무선 XML 스트림을 위한 색인 기법
(An Index Method for Wireless XML Streams)

정 연 돈 † 이 지 연 ††
(Yon Dohn Chung) (Ji Yeon Lee)

요 약 본 논문은 무선 정보 시스템 환경에서, 서버가 다수의 클라이언트들에게 무선 방송 기법을 통해 XML 데이타를 스트리밍 서비스할 때 필요한 색인 기법을 제안한다. 제안 하는 색인 방법은 XML 데이타의 스트리밍시 클라이언트들의 접근 시간 및 튜닝 시간을 효과적으로 제어하기 위하여, XML 데이타 및 색인 정보를 부분적으로 반복, 배치하여 스트림을 구성한다. 이를 위하여 트리형태로 표현되는 XML 데이타와 색인 정보를 2-레벨로 구분하여, 색인 및 데이타의 중복 배치 영역을 설정한다. 제안하는 색인 기법의 성능을 접근 시간 및 튜닝 시간 측면에서 분석하여, 분석의 결과로 최적의 레벨 깊이를 결정한다.

키워드 : XML 스트림, 색인, 무선 데이타 방송, 접근 시간, 튜닝 시간

Abstract In the paper, we propose an index method for XML streaming services, where a server broadcasts XML data to a lot of clients in wireless information systems. In order to control the access and tuning time of mobile clients, the proposed method constructs the XML stream through replicating partial index intermixed with parts of data. For this purpose, we propose a two-level tree structure for separating index and XML data into two parts: replicated vs. non-replicated. We analyze the performance of the proposed method with respect to access time and tuning time. With the analysis result, we derive the optimal level value.

Key words : XML Stream, Indexing, Wireless Data Broadcasting, Access time, Tuning time

1. 서 론

무선 통신 및 정보 기술의 발달로 무선 정보 시스템(Wireless Information System) 분야에 대한 관심이 최근 높아지고 있다[1,2]. 무선 정보 시스템이란 이동 통신 기술을 통하여 사용자들이 휴대용 단말 장비를 사용하여 지리적 제약을 받지 않고 정보 서비스를 제공받을 수 있는 시스템이다. 본 논문은 그림 1과 같이 무선 정보 시스템에서 XML로 표현된 데이타를 사용하는 환경을 가정한다.

XML은 인터넷에서 데이타를 교환하는데 사용되는 표준으로서, 그 응용 범위가 확대되고 있는 정보 기술이다[3,4]. 인터넷 혹은 유선 네트워크를 기반하는 정보 시스템에서 XML을 사용하고 활용하는 부분에 대한 연구는 현재 많이 이루어지고 있으나, 무선 통신을 사용하는 이동 컴퓨팅 환경을 기반으로 하는 무선 정보 시스템에서의 XML 활용 기술은 아직 활발히 연구되어지지 않고 있다.

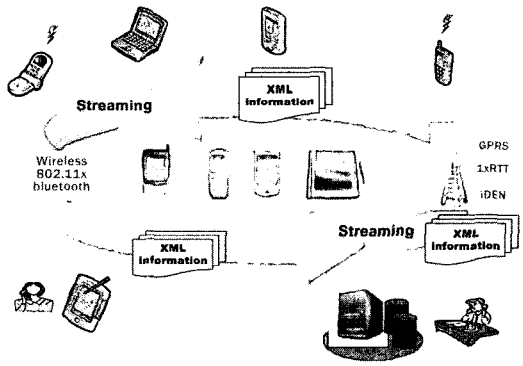


그림 1 무선 정보 시스템에서의 XML 정보 스트리밍 서비스

본 논문에서는 XML을 데이타 표현 방법으로 사용하는 무선 정보 시스템에서, 무선 스트리밍을 통한 데이타의 전송시 필요한 색인 기법을 제안한다. 먼저, 2장에서

· 본 연구는 "정보통신기초기술연구지원사업"의 부분적인 지원을 받아 수행되었음
† 종신회원 : 동국대학교 컴퓨터공학과 교수
ydchung@dgu.edu
†† 비 회 원 : 한국전산원 연구원
jylee@nca.or.kr
논문접수 : 2004년 9월 24일
심사완료 : 2005년 5월 26일

XML 및 무선 데이타 스트리밍에 대한 배경 설명 및 관련 연구를 소개하고, 본 논문에서 다루는 무선 스트림의 색인에 대한 필요성을 설명한다. 3장에서는, 본 논문이 제안하는 색인 기법을 설명하고, 4장에서 제안하는 방법의 성능을 분석한다. 마지막 장에서, 결론과 함께 향후 연구 방향을 제시한다.

2. 연구의 배경 및 필요성

2.1 XML

XML(eXtensible Markup Language)[3-5]은 HTML과 같이 태그(tag)와 애트리뷰트(attribute)를 사용하여 정보를 기술하는 마크업(Markup) 언어로서, HTML은 사용할 수 있는 마크업이 처음부터 고정되어 있는 반면에, XML은 사용자가 자신의 마크업을 정의할 수 있다. 따라서, 데이타의 표현 형식을 나타내는 태그로 이루어진 HTML과 달리 데이타의 내용을 표현하기 위해 태그를 정의하여 사용할 수 있다. 이러한 이유로 인터넷을 통한 데이타 전송 및 표현의 표준으로 XML에 대한 관심이 높아지고 있으며, XML을 사용한 다양한 응용분야들이 출현하고 있다. XML 애플리케이션에서 XML 문서는 주로 DOM[4]이라는 트리 형태로 구성되어 사용된다. 아래의 그림 2에 XML 문서의 예제와 그 문서의 DOM 트리를 보였다.

2.2 무선 데이타 스트리밍

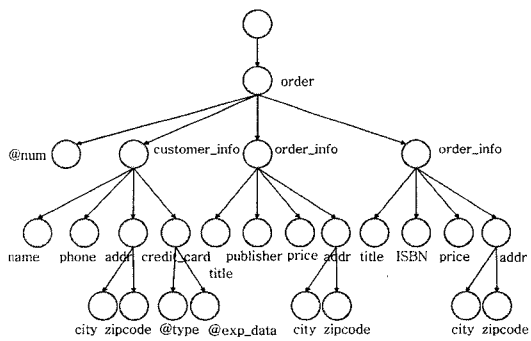
데이타 스트리밍(streaming)이란 정보가 임의의 출처에서 목적지로 끊임없이 전달되는 현상을 말하며, 이때 전송되는 데이타 열을 데이타 스트림(data stream)이라고 한다. 무선 정보 시스템 환경에서는 서버가 클라이언트에게 방송을 통해 데이타를 지속적으로 전송하는 과정이 이에 해당한다.

방송(broadcasting)이란 서버가 불특정 다수의 클라이언트들에게 방송 채널(broadcasting 또는 public channel)을 사용하여 데이타를 전달하는 방법이다. 서버와 클라이언트의 일대일 통신 방법에 비하여 서버로부터 전달되는 데이타를 수신만 하기 때문에 무선 환경에서는 주파수 사용 및 에너지 사용의 효율성이 높은 방법이다[6-10]. 무선 방송으로 이동 클라이언트들에게 전송되는 데이타를 본 논문에서는 '무선 데이타 스트림'으로 부르기로 한다.

무선 데이타 방송에 있어 두 가지 중요한 성능 요소로 *접근 시간(access time)*과 *튜닝 시간(tuning time)*이 있다[1,2]. 접근 시간(access time)이란 사용자가 방송 수신을 시작한 시점으로부터 자신이 원하는 데이타를 모두 읽는 시점까지의 소요 시간을 나타내며, *튜닝 시간(tuning time)*이란 이 접근 시간 동안 실제로 방송을 들어야 하는 시간을 나타낸다. 자신이 필요로 하지

```
<order num="b392-323">
  <customer_info>
    <name> Kim</name>
    <phone>010-1234-5678</phone>
    <addr>
      <city>Seoul</city>
      <zipcode>111-222</zipcode>
    </addr>
    <credit_card type="Master" exp_date="2006-12"/>
  </customer_info>
  <order_info>
    <title>Database Systems</title>
    <publisher>Addison Wesley</publisher>
    <price>25.00</price>
    <addr>
      <city>Busan</city>
      <zipcode>333-444</zipcode>
    </addr>
  </order_info>
  <order_info>
    <title>Information Systems</title>
    <ISBN>1-55860-622</ISBN>
    <price>39.95</price>
    <addr>
      <city>Seoul</city>
      <zipcode>111-555</zipcode>
    </addr>
  </order_info>
</order>
```

(a) XML 문서 예제



(b) XML 문서 예제 (a)에 대한 트리 표현

그림 2 XML 문서 예제 및 트리 형식의 표현

않는 데이타가 전송되는 시간 동안은 에너지 소모량이 적은 상태, 즉 *휴지 상태(doze mode)*로 지내게 된다. 그리고 실제로 방송 내용을 수신하는 동안은 *활성 상태(active mode)*로 있게 된다. 다시 말하면, 튜닝 시간은 접근 시간 동안 활성 상태로 지내는 시간을 말한다. 활성 상태에서 소모되는 에너지는 휴지 상태에서 소모되는 에너지의 수 천 배에 이른다[1]. 따라서 이동 단말기의 에너지 제약성을 고려할 때, 튜닝 시간을 단축하는

문제는 매우 중요하게 다루어지고 있다.

지금까지 무선 데이타 방송에 관련된 연구들에는 접근 시간을 줄이고자 하는 방법들-*데이타 캐싱(data caching)*[6,7], *불균등 데이타 방송(nonuniform data broadcasting)*[6], 다중 점 질의 혹은 부분 부합 질의를 위한 *클러스터링(clustering)*[9,10]-과 튜닝 시간을 줄이고자 하는 방법들-색인[1] 및 해싱[8,11]-이 있었다. (본 논문은 튜닝 시간을 줄이기 위한 방법이다.) 기존의 튜닝 시간을 줄이려는 연구들은 대부분 사용자가 데이타 레코드의 키를 사용한 *단일 점 질의(single point query)*[1]나 *다중 점 질의(multipoint query)*[9,10]를 사용한다는 가정을 기반으로, 디렉토리를 계층적으로 구성하는 방법들이었다.

2.3 무선 XML 데이타 스트리밍을 위한 색인의 필요성

무선 방송 데이타 스트림의 색인이란 방송으로 전달되는 데이타들을 클라이언트들이 적은 에너지 소모를 통해 찾을 수 있도록 하는 시간적인 주소 정보(그림 3에서 'Index' 부분에 해당) 를 방송 스트림에 데이타와 함께 전달하는 방법을 말한다. 즉, 클라이언트의 튜닝 시간을 줄이고자 하는 목적이다.

지금까지 무선 데이타 방송을 위한 색인 기법으로 *(1, M) Indexing* 기법, *Flexible Indexing* 기법, *Distributed Indexing* 기법 등이 제안되어 왔다[1]. 하지만, 이 방법들은 데이타들이 기본 키(primary key)로 식별되는 단일 구조의 레코드 집합을 스트리밍 하는 환경을 기반으로 하고 있다. 단순 레코드들에 대한 스트림이 아닌 XML 형태의 데이타를 스트리밍[12]하는 데에는 사용할 수 없는 방식이다. 기본 키를 사용하는 질의 형태가 "id=100인 레코드를 찾아라"와 같은 반면, XML 질의는 "/department/person/professor"와 같은 경로 질의 형태이기 때문이다.

기존의 색인 방식을 사용하여 XML 데이타에 적용한 매우 원시적인 색인을 예로 들자면 XML 데이타를 스트리밍할 때, 그림 3과 같이 XML 문서의 헤더부분에 색인을 추가하는 방법이 가능하다. 기본 키의 역할을 하는 부분인 대상 노드들을 나타내는 경로 정보(path specification)와 그에 해당하는 노드들의 주소(시작 주소 및 종료 주소) 쌍으로 이루어져 있다. 무선 스트리밍에서 주소란, 해당 데이타가 방송 채널에서 나타나는 시간적인 주소이다. 기존의 색인 기법과 비교하면 (1,M) 색인 기법에서 M=1인 경우에 대응되는 개념이다.

하지만, 위와 같은 단순 색인 구조는 색인이 스트림상에 분산 배치되지 않기 때문에, 색인을 찾기 위해 많은 시간을 소요하여야 하며(색인과 색인 사이에서 질의 검색이 시작될 경우 다음 색인이 방송될 때까지 기다려야 한다), 전체 데이타에 대한 색인을 모두 읽어야 하는 부담이 매우 크다. (위의 그림에서 Index로 표시되는 영역에서 특정 경로에 대한 주소 값을 찾으려면, 평균적으로 Index 크기의 절반에 해당하는 색인 정보들을 검색하여야 한다.) 기존의 색인 기법 중에서 Distributed Indexing 기법은 이와 같은 문제점을 해결하기 위해 전체 색인을 작은 단위의 색인으로 분해하여 방송 스트림의 여러 부분에 걸쳐 배치하는 방법을 사용하였다. 본 논문에서 제안하는 색인 방법 역시 Distributed Indexing의 색인 분산 철학을 따르는 방법이다. 하지만, Distributed Indexing이 평면적인 데이타를 구조화 색인으로 색인 분산을 시도한 방법이기에, 구조화 데이타인 XML 데이타를 그대로 적용하기는 불가능하다. Distributed Indexing 방법은 방송 스트림을 구성하는 레코드들의 주 키(primary key) 값을 사용하여 B+ 트리 형태의 균형 트리를 구성하는 방법이기 때문에, 데이타들 사이에 계층적 관계가 존재하는 XML 데이타에는 직접 적용할 수

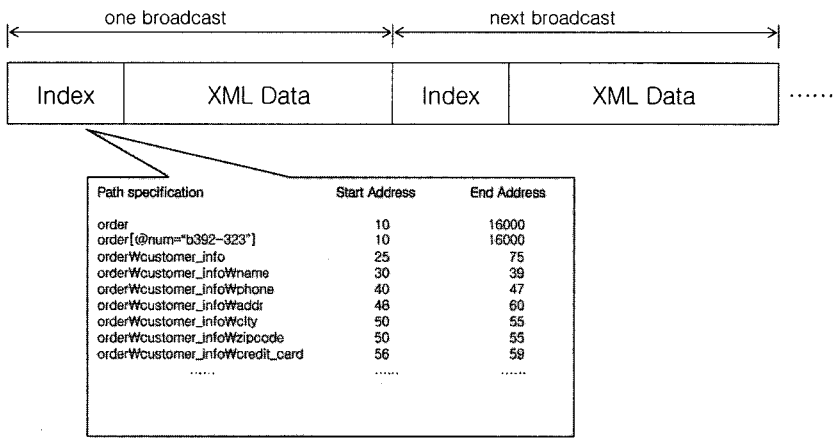


그림 3 단순한 XML 스트림 색인 방법

없기 때문이다. 이를 해결하기 위하여 본 논문이 제안하는 방법은, 색인과 함께 해당 데이타를 부분적으로 반복하는 접근 방법을 사용한다.

3. 제안하는 색인 기법

3.1 가정

본 논문에서는 무선 정보 시스템에서 방송 기법을 사용하여 XML 데이타를 스트리밍하는 환경을 위한 색인 기법을 이야기 하고 있다. 이를 위해 다음과 같은 가정을 사용한다.

가정 1. 스트리밍하는 XML 데이타는 XML 문서의 형태인 텍스트 형식이다.

가정 2. 스트리밍을 위해 원래 XML 문서의 내용에 대하여 수정을 가하지 않아야 한다.

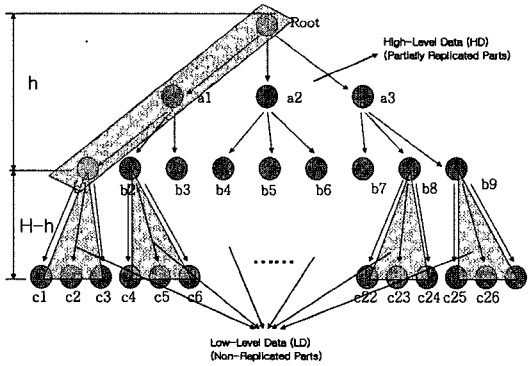
가정 3. 색인의 구성은 모든 데이타에 대한 균등한 성능을 제공하도록 한다.

첫 번째 가정은 XML을 DOM 형태로 메모리에 로드한 상태의 이진(binary) 형식이 아니라 디스크에 저장되어 있는 텍스트 형식의 XML 데이타를 스트림으로 전송한다는 의미이다. 일반적으로 DOM 형태로 변환된 XML 데이타는 텍스트 형식에 비해 크기가 매우 증가하므로, 주파수 영역의 사용이 제한적인 무선 환경에서는 바람직하지 않을 것이다. 두 번째 가정은 색인을 위한 작업을 위해 XML 데이타에 수정 혹은 삽입 등의 데이타 변경 작업을 가할 경우, 원시 XML 문서로의 복원 불가능한 상황이 생기지 않아야 함을 의미한다. 특히 스트림으로 데이타 전송시 데이타들이 조각으로 분리되어 전송되는데, 이 조각들을 결합하기 위한 정보를 스트림 상에 추가하는 Hole-Filler 모델[12]과 같은 테이터 내용 변경을 허용하지 않는다. 마지막 가정은, 색인 기법에 따라 데이타별로 서로 다른 성능을 보이도록 구성하지 않고, 모든 데이타에 대해 균등한 접근 성능을 보이도록 한다는 의미이다. 이는 동일한 수준의 노드들에 대해 동일한 접근 시간 및 튜닝 시간을 제공함을 의미한다. (또한, 본 논문에서는 XML 데이타를 접근하는 사용자의 질의 패턴에 대한 고려도 하지 않는다.) 만일 사용자 질의 패턴을 고려한다면, Hot 데이타와 Cold 데이타에 대하여 차별화된 접근성을 갖도록 색인을 설계할 수도 있기 때문이다. (향후 연구 방향에서 다시 설명하기로 한다.)

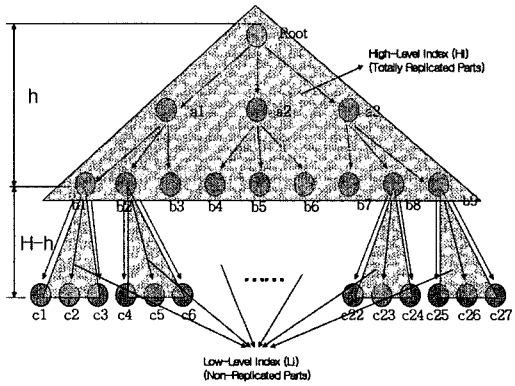
3.2 색인의 구조 및 스트림의 구성

색인의 구조 설명과 다음 장의 분석의 편의성을 위하여 XML 데이타가 차수가 n이고, 트리의 높이가 H인 완전 균형 트리 형태로 표현된다고 가정하자. (이 가정은 설명과 분석을 위한 가정이며, 모든 형태의 XML 데이타에 대하여, 제안하는 색인 기법이 사용될 수 있다). 다음은 본 논문에서 예제로 사용할 XML 문서로서 n=3, H=4인 문서의 예이다. 1번째 수준의 노드를 Root로 나타냈으며, 2번째 수준의 노드들을 a1, a2, a3, 3번째 수준의 노드들을 b1, b2, ... 4번째 수준의 노드들을 c1, c2, ... 로 표시하고 있다. 이 문서에 대한 트리 형태의 표현은 그림 4와 같다.

```
<Root>
  <a1>
    <b1>
      <c1> </c1> <c2> </c2> <c3> </c3>
    </b1>
    <b2>
      <c4> </c4> <c5> </c5> <c6> </c6>
    </b2>
    <b3>
                        ...
    </b3>
  </a1>
  <a2>
                        ...
  </a2>
  <a3>
                        ...
  </a3>
</Root>
```



(a) 데이타의 2-레벨 구조



(b) 색인의 2-레벨 구조

그림 4 색인 및 데이타의 2-레벨 구조

2.3에서 소개한 '단순 색인 구조' 같이 전체 색인 정보를 매 방송 스트림의 헤더에 한번만 배치하는 경우와 달리, XML 데이타 스트림에 부분적인 색인 정보들을 분산 배치하여 구성할 때는, 색인 정보의 배치가 균등하지 않을 경우 각 데이타 노드마다 접근 성능이 달라진다. 본 논문에서는 균등 접근 성능을 가정하고 있기 때문에, 색인들 사이의 거리가 동일하도록 균등 배치를 사용한다. 그리고, XML 데이타의 균등 배치를 위해서는 일부 XML 데이타의 반복 배치가 필수적이다. 가령 그림 4의 트리를 DFS(Depth-First Search) 방법으로 나열할 경우를 생각해 보자.

Root, a1, b1, c1, c2, c3, b2, c4, c5, c6, b3, c7, c8, c9, a2, b4, c10, c11, ..., a3, b7, ...

여기에 전체 색인을 3등분한 색인을 스트림의 3곳에 분산 배치하여 보자. 아마도, 전체 데이타를 3등분 할 수 있는 단위가 깊이 2인 a1, a2, a3 노드와 일치하므로 다음과 같이 색인을 배치할 수 있을 것이다.

Index, Root, a1, b1, c1, c2, c3, b2, c4, c5, c6, b3, c7, c8, c9, **Index**, a2, b4, c10, c11, c12, ..., c18, **Index**, a3, b7, ...

이 경우, a1의 경우 바로 앞에 위치에 Root노드가 존재하여 색인을 Root를 포함하도록 구성한 다음 Root 노드 앞에 배치하여야 하지만, a2의 경우는 그렇지 않다. 이는 동일한 수준의 두 노드 사이의 간격이 다르다는 의미이며, 이러한 노드를 기준으로 색인을 분산 배치할 경우, 색인들 사이의 거리 역시 차별화 되어, 접근 성능의 차이를 야기한다. 따라서, 색인을 스트림 상에서 여러 군데 배치할 경우에는 색인들 사이의 간격이 동일하게 배치하여야 하며, 이를 위해 필요에 따라 데이타의 중복이 필요하게 된다. 위의 예에서 데이타를 중복 배치하여 색인 배치 간격을 동일하게 한다면 다음과 같은 배치가 가능할 것이다. (여기서 Root', Root''는 Root노드의 첫 번째, 두 번째 중복을 나타낸다.)

Index, Root, a1, b1, c1, c2, c3, b2, c4, c5, c6, b3, c7, c8, c9, **Index, Root'**, a2, b4, c10, c11, c12, ..., c18, **Index, Root''**, a3, b7, ...

2.3에서 지적한대로, 스트림 상에서 색인 정보를 분산 배치시키기 위하여, 본 논문에서 제안하는 색인은 XML 데이타 트리에 대하여 두 가지 레벨(level)을 정의한다. 그림 4와 같이 상위 h까지의 부분을 상위 레벨(High-Level)이라고 부르며, 나머지 부분 (깊이 h+1에서 H까지)의 부분을 하위 레벨(Low-Level)이라고 부른다. 제안하는 방법에서는 상위 레벨에 해당되는 데이타 및 색인 정보를 전체적으로 혹은 부분적으로 (totally or partially) 반복시키기 때문에 '반복되는 부분(replicated parts)라고 표시하였다. 하위 부분은 반복되지 않는다 (non-replicated parts). ('h' 값에 따라 반복의 정도가 결정되며, 이 값을 어떻게 설정하는 것이 최적의 성능을 나타내는지에 대해서는 4장에서 설명한다.)

스트림 구성 방법을 설명하기 위하여, 프래그먼트(fragment)라는 개념을 사용한다. 스트림 상에서 연속하는 데이타들의 영역을 데이타 프래그먼트라고 하며, 연속된 색인 영역을 색인 프래그먼트라고 부른다. 데이타 프래그먼트와 색인 프래그먼트는 담당하는 영역의 위치에 따라 상위 레벨 및 하위 레벨 데이타 프래그먼트와 색인 프래그먼트로 구분된다.

정의 1.

- 하위 레벨 데이타 프래그먼트(LD: Low-level Data fragment): XML 데이타 트리에서 깊이 'h'에 존재하는 각 노드를 루트로 하는 서브트리에 해당하는 데이타 노드들이 연속 할당되는 스트림 구간. (그림 4(a)에서 아래에 위치한 작은 삼각형들에 해당된다. 루트 노드는 제외되며, DFS 방법으로 순회한 결과로 정렬된다.)
- 상위 레벨 데이타 프래그먼트 (HD: High-level Data fragment): 하위 레벨 데이타 프래그먼트에 해당되는 접근 경로 상의 데이타 노드들이 할당되는 스트림 구간. (그림 4(a)에서 가늘고 긴 사각형 모양에 해당함.)
- 하위 레벨 색인 프래그먼트 (LI: Low-level Index fragment): 각 LD에 포함되는 데이타 노드들에 대한 색인 정보, 즉 각 노드의 경로 정보와 시작 노드 및 종료 노드의 주소 값들이 할당되는 스트림 구간. (그림 4(b)에서 아래 삼각형에 해당한다.) 각 노드에 대한 경로식에서 상위 레벨 색인에서 지정한 나머지 부분만 표현한다.
- 상위 레벨 색인 프래그먼트 (HI: High-level Index fragment): 레벨 깊이 h까지의 서브 트리에 포함되는 데이타 노드들에 대한 색인 정보들이 할당되는 스트림 구간으로, 그림 4(b)에서 윗부분의 큰 삼각형에 해당한다.

그림 4(a)는 b1~b9를 꼭지점으로 하는 9개의 삼각형이 있으므로, 9개의 하위 레벨 데이타 프래그먼트(LD1~LD9)가 존재한다. 각 LD 대한 색인 정보들로 구성한 하위 레벨 색인 프래그먼트 역시 9개 (LI1~LI9) 존재한다. 상위 레벨 데이타 프래그먼트는 각 LD에 이르는 경로 상에 존재하는 데이타 노드들이 된다. 각각 HD1~HD9라고 부르며, 그림 4(a)에서 긴 사각형 모양에 해당된다. HD들 사이에는 경로의 일부가 서로 중복된다. 상위 레벨 색인 프래그먼트는 그림 4(b)에서 위의 큰 삼각형으로 표현되는 부분으로, 'h'깊이의 서브트리에 포함되는 노드들에 대한 색인 정보들이다.

예제 1. 그림 4를 기준으로 HD 및 LD, 그리고 HI

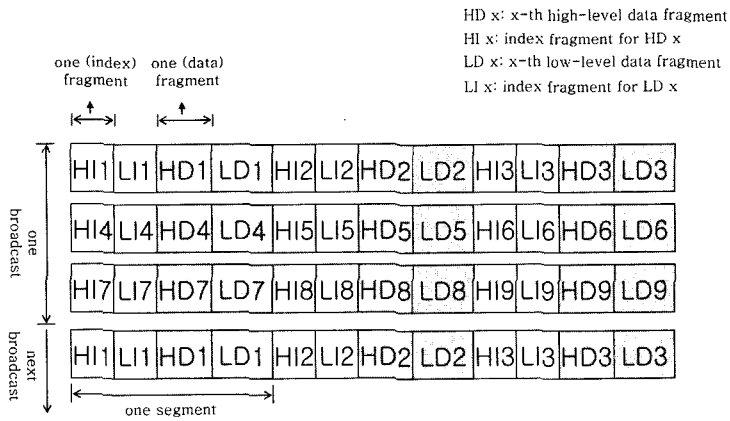


그림 5 그림 4의 XML 데이타에 대한 스트림 구성

및 LI는 다음과 같은 내용으로 구성된다. (여기에서 s_addr 및 e_addr은 스트림 상에서 해당 시작 태그 및 종료 태그가 나타나는 주소 (시간) 값을 나타낸다.)

- HI = {("root", s_addr, e_addr), ("root/a1", s_addr, e_addr), ("root/a1/b1", s_addr, e_addr), ("root/a1/b2", s_addr, e_addr), ("root/a1/b3", s_addr, e_addr), *("root/a2", s_addr, e_addr), ..., ("root/a3",* s_addr, e_addr), ..., ("root/a3/b9", s_addr, e_addr) }
- LI1 = {("c1", s_addr, e_addr), ("c2", s_addr, e_addr), ("c3", s_addr, e_addr)}[1]
- LI2 = {("c4", s_addr, e_addr), ("c5", s_addr, e_addr), ("c6", s_addr, e_addr)}
- .....
- HD1 = "<Root> <a1> <b1> ..... </b1> </a1> </Root>"[2]
- HD2 = "<Root> <a1> <b2> ..... </b2> </a1> </Root>"
- .....
- LD1 = "<c1> ..... </c1> <c2> ..... </c2> <c3> ..... </c3>"
- LD2 = "<c4> ..... </c4> <c5> ..... </c5> <c6> ..... </c6>"
- .....

프래그먼트 단위로 구성된 데이타들을 스트림으로 구성하는 방법은 다음과 같다.

1. 프래그먼트들을 DFS 방법으로 탐색하여 스트림을 구성한다. 이때, 상위 레벨 프래그먼트는 하위 레벨 프레그먼트가 배치될 때 반복 배치된다. 하나뿐인 HI가 9개의 하위 레벨 프레그먼트가 배치될 때마다 반복되어 *HI1~HI9*로 표현되었다.[3]
2. 각 데이타 프래그먼트 앞에는 해당 데이타 프래그먼트에 대한 색인 프래그먼트가 선행한다.

위의 프래그먼트 구성 방법에 따라 그림 4를 따르는 XML 데이타를 스트림으로 구성한 결과는 그림 5와 같다. 그림에서 H는 상위 레벨, L은 하위 레벨을 나타내며, I는 색인, D는 데이타를 나타낸다. 하나의 HI, LI, HD, LD 묶음을 세그먼트(segment)라고 부르며, XML 데이타 트리에서 세그먼트 개념은 'h' 길이 만큼의 경로와 경로 끝에 존재하는 노드를 루트로 하는 'H-h'높이의 서브트리에 해당한다. 색인 측면에서도 동일한 개념이지만, 자신이 아닌 다른 세그먼트를 검색하는 질의를 위해 상위 색인에서 다른 세그먼트에 대한 색인 정보를 추가하고 있어 윗 부분이 경로가 아닌 서브트리(삼각형 모양)라는 점이 차이점이다(그림 6 참조).

3.3 색인 및 데이타 버켓

무선 스트림 데이타는 버켓(bucket)이라 불리는 단위로 전송된다. 이는 디스크의 *페이지(page)*에 해당하는 개념으로 데이타 전송 및 수신의 기본단위가 된다. 무선 데이타 스트림에 색인 정보를 같이 전송할 경우, 데이타와 색인 정보를 구분하여 버켓을 구성해야 하는데, 이를 위해 본 논문에서는 다음과 같은 필드 정보를 사용하여 데이타 버켓과 색인 버켓을 정의한다.

정의 2. 색인 프래그먼트를 할당하는 색인 버켓은 다음과 같은 필드들로 구성된다.

- boolean BUCKET_TYPE: 색인 버켓 혹은 데이타 버켓 여부를 나타내는 필드

1) 하위 레벨 색인 세그먼트에 나타나는 경로 정보들은 해당 상위 레벨 색인의 마지막 경로 정보에서 이어지는 경로 정보만을 기술한다. 즉, "c1"이 나타내는 경로는 "root/a1/b1/c1"이다.

2) 해당 노드들에서 포함되는 속성 및 텍스트 등의 내용들이 모두 포함된다.

3) 각 HI가 스트림에서의 위치가 다르기 때문에, HI 내부에 나타나는 주소 값들은 상이할 수 있다.

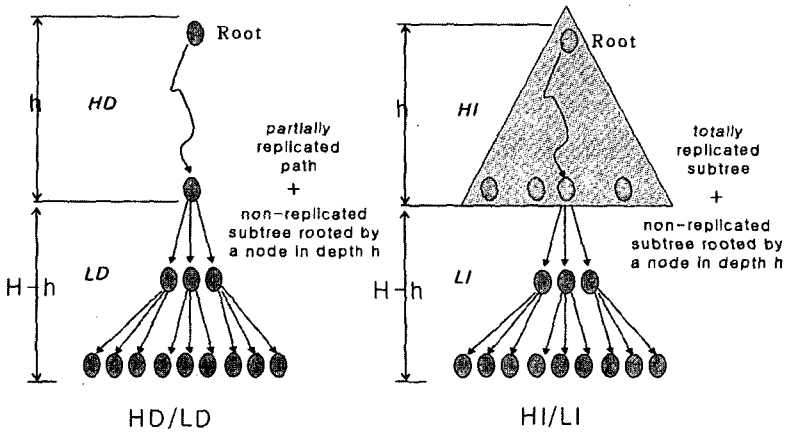


그림 6 XML 트리에서 하나의 세그먼트가 나타내는 데이타 및 색인의 범위

- boolean START_INDEX: 색인 프래그먼트를 구성하는 첫 번째 색인 버켓인지 아닌지를 나타내는 필드
- char ADDRESS[B[4]]: 색인 정보(특정 노드를 나타내는 경로와 해당 노드의 시작/종료 주소)가 저장되는 배열
- integer LINK_TO_NEXT_INDEX: 현재 버켓에서 가장 가까운 다음 색인 프래그먼트의 첫 번째 색인 버켓에 대한 주소 필드

정의 3. 데이타 프래그먼트를 할당하는 데이타 버켓 구성은 다음과 같다.

- boolean BUCKET_TYPE: 색인 버켓 혹은 데이타 버켓 여부를 나타내는 필드
- boolean START_DATA:데이타 프래그먼트를 구성하는 첫 번째 버켓인지 아닌지를 나타내는 필드
- char DATA[B]: XML 데이타가 저장되는 배열이다.
- integer LINK_TO_NEXT_INDEX: 현재 버켓에서 가장 가까운 다음 색인 프래그먼트에 대한 주소 필드

버켓의 크기 값에 따라 색인 및 데이타 프래그먼트를 저장하는데 하나의 버켓으로 충분한 경우도 있지만, 여러 개의 버켓에 걸쳐서 저장되어야 하는 경우도 발생한다. 하나의 색인 및 데이타 프래그먼트가 여러 버켓에 할당되는 경우, 프래그먼트의 첫 번째 버켓인지 여부를 판별하기 위해 사용되는 필드가 "START_INDEX"와 "START_DATA"이다.

무선 대역폭의 효과적인 사용을 위해, 실제 시스템 구현시에는 상위 레벨 색인 정보와 하위 레벨 색인 정보를 연속적으로 배열한 다음 버켓 단위로 할당하는 방법을 사용할 수도 있다. 상위 레벨 데이타와 하위 레벨 데이타에 대한 경우도 마찬가지이다. 본 논문에서는 내용 전개의 편의성을 위해 레벨 단위로 구분하여 설명하고 있다.

3.4 스트림 접근 방법

본 논문에서 제안한 XML 스트림 색인 기법을 사용하여 구성된 스트림을 클라이언트가 접근하기 위한 방법은 다음과 같다. 사용자 입력은 경로식으로 주어지며, 경로식은 단일 노드를 찾는 경로로 가정한다. "root/*/b4"나 "root//" 등과 같이 다수의 노드를 검색하는 경로식의 경우, 단위 노드를 검색하는 경로식의 집합으로 처리할 수 있다.

스트림 접근 과정은 다음과 같다. 사용자가 질의를 한 시점을 기준으로 현재 버켓을 수신한 다음, 색인 프래그먼트의 시작을 나타내는 버켓인 경우에는 곧 바로 경로 탐색에 들어갈 수 있지만, 그렇지 않은 경우에는 현재의 위치에서 가장 가까운 시간에 전달될 색인 프래그먼트의 시작 주소로 이동한다. 모든 버켓에는 가장 가까운 (미래의 방향으로) 색인 프래그먼트의 시작 주소 값을 저장하고 있기 때문에 가능하다. 색인 프래그먼트의 시작에는 상위 레벨 색인(HI)이 위치하므로, 사용자가 입력한 경로와 부합하거나 사용자 질의 경로를 포함하는 경로를 반드시 하나 이상 찾을 수 있다. HI에서 가장 깊은 경로 정보가 사용자의 입력 경로를 포함하는 경우, LI 버켓들을 읽어서 해당 노드들이 위치한 데이타 버켓들의 주소를 확인한 후, 주소 값을 사용하여 데이타를 수신하면 질의 결과를 생성할 수 있다.

3.5 스트림으로부터 원시 XML 문서의 생성

제안하는 색인 및 스트림 구성 방법은 원시 XML 문서의 내용에 대한 수정을 전혀 가하지 않는다. 다만, 상위 레벨 데이타 프레그먼트들에 해당하는 부분에서 겹치는 경로상의 노드들에 대한 데이타를 중복시킬 뿐이

4) 버켓의 크기에서 3개의 다른 필드를 위한 공간을 제외한 크기로 결정된다. 즉, 하나의 버켓에서 순수하게 데이타 정보와 색인 정보가 저장되는 공간의 크기이다.

```
ALGORITHM XML_Stream_Access
INPUT: a path expression P
OUTPUT: a set of nodes Result
VARIABLES: temp, t

BEGIN
        Set Result empty;
        temp <-- read the current bucket from the wireless stream;
        IF (temp is a data bucket) THEN {      //Using the BUCKET_TYPE field
                Wait until the next index bucket arrives on the air;
                // Using the LINK_TO_NEXT_INDEX field
                GOTO INDEX_PROBE;
        } ELSEIF (temp is not a starting index bucket) THEN {
                // Using the START_INDEX field
                Wait until the next index bucket arrives on the air;
                // Using the LINK_TO_NEXT_INDEX field
                GOTO INDEX_PROBE;
        }

INDEX_PROBE:
        // Currently, A HI index bucket arrives.
        temp <-- read the current index bucket;
        Find the longest path t among the index entries in temp;
        Find the best matched path with P;
        Probe to a corresponding LI fragment;
        Get the address of target data bucket(s);


DATA_DOWNLOAD:
        FOR (all target addresses) {
                Probe to the target address;
                temp <- read the current bucket;
                IF (temp is a data bucket) THEN {
                        Result <-- Result + DATA array in temp;
                        // '+' denotes "Embedding and Concatenation"
                }
        }

END_OF_ALGORITHM
```

그림 7 제안하는 색인과 데이타로 구성된 스트림을 위한 접근 알고리즘

다. 따라서, 스트림으로 전송되는 데이타를 통하여 원시 XML 문서를 완전하게 생성할 수 있다. 원시 문서를 생성하는 방법을 간단히 나타내면 다음과 같다.

HD 1 + LD 1 + ***Remove_Replicated_Data****(HD 2) + LD 2 +* ***Remove_Replicated_Data****(HD 3) + LD 3 + ...*

위 표현식에서 ***Remove_Replicated_Data()***는 HD들 사이에서 중복되는 부분을 제거하는 함수이며, '+' 표시는 뒤의 XML 문서를 앞의 XML 문서의 적절한 위치에 내포(embed)시키는 연산자이다. 가령, HD 1이 "<a>   "이고, LD 1이 "<c> </c>"인 경우, 그리고 LD 1이 내포될 위치가 태그의 자식에 해당한다면, HD 1 + LD 1은 "<a> <c> </c> "가 된다. 논문에서 가정하였듯이, XML 노드들의 내용 수정 없이, 단순 조합으로 원시 XML 문서를 복원할 수 있다.

4. 분 석

이 장에서는 제안하는 색인 및 스트림 구성 기법을 사용하는데 따르는 성능 요소들에 대하여 분석한다. 색인 정보는 스트림으로 전달되는 데이타를 에너지 효과적으로 검색하기 위한 방법으로, 일반적으로 튜닝 시간은 효과적으로 감소시키지만, (색인의 크기 만큼 검색하고자 하는 데이타가 지연되기 때문에) 데이타를 접근하는데 필요한 접근 시간은 오히려 증가시키게 된다. 따라서, 접근 시간의 증가와 튜닝 시간의 감소를 적절히 중재하는 방향으로 색인을 구성하여야 한다.

예를 들어 전혀 색인을 사용하지 않을 경우에는 다음과 같은 튜닝 시간과 접근 시간이 소요된다. XML 데이타가 차수(degree 또는 fan-out)가 n이고 높이(height)가 H인 완전 균형 트리 형태로 표현되고, 각 노드의 크기가 $\nabla data$라고 하자. 전체 데이타의 크기($\nabla DATA$)는 다음과 같다.

$$\nabla DATA = \nabla data \times \frac{n^H - 1}{n - 1} \quad (1)$$

따라서, 임의의 데이타 노드를 찾는데 필요한 평균 접근 시간 및 평균 튜닝 시간은, 전체 방송 스트림의 절반 크기인 $0.5 \times \nabla DATA$에 $\nabla data$를 더한 값이 된다. (실제 무선 시스템에서는 버켓 단위로 전송이 이루어지므로, 버켓의 크기를 이 식에서 나누어, 수신하는 버켓의 개수

로 표현될 것이다. 이 장에서는 계산에 중점을 두고 있으므로 버켓의 크기로 나누는 작업을 생략하고, 데이타의 크기 단위로 설명한다.) 여기에서 앞부분($0.5\times\nabla DATA$)은 해당 데이타를 찾는 과정에 소요되는 접근 시간 및 튜닝 시간이며, 뒷부분($\nabla data$)은 실제 데이타를 수신하는데 사용되는 시간이다. 어떠한 색인 방법을 사용하든지, 뒷부분의 $\nabla data$는 동일하므로, 앞으로의 분석에서는 해당 데이타를 찾는 과정에서 소요되는 접근 시간과 튜닝 시간만을 고려하기로 한다. (참고로, 2장에서 소개한 전체 색인을 한번만 배치하는 '단순 색인 구조'는 본 논문에서 제안한 방법에서 h=0인 특별한 경우에 해당한다.)

4.1 색인의 양에 대한 분석

먼저, 제안하는 색인 구조를 사용하는 경우 요구되는 색인 정보의 양을 계산해 보자. 그림 4에서 HI 색인과 LI 색인의 크기를 계산하는 것으로, 모든 HI의 크기가 동일하며, 또한 모든 LI의 크기가 동일하므로, 하나의 HI 색인의 크기를 ∇HI, 하나의 LI의 크기를 ∇LI라고 하면, 전체 색인의 양(∇INDEX)은 다음과 같이 계산할 수 있다.

$$\nabla INDEX = n^h(\nabla HI + \nabla LI) = n^h(f(h)) + g(H-h))$$

이 식에서 h는 상위 레벨의 깊이를 나타내는 값으로, H>=h>=0을 만족하여야 하며, f(h)는 높이가 h인 서브트리에 해당하는 HI를 나타내는데 필요한 색인의 양이며, g(H-h)는 높이가 H-h인 서브트리인 LI를 나타내는데 필요한 색인의 양이다. 상위 레벨 깊이가 h인 경우 LI 및 HI의 개수는 각각 n^h이다.

하나의 색인 튜플이 대상 노드를 나타내는 경로 정보와 해당 노드의 시작 및 종료 태그 주소를 나타내므로, 하나의 색인 튜플을 표현하는데 필요한 저장 공간은 경로 길이에 비례하는 선형 함수로 볼 수 있다. 따라서, f(), g()를 해당 색인에 포함되는 노드의 수와 각 노드의 경로 깊이를 고려한 식으로 표현하면, 위 식은 다음과 같이 나타난다. (여기에서 k는 경로 길이가 1인 하나의 색인 튜플을 저장하는데 필요한 저장 공간을 나타내는 선형 함수의 계수이다.)

$$\begin{aligned}\nabla INDEX &= n^h(f(h)+g(H-h)) \\ &= n^h\left[k\sum_{i=1}^{h} n^i + k\sum_{i=1}^{H-h} n^{i-1}\right] \\ &= kn^h\left[\frac{n^h-1}{n-1} + \frac{n^{H-h}-1}{n-1}\right] \end{aligned} \quad (2)$$

HI와 LI 모두 트리 형태이므로, 트리에 포함되는 노드들의 개수를 구하면 등비수열의 합이 된다. 각 트리의 높이가 'h'와 'H-h'이므로 위와 같은 식이 생성된다.

4.2 접근 시간에 대한 분석

색인을 사용하여 구성된 스트림에서 특정 노드를 검색하는데 소요되는 접근 시간은, 찾고자 하는 데이타의 위치와 검색을 시작한 시점 사이의 순서에 따라 결정된다. 가장 접근 시간이 짧은 경우는 현재 수신한 버켓이 HI의 첫 버켓이고, 해당 세그먼트에 검색 대상 데이타가 존재하는 경우이며, 접근 시간은 ∇HI+∇LI이 된다.[5] 반면 접근 시간이 가장 긴 경우는, 처음 수신한 버켓이 위치한 세그먼트에 검색 대상 데이타가 존재하지만, 현재 수신한 버켓이 HI의 두 번째 버켓인 경우이다. 즉, 접근 시간이 가장 짧은 경우보다 한 버켓 지나서 질의를 시작하는 경우이다. 찾고자하는 데이타가 현재 세그먼트에 있음에도 불구하고, 그 데이타를 찾는데 필요한 색인을 찾을 수 없기 때문에 다음 브로드캐스트로 이동해서 루트 색인부터 찾아야 한다. 따라서, 접근 시간은 ∇HI + ∇LI + ∇DATA + ∇INDEX -1이 된다. 평균 검색 시간을 계산하면 다음과 같다. ('∇HI + ∇LI + ∇DATA'는 브로드캐스트 스트림 하나의 크기이며, '∇INDEX -1'는 루트 색인부터 말단 색인까지의 접근 시간에 처음에 읽은 색인 버켓 1을 뺀 크기이다.)

$$\begin{aligned} AvgAccessTime &= \frac{1}{2}\left[(\nabla HI+\nabla LI)+(\nabla HI+\nabla LI+\nabla DATA+\nabla INDEX)-1\right] \\ &= (\nabla HI+\nabla LI)+\frac{1}{2}(\nabla INDEX+\nabla DATA)-\frac{1}{2} \\ &= (f(h)+g(H-h))+\frac{1}{2}\left[n^h(f(h)+g(H-h)+f'(h)+g'(H-h))\right]-\frac{1}{2} \\ &= (\frac{1}{2}n^h+1)(f(h)+g(H-h))+\frac{1}{2}n^h(f'(h)+g'(H-h))-\frac{1}{2} \\ &= (\frac{1}{2}n^h+1)k(\frac{n^h-1}{n-1}+\frac{n^{H-h}-1}{n-1})+\frac{1}{2}wn^h(h+\frac{n^{H-h}-1}{n-1})-\frac{1}{2} \end{aligned} \quad (3)$$

위 식에서 f'()와 g'()는 HD 및 LD 프래그먼트에 대한 식이다. f()의 경우 트리형태로 표현되는 부분의 노드들의 개수이지만, f'()는 LD에 이르는 경로상의 노드들이므로, 경로의 길이 만큼의 노드들만 계산하면 된다. f(), g()와 마찬가지로 f'()와 g'()역시 노드의 개수에만 의존하며, w라는 각 노드별 크기에 비례하는 선형 함수로 표현될 수 있다. 즉, f'(h) = w*h, g'(H-h) = w*$(n^{(H-h)}-1)/(n-1)$이다.

4.3 튜닝 시간에 대한 분석

색인을 사용하여 구성된 스트림에서 특정 노드를 검색하는데 소요되는 튜닝 시간 계산은 다음과 같다. 먼저 하나의 버켓은 반드시 읽어야 하며, 그 버켓의 내용을 판단한 다음 가장 가까운 HI의 첫 번째 버켓으로 이동하게 된다. (만일 처음 읽은 버켓이 HI의 첫 버켓이라면 이동할 필요가 없다.) 그런 다음 HI 및 LI의 색인 프래그먼트를 튜닝하게 되면 모든 검색 대상 데이타의

5) 데이타 프레그먼트의 처음에 검색 대상 데이타가 있다고 가정한다.

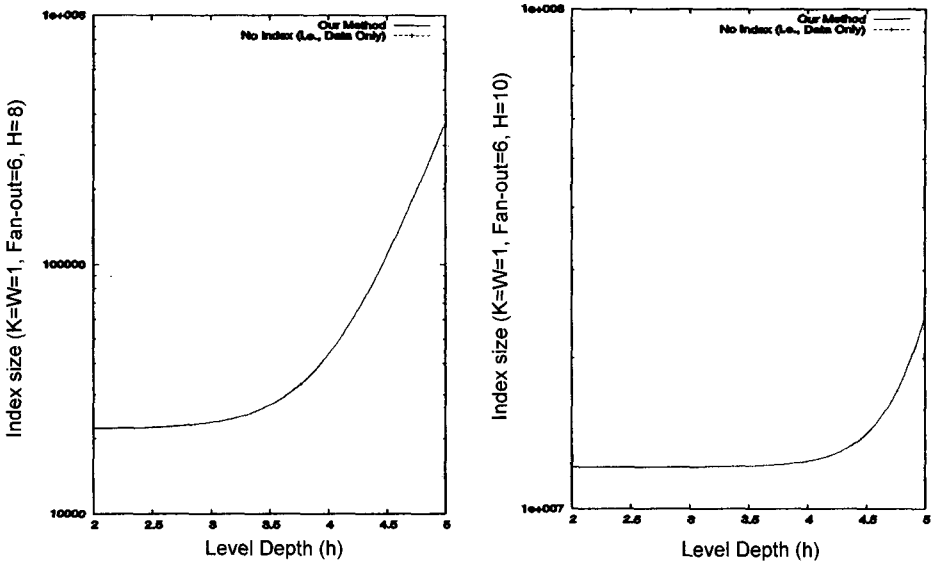


그림 8 상위 레벨 깊이에 따른 색인 정보의 크기

위치를 찾을 수 있다.[6] HI의 첫 버켓에서 수신을 시작하는 드문 경우를 배제하면 튜닝 시간은 다음과 같다.

$$AvgTuningTime = 1 + \nabla HI + \nabla LI$$
$$= 1 + f(h) + g(H-h)$$
$$= 1 + k\left(\frac{n^h - 1}{n-1} + \frac{n^{H-h} - 1}{n-1}\right) \quad (4)$$

4.4 최적의 상위 레벨 깊이(h)

상위 레벨을 결정하는 값 h에 따라 색인의 양이 결정되고, 또한 색인의 양과 배치에 따라 접근 시간과 튜닝 시간이 달라진다. 그림 8은 트리의 차수(n)와 상위 레벨의 깊이 값(h)에 따라 색인의 양을 표현한 그래프이며, 그림 9와 그림 10은 각각 평균 접근 시간과 평균 튜닝 시간을 나타낸 그림이다. XML 데이타를 나타내는 트리의 차수가 4와 6, 그리고 트리의 높이(H)가 8과 10인 경우에 대해 실험한 결과이며, 편의상 상수 k, w와 ▽data의 값을 1로 설정하였다.

식 (2)를 사용하여 색인의 양을 측정한 그림 8에서 나타난 결과를 보면, 색인을 사용하지 않고 데이타만을 스트림으로 구성한 경우, 당연히 색인의 크기는 '0'을 나타낸다. 그리고 제안한 색인 기법은 레벨의 깊이에 따라, 지수 함수 형태로 색인의 크기가 증가함을 알 수 있다. 무선 통신에서 대역폭은 매우 제한적인 자원이기 때문에, 대역폭의 남용은 시스템 전체 성능을 크게 저하시킬 수 있으므로, 적절한 색인의 크기를 사용하는 것이 중요하다.

그림 9는 상위 레벨 깊이 값에 따른 평균 접근 시간 식 (3)의 분포를 나타낸다. 색인을 사용하지 않고 데이타만으로 스트림을 구성한 경우는 전체 데이타 크기의 절반에 해당하는 $0.5*(n^H-1)/(n-1)$ 만큼의 접근 시간을 필요로 한다. 그래프에서는 상대적인 크기 차이로 그래프 하단에 붙어서 표기되었다. 접근 시간은 일정 수준에서 최소 값을 나타낸 후, 지수 함수 형태로 접근 시간이 증가하는 형상을 보이고 있다. 제안하는 색인 사용 시 평균 접근 시간을 최소화하는 레벨의 깊이는 식 (3)을 최소화하는 'h' 값을 구하여 결정할 수 있다.

그림 10은 평균 튜닝 시간 (식 4)의 결과로서 H/2를 중심으로 하는 좌우 대칭의 그래프가 나타난다. H/2에서 최소의 튜닝 시간을 보인다. 상대적으로 색인을 사용하지 않고 데이타만 스트림으로 구성하는 경우의 튜닝 시간은 접근 시간과 동일하기 때문에, n=4, H=8인 좌측 그래프의 경우에는 약 10,100의 평균 튜닝 시간이 요구되며, n=6, H=10인 우측 그래프에서는 약 6,050,000의 평균 튜닝 시간이 요구된다. 제안하는 색인 방법은 최소 튜닝 시간이 각각 약 171, 1556이다 (식 4의 대입 결과). 트리의 규모가 커지면서 색인을 사용하지 않는 경우의 튜닝 시간은 지수 함수 형태로 증가하지만, 제안하는 방법의 경우, 색인의 효율적인 배치로 튜닝 시간이 현저히 줄어든다.

식 (4)와 달리, 식 (3)의 경우 미분한 형태가 지수 함수와 다항 함수가 혼합된 형태이기 때문에 분석적(analytical)인 방법으로 최적의 'h'값을 구할 수 없지

6) 색인 프래그먼트를 읽는 도중에 검색 대상 데이타의 주소 값을 알아 낼 수 있지만, 분석의 편의상 색인 정보를 모두 읽어야 하는 경우를 가정한다.

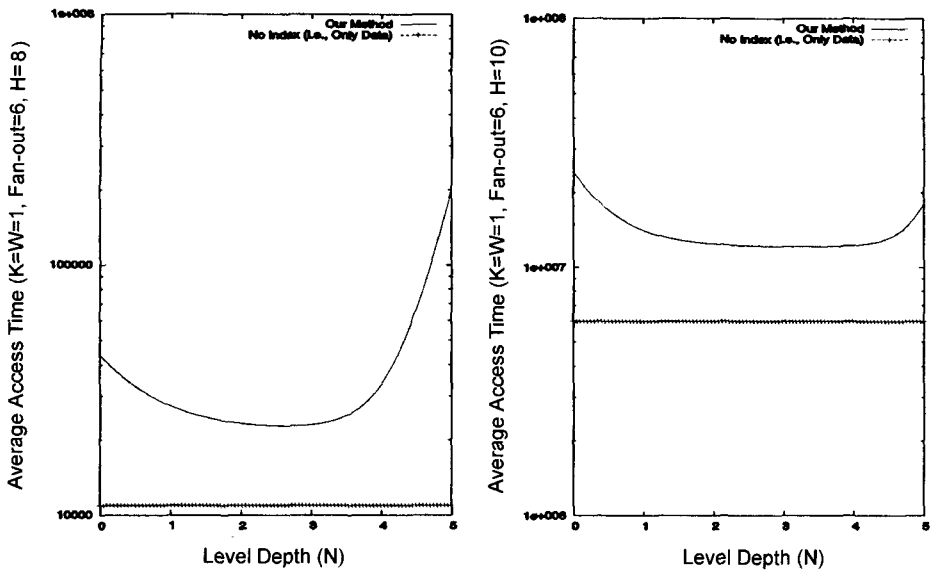


그림 9 상위 레벨 깊이에 따른 평균 접근 시간

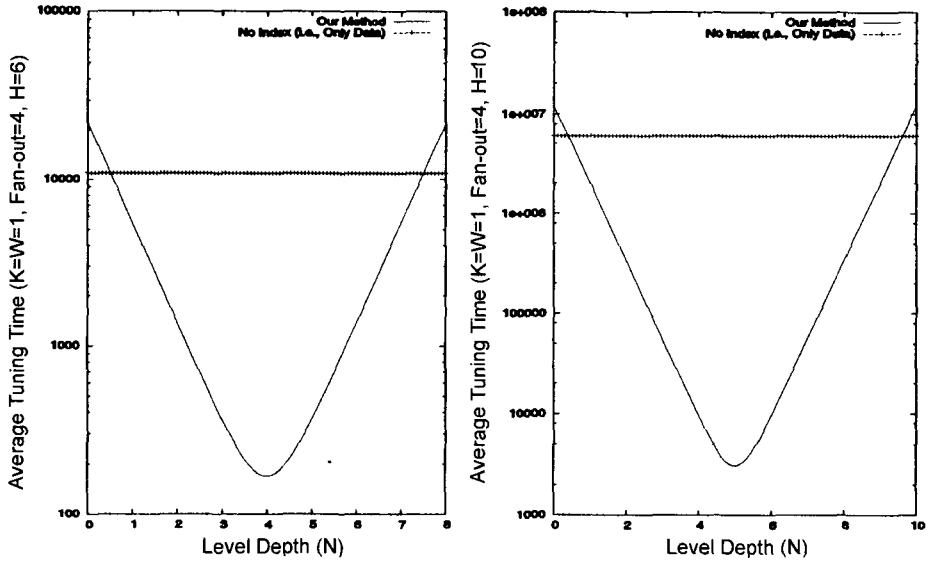


그림 10 상위 레벨 깊이에 따른 평균 튜닝 시간

만, 우리가 찾고자 하는 'h'값이 정수 형태이어야 하기 때문에, 그림 11과 같은 알고리즘을 사용하여 수치적인 (numerical) 방법으로 최적의 'h' 값을 쉽게 구할 수 있다. 따라서, 최적화하고자 하는 대상에 따라 (평균 접근 시간 또는 평균 튜닝 시간) 해당 분석식을 최소화하는 'h' 값을 구하여 레벨을 설정할 수 있으며, 시스템 요구 조건에 따라, "a * 평균접근시간 + b * 평균튜닝 시간"와 같이 가중치를 둔 평균 시간의 합을 최소화하고자 하는 경우에도, 분석 식들을 가중 합(weighted sum) 형태로 만든 후, 이 식을 최소화하는 상위 레벨 깊이 값 'h'를 구할 수 있다.

5. 결 론

본 논문에서는 무선 정보 시스템 환경에서 XML 데이타를 스트리밍 서비스할 경우 필요한 색인 기법을 제안하였다. XML은 최근 급속히 그 응용 분야를 넓혀가고 있는 데이타 표현 및 전송 표준 기법으로, 무선 정보 시스템에서도 그 사용이 활발해 질 것으로 예상되므로,

```
ALGORITHM Selection_Of_Optimal_h
INPUT: target metric M (or weighted combination of metrics), fan-out N and tree height H
OUTPUT: optimal value of h
VARIABLES: temp, count

BEGIN

        Let F be the equation of the target metric (or combination of target equations);
        temp = F(n, H);
        count = int (H/2) + 1;
        WHILE (temp > F(n, count)) DO
                temp = F(n, count);
                count = count - 1;
        END-WHILE
        return count;

END_OF_ALGORITHM
```

그림 11 최적의 상위 레벨 깊이(h)를 결정하는 알고리즘

XML 스트리밍을 위한 기반 기술인 색인의 필요성은 매우 높다고 할 수 있다.

기존의 무선 데이타 스트리밍에 대한 색인 기법은 동일한 구조의 레코드 집합을 스트리밍 하는 경우를 고려한 방법이다. 각 레코드의 식별자를 이용하여, 식별자들을 계층구조 형태로 색인을 구성하는 방법을 사용하였다. 하지만, XML은 데이타 자체적으로 구조적인 성격을 지니고 있으며, 데이타를 나타내는 식별자의 역할을 하는 경로식의 사용을 고려하여야 한다.

본 논문 제안하는 방법은 스트림 구성시 원시 XML 데이타의 내용에 대한 수정을 가하지 않으며, 부분 색인의 효과적인 반복 배치를 통해 XML 스트림에 대한 접근 성능을 효과적으로 개선한다. (접근 시간의 과도한 증가없이 튜닝 시간을 최소화 시킨다.) 부분 색인의 반복 배치를 위하여 데이타와 색인 모두를 특정 레벨로 구분하여 스트림에서 반복 배치하도록 하였다.

제안한 방법의 접근 성능에 대해, 색인의 크기, 평균 접근 시간 및 평균 튜닝 시간에 대한 분석 식을 도출했으며, 이를 기반하여 시스템 환경에 적합한 최적의 상위 레벨 깊이 값을 도출하였다.

XML 데이타에 대한 사용자 질의는 주로 XPath등과 같은 경로식을 사용한다. 경로식을 사용하는 경우, *, ?, // 등과 같이 부분 부합 질의를 포함하는 경우가 빈번할 것이다. 본 논문에서는 각 노드 단위의 접근 성능을 고려하여 색인을 제안하였지만, 이러한 부분 부합 질의를 고려한 색인의 개발이 향후 필요할 것으로 판단된다. 또한, 본 논문에서는 사용자의 접근 패턴에 대한 분석이 불가능한 순수 방송(pure broadcasting) 환경을 고려하였지만, 혼합 방송(hybrid broadcasting)과 같이 사용자들과의 개별 통신이 존재하는 경우, [6]에서와 같이 데이타의 접근 패턴에 따라, 접근 성능을 차별화하는 색인 응용도 존재할 것이다. 이를 위해 Hot 데이타 노드들에 대한 접근 성능을 Cold 데이타에 비해 우수하게 제공하는 색인 기법의 연구도 필요하다.

참 고 문 헌

[1] T. Imielinski, S. Viswanathan, and B. R. Badrinath. "Data on Air : Organization and Access," *IEEE Transactions on Knowledge and Data Engineering*, 9(3), 1997.

[2] T. Imielinski and B. R. Badrinath. "Data Management for Mobile Computing," *SIGMOD RECORD*, 22(1), 1993.

[3] Tim Bray, et. al., Extensible markup language (XML) 1.0 second edition W3C recommendation. Technical Report REC-xml-20001006, World Wide Web Consortium, 2000.

[4] W3C. Document Object Model (DOM), http://222.w3.org, Feb. 2000.

[5] Don Chamberlin, et. al., XQuery: A Query Language for XML, W3C working draft. Technical Report WD-query-20010215, World Wide Web Consortium, 2001.

[6] S. Acharya, R. Alonso, M. Franklin, and S. Zdonik. "Broadcast Disks : Data Management for Asymmetric Communication Environments," In *Proceedings of ACM SIGMOD Conference*, pages 199-210, 1995.

[7] J. Jing, O. Bakhres, A. Elmagarmid, and R. Alonso. "Bit-Sequences : A Adaptive Cache Invalidation Method in Mobile Client/Server Environments," *Mobile Networks and Applications (MONET)*, 2(2):115-127, 1997.

[8] T. Imielinski, S. Viswanathan, and B. R. Badrinath, "Power Efficient Filtering of Data on Air," In Proceedings of Extending Database Technology, 1994.

[9] Y. D. Chung and M. H. Kim, "Effective Data Placement for Wireless Broadcast," Distributed and Parallel Databases, 9:133-150, 2001.

[10] J. Y. Lee, Y. D. Chung, Y. J. Lee and M. H. Kim. "Gray Code Clustering of Wireless Data for Partial Match Queries," Journal of Systems Architecture, 47:445-458, 2001.
[11] Y. D. Chung and M. H. Kim, "An Index Replication Scheme for Wireless Data Broadcasting," Journal of Systems and Software, 51(3), pp. 191-199, May, 2000.
[12] S. Bose and L. Fegaras, "Data Stream Management for Historical XML Data," In *Proceedings of ACM SIGMOD Conference*, pages 239-250, 2004.

정 연 돈
정보과학회논문지 : 데이타베이스
제 32 권 제 3 호 참조

이 지 연
정보과학회논문지 : 데이타베이스
제 32 권 제 3 호 참조