XML 데이타를 위한 EP2 레이블링 스킴

진주용[0], 배진욱, 이석호

서울대학교 전기컴퓨터공학부

{ dansun[0], oblody }@db.snu.ac.kr, shlee@cse.snu.ac.kr

EP2 Labeling Scheme for XML Data

Juyong Jin[0], Jinuk Bae, Sukho Lee

School of Electrical Engineering and Computer Science, Seoul National University

요약

범위 기반 레이블링 스킴(range-based labeling scheme)을 이용하면 임의의 두 노드에 대한 조상-자손 관계를 쉽게 판별할 수 있으므로, XPath나 XQuery 형태의 질의를 효율적으로 처리할 수 있다. 그러나 노드의 삽입이 일어나는 동적인 상황에서는 불가피하게 전체 또는 일부의 레이블을 다시 할당(re-labeling)할 가능성이 있다는 문제점이 있다. 본 논문에서는 Dietz 레이블링 스킴을 개선한 EP2(extended preorder & postorder) 레이블링 스킴을 제안한다. 제안하는 스킴은 동일한 저장 공간상에서 범위 기반 레이블링 스킴에 비해 동적인 갱신에 유리하며, 기존의 구조 조인 알고리즘(structural join algorithm)을 이용하여 효율적으로 구조 질의(structural query)를 처리할 수 있다.

1. 서론

XML[1]은 데이타 표현의 유연성으로 인해 인터넷 상에서 데이타를 교환하는 실질적인 표준으로 자리 잡고 있다. 또한 XML 문서가 담고 있는 데이타에 대해 질의를 표현할 수 있도록 XPath[2]나 XQuery[3] 등의 질의 언어도 고안되었다. XML 문서에 대한 질의를 효율적으로 처리하기 위해서는 문서 내의 임의의 두 노드에 대해 조상-자손 관계(ancestor-descendant relationship)를 쉽고 빠르게 판별할 수 있게 해주는 레이블링 스킴(labeling scheme)이 필요하다.

본 논문에서는 Dietz 레이블링 스킴[4]을 개선한 EP2(extended preorder & postorder) 레이블링 스킴을 제안한다. 제안하는 스킴은 같은 크기의 저장 공간 상에서 범위 기반(range-based) 레이블링 스킴에 비해 한 노드에 더 많은 자식 노드를 삽입할 수 있다. 따라서, 노드 삽입으로 인한 레이블의 재할당(re-labeling) 가능성을 보다 줄임으로써 동적인 갱신에 보다 유연하게 대처하는 것이 가능하다. 그리고 조상-자손 관계를 판별하는 방법이 범위 기반 스킴과 거의 유사하기 때문에, holistic twig 조인 알고리즘[5,6,7]과 같은 구조 조인 알고리즘(structural join algorithm)을 조금만 변형하면 빠르게 구조 질의(structural query)를 처리할 수 있다.

본 논문의 구성은 다음과 같다. 1장의 서론에 이어, 2장에서는 관련 연구에 대한 내용을 다룬다. 3장에서는 EP2 스킴에 대해 설명하고, 범위 기반 스킴과 비교한다. 4장에서는 EP2 스킴을 이용하여 조인 알고리즘을 처리하는 방법에 대해 설명하고, 5장에서 결론을 맺는다.

2. 관련 연구

XML 문서에 대한 질의를 효율적으로 처리하기 위한 레이블링 스킴에 관한 기존 연구로서는 범위 기반(range-based)[8,9], 프리픽스 기반(prefix-based)[10,11], 그리고 소수 기반(prime number)[12]의 스킴 등이 있다. 범위 기반 스킴에서는 각 노드에 order와 size 값을 할당하고, (order, order+size) 구간을 해당 노드에 대한 레이블로 할당한다.[8] 임의의 두 노드 x와 y에 대해 order(x) < order(y) < order(x)+size(x)의 관계가 성립한다면 그 두 노드는 조상-자손 관계가 된다. [9]에서는 order 대신에 begin, order+size 대신에 end를 사용하여 보다 직관적으로 (begin, end) 사이의 구간(범위)이 레이블로 할당되는 것을 설명하였다. 범위 기반 스킴은 레이블의 크기가 전체 노드의 개수에만 영향을 받을 뿐, 문서의 팬아웃(fan-out)이나 깊이에 무관하기 때문에, 상대적으로 작은 크기의 레이블을 만들 수 있다는 장점이 있다.[12] 그러나, 노드의 삽입이 빈번하게 일어나는 경우에는 불가피하게 전체 또는 일부의 레이블을 다시 할당해야 하는 문제가 발생할 수 있다.

프리픽스 기반 스킴은 문서의 동적인 갱신이 빈번한 경우에 적합하다. 모든 노드의 레이블은 자신의 부모 노드에 할당된 레이블을 프리픽스로 가지고 있으며, 임의의 두 노드에 대한 조상-자손 관계는 한 노드의 레이블이 다른 노드의 레이블의 프리픽스인지 여부를 살펴보면 된다. 프리픽스 기반 스킴은 노드의 삽입에 제약이 없다는 장점을 가지고 있으나, 레이블의 크기가 노드의 전체 개수 뿐만 아니라, 문서의 팬아웃과 깊이에 비례해서도 증가하게 되는 문제가 있으며, 특히 팬아웃이 클 경우에는 레이블의 크기가 비효율적으로 커진다. 따라서, XPath나 XQuery 등의 질의를 처리하기 위해 더 많은 디스크 I/O를 발생시켜야 하므로 질의 처리 시간이 증가하게 된다.

소수 기반 스킴은 프리픽스 기반 스킴과 마찬가지로 노드의 삽입에 아무런 제약이 없다. 게다가, 모든 노드의 논리적인 전체 순서(global order)를 쉽게 알 수 있기 때문에, 임의의 노드에 대한 Preceding, Following 노드를 찾는 순서 쿼리(order-sensitive query)에도 적합하다. 그러나, 삽입이 일어난 뒤에도 노드들의 물리적인 순서를 지속적으로 유지하기 위해서는 노드와 같은 개수의 order number 정보를 관리해야 하며, 이 정보와 레이블을 연결하는 SC table을 통해 레이블의 정렬을 수행해야 하므로 holistic twig 조인 알고리즘[5,6,7] 등을 이용해 구조 질의를 처리하기가 수월하지 않다. 또한, 레이블의 크기가 문서의 전체 개수뿐만 아니라 문서의 깊이에 비례해서도 증가하게 되므로 프리픽스 기반 스킴과 마찬가지로 범위 기반 스킴에 비해 질의 처리에 더 많은 시간이 소요된다.

* 본 연구는 2004년도 두뇌한국21사업과, 정보통신부의 대학 IT연구센터(ITRC) 지원을 받아 수행되었습니다.

3. EP2 레이블링 스킴

3.1. 표기법(Notation)

앞으로 사용될 기호들을 표 1에 정의하였다.

기호	의미
N_x	노드 x
pre(N_x), post(N_x)	N_x의 시작 태그(start tag)와 끝 태그(end tag)에 부여되는 번호
level(N_x)	문서 상에서 N_x의 트리 레벨
N_{x_pre1}, N_{x_pre2}	전위 순회(preorder traversal) 상에서 N_x의 바로 이전 노드와 다음 노드
N_{x_post1}, N_{x_post2}	후위 순회(postorder traversal) 상에서 N_x의 바로 이전 노드와 다음 노드
L_{max}, L_{min}	레이블이 가질 수 있는 최대값과 최소값

표 1. 앞으로 사용될 기호들

3.2. EP2 레이블링 스킴의 정의

EP2(extended preorder & postorder) 레이블링 스킴은 동적인 갱신을 고려하여 Dietz 레이블링 스킴[4]을 확장한 것이다. 아래의 그림 1(a)는 Dietz 스킴을 이용하여 XML 문서의 각 노드에 레이블을 할당한 예이다.[1]

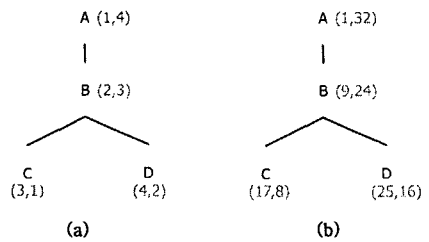

그림 1. (a) Dietz 레이블링 스킴, (b) EP2 레이블링 스킴

이 스킴은 전위 순회 순서로 시작 태그는 시작 태그 별로, 끝 태그는 끝 태그 별로 각각 1씩 증가하는 번호를 부여하며, 임의의 N_x에 대해 (pre(N_x), post(N_x))를 레이블로 할당한다. 이렇게 하면, 모든 노드가 시작 태그는 전위 순회 순서로, 끝 태그는 후위 순회 순서로 정렬되는 값을 얻게 되며, N_x에 대해

$$pre(N_{x\_pre1}) < pre(N_x) < pre(N_{x\_pre2})$$
$$post(N_{x\_post1}) < post(N_x) < post(N_{x\_post2}) \quad (1)$$

을 만족한다. 또한, N_x의 자손 N_y에 대해

$$(pre(N_x) < pre(N_y)) \wedge (post(N_y) < post(N_x)) \quad (2)$$

을 만족한다. 따라서, 임의의 두 노드에 대한 조상-자손 관계는 두 노드에 할당된 레이블 값을 비교하면 쉽게 판별할 수 있다. 그러나, Dietz 스킴은 추가적인 노드의 삽입이 발생하는 상황에서 레이블의 재할당(re-labeling)이 불가피하다는 단점이 있다.

이 문제를 개선하기 위해, Dietz 스킴을 개선한 EP2 레이블링 스킴을 제안한다. EP2 스킴은 Dietz 스킴과 마찬가지로 (1), (2)를 만족시키지만, 시작 태그와 끝 태그 별로 1씩 증가하는 대신 임의의 양수만큼 증가하게 된다. 즉, 임의의 양수 s_1, s_2와 N_x에 대해

$$pre(N_{x\_pre1}) + s_1 = pre(N_x)$$
$$post(N_{x\_post1}) + s_2 = post(N_x) \quad (3)$$

을 만족한다. 여기서 s_1, s_2는 인접한 노드들 사이에 남겨둔 여유 공간의 크기를 의미한다. 그림 1(b)는 $s_1=s_2=8$인 경우에, EP2 스킴을 이용하여 레이블을 할당한 예이다.

3.3. 새로운 노드의 삽입

EP2 레이블링 스킴에서 새로 삽입되는 노드 N_{new}에 할당되는 레이블은 다음과 같은 성질을 만족해야 한다.

$$pre(N_{new\_pre1}) < pre(N_{new}) < pre(N_{new\_pre2})$$
$$post(N_{new\_post1}) < post(N_{new}) < post(N_{new\_post2}) \quad (4)$$

pre(N_{new})와 post(N_{new}) 사이에 직접적인 상관 관계는 존재하지 않는다. 그림 2는 EP2 스킴에서 N_{new}의 레이블 값으로 취할 수 있는 여유 공간의 범위를 보여주고 있다.

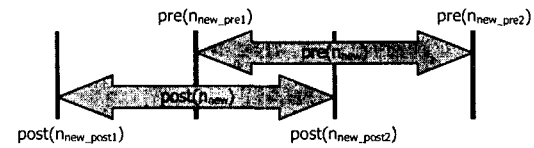

그림 2. 새로 삽입되는 노드 N_{new}의 레이블 할당 가능 범위

그리고, 이 여유 공간 내에서 다음의 휴리스틱(heuristic)에 따라 레이블 값을 결정한다.

$$pre(N_{new}) = pre(N_{new\_pre1}) + \frac{pre(N_{new\_pre2}) - pre(N_{new\_pre1})}{level(N_{new}) - level(N_{new\_pre2}) + 2}$$
$$post(N_{new}) = post(N_{new\_post2}) - \frac{post(N_{new\_post2}) - post(N_{new\_post1})}{level(N_{new}) - level(N_{new\_post1}) + 2} \quad (5)$$

N_{pre2}와 N_{post1}이 없는 경우에는 아래의 기본(default) 값을 취한다.

$$pre(N_{pre2}) = L_{max} + 1$$
$$post(N_{post1}) = L_{min} - 1$$
$$level(N_{pre2}) = level(N_{post1}) = 1$$

위 휴리스틱은 N_{new}를 삽입할 때 여유 공간을 어떻게 분할해야 하는지를 보여준다. 예를 들어, 그림 1(b)에서 노드 C의 자식으로 N_{new}를 삽입한다면, pre(N_{new})는 17과 25 사이, post(N_{new})는 0과 8 사이의 여유 공간에 있는 값을 가져야 한다. pre(N_{new})의 경우에는 전위 순회 상으로 N_{new}보다 작은 값을 가지는 1개의 노드(N_{new}의 왼쪽 형제 노드)와 큰 값을 가지는 3개의 노드(N_{new}의 자식 노드, N_{new}의 오른쪽 형제 노드, 노드 C의 오른쪽 형제 노드)가 추가로 삽입 될 수 있으므로 25와 37사이의 1/4 지점 값을 취한다. post(N_{new})의 경우에는 후위 순회 상으로 N_{new}보다 작은 값을 가지는 3개의 노드(N_{new}의 왼쪽 형제 노드, 노드 B의 왼쪽 형제 노드, 노드 A의 왼쪽 형제 노드)와 큰 값을 가지는 1개의 노드(N_{new}의 자식 노드)가 추가로 삽입 될 수 있으므로 0과 8사이의 3/4 지점 값을 취한다. 따라서 N_{new}는 (27, 6)을 레이블로 할당 받게 된다.

3.4. 범위 기반 레이블링 스킴과의 비교 분석

EP2 레이블링 스킴은 동적인 갱신이 일어나는 상황에서 범위 기반 레이블링 스킴보다 유리하다. 같은 저장 공간에 좀더 넓은 여유 공간을 만들 수 있기 때문이다. 이것을 보이기 위해, 문서 내의 모든 삽입 가능한 지점에 같은 확률로 삽입이 일어난다고 가정하자. 그러면, 모든 여유 공간의 크기가 같아야 한다. EP2 스킴에서의 이 공간을 $size_{EP2}$, 범위 기반 스킴에서의 이 공간을 $size_R$이라고 하면, 노드의 개수가 N일 때

$$size_{EP2} = (L_{max} - N)/N$$
$$size_R = (L_{max} - 2N)/2N \quad (6)$$

[1] 모든 레이블링 스킴에서 레이블 값은 양의 정수만 사용하는 것으로 가정한다.

이 된다.[2] 그림 3은 N=4, T=36인 경우에 $size_{EP2}$=8인 EP2 레이블링 스킴과 $size_R$=4인 범위 기반 레이블링 스킴을 수행한 결과이며, 표 2는 문서의 태그에 할당된 값들을 전위 순회 순서로 나열한 것이다.

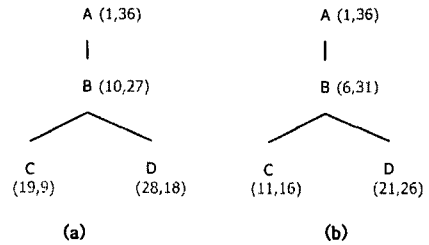

그림 3. (a) EP2 레이블링 스킴, (b) 범위 기반 레이블링 스킴

	<A>	<B>	<C>	</C>	<D>	</D>	</B>	</A>
EP2	1	10	19	9	28	18	27	36
범위 기반	1	6	11	16	21	26	31	36

표 2. 문서의 태그에 할당된 값들

그림 4는 $T=2^{32}$일 때, N에 따른 $size_{EP2}$와 $size_R$을 비교한 것이다. EP2 스킴과 범위 기반 스킴은 여유 공간의 크기에서 약 2배의 차이를 보이고 있다. 또한, 범위 기반 스킴은 삽입이 일어나는 해당 여유 공간에서 begin과 end 두 값을 모두 얻어야 하므로 실제 최대 레이블 할당 개수의 차이는 더 커지게 된다.

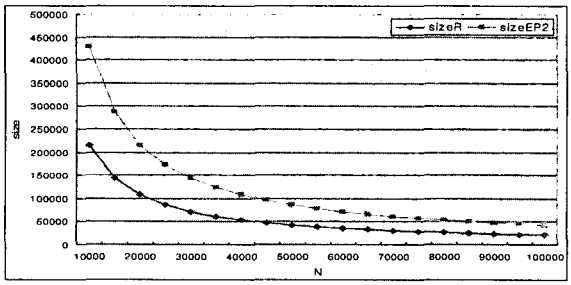

그림 4. EP2 레이블링 스킴과 기존 레이블링 스킴의 size 비교

예를 들어, 표 2에서 노드 B의 자식으로 </C>와 <D> 사이에 새로운 노드 N_{new}를 추가 한다면 EP2 스킴에서는 19와 28사이의 한 값을 $pre(N_{new})$에 할당하고, 9와 18사이의 한 값을 $post(N_{new})$에 할당하면 되지만, 범위 기반 스킴에서는 16과 21사이에서 $begin(N_{new}) < end(N_{new})$ 조건을 만족하는 두 값을 모두 얻어야만 한다.

4. EP2 레이블링 스킴을 이용한 조인 알고리즘

EP2 레이블링 스킴을 이용하면서 기존의 효율적인 구조 조인 알고리즘으로 구조 질의를 처리할 수 있다. 조상-자손 관계를 판별하는 방법이 범위 기반 레이블링 스킴과 매우 유사하기 때문하다. 임의의 N_x와 이 노드의 자손 N_y에 대해, 범위 기반 스킴에서 $begin(N_x) < begin(N_y)$와 $end(N_y) < end(N_x)$가 성립하듯이, EP2 스킴에서는 $pre(N_x) < pre(N_y)$와 $post(N_y) < post(N_x)$가 성립한다. 따라서, 기존의 구조 조인 알고리즘을 조금만 변형하면 EP2 스킴을 사용하면서 효율적인 구조 질의가 가능하게 된다. 그림 5는 [7]의 Algorithm 2인 getNext(q)를 변형하여 EP2 스킴을 이용하면서 holistic twig 조인을 이용해 twig 질의를 수행할 수 있게 한 것이다.

[2] 임의의 N_x의 레이블 $pre(N_x)$와 $post(N_x)$에 각각 $\log_2 L_{max}$개의 비트 수를 할당했을 때 최대 값이 L_{max}가 된다.

```
1: if isLeaf(q) then
2:    return q;
3: for qi in children(q) do
4:    ni = getNext(qi);
5:    if ni ≠ qi then
6:       return ni;
7: end for
8: nmin = minargni { Cni→pre };
9: nmax = maxargni { Cni→pre };
10: while Cq→pre < Cnmax→pre and Cq→post < Cnmax→post do
11:    Cq→advance();
12: end while
13: if Cq→pre < Cnmin→pre then
14:    return q;
15: else
16:    return nmin;
```

그림 5. 구조 질의를 위한 getNext(q) 알고리즘의 변형

5. 결론

본 논문에서는 Dietz 레이블링 스킴을 개선한 EP2 레이블링 스킴을 제안하였다. EP2 스킴은 범위 기반 레이블링 스킴보다 저장 구조면에서 더 우수하기 때문에 동적인 갱신에 보다 유연하게 대처할 수 있으며, 기존의 구조 조인 알고리즘을 사용하여 효율적으로 구조 질의를 수행할 수 있다. 따라서, 범위 기반 스킴을 사용하고 있는 XML 문서에서 성능 향상을 위한 대체 스킴으로 사용하기에 적합하다.

향후 과제로, DTD나 XML 스키마와 같은 정보가 있는 상황에서 노드 삽입시의 효율적인 레이블 할당에 대해 연구할 계획이다.

참고문헌

[1] W3C, Extensible Markup Language,(XML) 1.0 (Third Edition) http://www.w3.org/TR/REC-xml

[2] W3C, XML Path Language (XPath), http://www.w3.org/TR/xpath

[3] W3C, XQuery 1.0: An XML Query Language, http://www.w3.org/TR/xquery/

[4] P. F. Dietz, Maintaining order in a linked list. In STOC, 1982

[5] S. Al-Khalifa, H.V. Jagadish, N. Koudas, J. M. Patel, D. Srivastava and Y. Wu, Structural Joins: A Primitive for Efficient XML Query Pattern Matching, In ICDE, 2002

[6] N. Bruno, N. Koudas, and D. Srivastava, Holistic Twig Joins: Optimal XML Pattern Matching, In SIGMOD, 2002

[7] H. Jiang, W. Wang and H. Lu, Holistic Twig Joins on Indexed XML Documents, In VLDB, 2003

[8] Q. Li and B. Moon, Indexing and Querying XML data for Regular Path Expressions, In VLDB, 2001

[9] C. Zhang, J. Naughton, D. DeWitt, Q. Luo, On Supporting Containment Queries in Relational Database Management Systems, In SIGMOD, 2001

[10] E. Cohen, H. Kapalan and T. Milo, Labeling Dynamic XML Trees, In PODS, 2002

[11] I. Tatarinov, S. D. Viglas, K. Beyer, J. Shanmugasundaram, E. Shekita and C. Zhang, Storing and Querying Ordered XML Using a Relational Database System, In SIGMOD, 2002

[12] X. Wu, M. L. Lee, W. Hsu, A Prime Number Labeling Scheme for Dynamic Ordered XML Trees, In ICDE, 2003