML Path Language (XPath), Version 1.0, W3C Recommendation, November 1999.

박 준 표

2005년 8월 동국대학교 컴퓨터공학과 졸업(학사). 2007년~현재 고려대학교 컴퓨터학과 석사과정. 관심분야는 데이타베이스, XML, XMLTM, Sensor network 등

박 창 섭

1995년 2월 한국과학기술원 전산학과 학사. 1997년 2월 한국과학기술원 전산학과 석사. 2002년 2월 한국과학기술원 전자전산학과 전산학전공 박사. 2002년 3월~2005년 2월 KT 서비스개발연구소 선임보연구원. 2005년 3월~현재 수원대학교 인터넷정보공학과 조교수. 관심분야는 Semantic Web, Web Services, XML, Database Systems 등

정 연 돈

1994년 2월 고려대학교 전산과학과 학사 1996년 2월 한국과학기술원 전산학과 석사. 2000년 8월 한국과학기술원 전자전산학과 전산학전공 박사. 2000년 9월~2001년 8월 한국과학기술원 정보전자연구소 Post-Doc. 연구원. 2001년 9월~2003년 2월 한국과학기술원 전자전산학과 전산학전공 연구교수. 2003년 3월~2006년 2월 동국대학교 컴퓨터공학과 교수. 2006년 3월~현재 고려대학교 컴퓨터 · 통신공학부 교수 관심분야는 XML, Mobile/Broadcast Databases, Sensor Networks, Spatial Databases, Database Systems 등