제26회 한국정보처리학회 추계학술발표대회 논문집 제13권 제2호 (2006. 11)

효율적인 XML 상향식 질의 처리

서동민*, 유재수*
*충북대학교 정보통신공학과
e-mail:{dmseo, yjs}@chungbuk.ac.kr

The Efficient XML Bottom-Up Query Processing

Dong-Min Seo*, Jae-Soo Yoo*
*Dept of Computer and Communication Engineering
Chungbuk National University

요 약

경로 질의는 XML에서 가장 일반적으로 사용되는 질의이며, 기존에 경로 질의를 효율적으로 처리하기 위한 다양한 색인 기법들이 연구되었다. 최근에는 suffix 트리를 사용하는 구조 조인 기법들이 제안되어 경로 질의 성능을 향상시키고 있다. ViST는 가장 대표적인 구조 조인 기법으로 XML 문서에 대한 검색 시간을 줄이기 위해 suffix 트리와 B^+ 트리를 사용한다. 하지만, ViST는 suffix 트리에 최적화되지 못한 번호 부여 기법을 사용함으로써, 질의 처리 시 불필요한 조인과 디스크 접근이 야기된다. 이와 같은 문제들은 경로 질의 성능을 급격히 감소시킨다. 따라서 본 논문에서는 ViST의 문제들을 해결하기 위한 새로운 색인 구조를 제안한다. 제안하는 색인 구조는 suffix 트리를 사용하는 구조 조인 기법의 성능을 향상시키기 위해서 최적화된 번호 부여 기법과 상향식 질의 처리 기법을 사용한다. 본 논문의 성능 평가에서는 제안하는 색인 구조를 ViST와 비교하여, 제안하는 색인 구조가 와일드-카드('*'와 '//')를 포함하는 다양한 단일 경로 질의와 분기 질의에 대해 향상된 성능을 나타냄을 보인다.

1. 서론

XML은 문서 구성 요소들 사이에 계층적인 구조를 가지고 있으나 그 형태가 일정하게 고정된 스키마를 따를 필요가 없는 반구조적인(semi-structured) 특징을 지닌다[1]. 이러한 특성을 지니는 XML 데이터의 처리를 위해 일반적으로 트리 구조의 모델을 사용한다. 그리고 XML 질의 처리를 위해 XPath, Quilt, XML-QL, XQuery 등과 같은 질의 언어들이 연구되었고 이러한 질의 언어들은 그래프로 표현할 수 있다. XML 문서에 대한 경로 질의를 효율적으로 처리하기 위한 색인 기법에 대한 연구도 많이 수행되었는데, 초기에는 주로 경로 색인 기법에 대한 연구가 이루어졌다. 이 기법은 XML 데이터 구조에서 발생 가능한 모든 경로에 대한 색인 그래프를 별도로 구축하고 경로 질의 처리를 위해 원본 데이터 그래프의 탐색이 아닌 경로 색인 그래프를 탐색함으로써 탐색 공간의 크기를 줄여 질의 처리 비용을 줄인다[2,3].

이 논문은 2006년도 교육인적자원부 지방연구중심대학 육성사업과 한국과학재단 특정기초 연구(과제번호R01-2006-000-10809-0)의 지원에 의하여 연구되었음

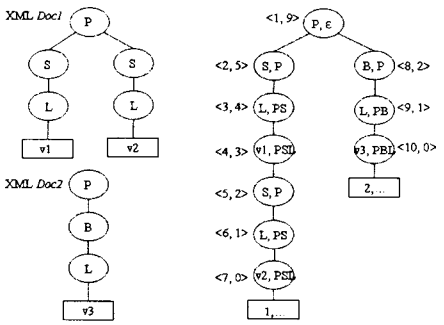
최근에는 조상-후손 관계 등의 포함 질의를 효율적으로 처리할 수 있는 구조 조인 기법에 대한 연구가 많이 이루어졌다[4]. 하지만 구조 조인 기법은 단순 경로 질의를 효율적으로 처리 하지만 분기 구조 질의는 항상 두 개의 부 질의(sub query)로 분해한 뒤에, 각각의 경로 질의를 통해 처리하고 최종 결과는 많은 비용이 요구되는 부 질의 결과들에 대한 조인 연산을 통해 얻는다. 구조 조인 기법의 문제를 해결하기 위해 최근에는 suffix 트리를 이용해 엘리먼트 간의 포함 관계를 살펴 질의를 처리하고, 문서 전체에 대한 탐색을 줄이기 위해 B^+ 트리를 이용하여 해당하는 엘리먼트들만을 비교하여 질의를 처리하는 기법들이 제안되었다[5]. 하지만 suffix 트리에서 사용하는 노드 번호 부여 기법의 특성으로 인해 질의 처리 시 실제 문서에서 자식이 아닌 엘리먼트도 자식으로 보고 불필요한 노드 접근이 발생해 질의 처리 속도가 감소하는 문제가 발생한다.

이러한 문제를 해결하기 위해 본 논문에서는 효율적인 번호 부여 기법을 사용하는 색인 구조를 제안하고, 이 색인구조에 알맞은 상향식 질의 처리 기법을 제안한다. 그리고 다양한 환경에서 성능 평가를 수행하여 제안하는 색인구조가 ViST보다 우수함을 보인다.

2. 관련연구

2.1 Suffix 트리를 사용한 질의 처리 기법

구조 조인 기법에서 발생하는 조인 비용을 줄이기 위해 최근에는 suffix 트리를 이용해 엘리먼트들의 포함 관계를 살펴 질의를 처리하는 기법이 제안되었다[5]. 이 기법은 XML 문서 내의 구조적인 관계를 suffix 트리로 표현하고, 색인을 이용 경로 매칭(sequence matching)을 통해 처리하는 기법이다. XML 질의를 경로 매칭을 통해 모델링하는 목적은 분기 질의를 분해하지 않고 처리함으로써 조인 연산 비용을 줄이기 위한 것이다. 그림 1은 두 XML 문서에 대한 suffix 트리를 나타낸 것이다. 질의 처리는 트리의 모든 노드들을 순회하면서 질의를 구성하는 노드들이 suffix 트리에 포함되는지를 살펴 처리한다. 하지만 이렇게 질의를 처리하기 위해서는 많은 양의 부 트리를 순회해야 하기 때문에 비용이 많이 드는 단점이 있다. 또한 suffix 트리는 메인 메모리 기반의 색인구조이고 상업용 DBMS에 적합하지 못한 문제가 있다.

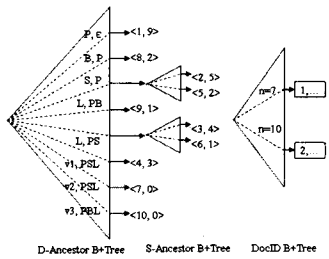


(a) XML 문서 트리 (b) (a)에 대한 suffix 트리

(그림 1) ViST에서 사용하는 suffix 트리

2.2 ViST(Virtual Suffix Tree)

ViST는 suffix 트리의 문제점을 개선한 가장 대표적인 기법이다. ViST는 기존에 제안되어진 질의 처리 기법의 문제점인 조인 비용을 줄이기 위해서 문서를 suffix 트리로 변환한다. 그리고 질의 처리 시 요구되는 suffix 트리의 순회 비용을 줄이기 위해 각 노드마다 번호를 부여하고, 번호가 할당된 suffix 트리의 노드들을 B^+ 트리에 구축함으로써 최소 노드 접근 기법을 제공한다[5]. 그림 2는 그림 1에 대한 ViST를 보여준다.



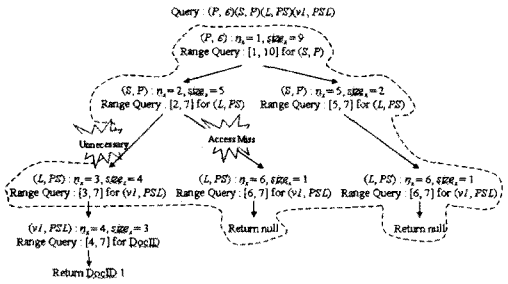
(그림 2) ViST의 색인 구조

3. 제안하는 색인구조

3.1 ViST의 문제점

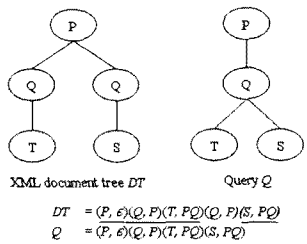
ViST에서의 질의 처리 기법은 그림 3과 같다. "*P/S/L/v1*" 질의가 들어오면 먼저 질의를 분석 후 structure-encoded 경로를 만든다. 그리고 처음 나오는 (*symbol*, *prefix*) 쌍인 (*P*, *ε*)을 그림 2의 D-Ancesotr B^+ 트리를 통해 찾고 해당 n_x, $size_x$의 값을 S-Ancesotr B^+ 트리를 통해 획득한다. 이 값을 이용하여 다음 순서쌍인 (*S*, *P*)에 대해 (n_x, n_x+size_x)로 각각 범위 질의를 한다. 그리고 범위 질의를 통해 획득한 결과들을 다음 순서쌍인 (*L*, *PS*)에 대해서도 범위 질의를 수행하고, 이 결과를 가지고 마지막 순서쌍인 (*v1*, *PSL*)에 대해 범위 질의를 수행한다. 마지막으로 요청된 질의에 부합되는 문서를 찾기 위해 질의 대상이 되는 경로의 마지막 노드의 n_x 값에 할당된 문서를 DocID B^+ 트리에서 찾는다.

하지만 ViST에서 두 노드의 S-Ancestorship 관계를 표현하기 위해 사용하는 번호 부여 기법으로 인해 실제 문서 트리에서 자식이 아닌 노드에 대한 노드도 자식으로 보는 문제가 발생한다. 예를 들어, 그림 3에서 n_x가 2인 *SP*의 자식은 n_x가 3인 *LPS*만이 존재하지만 n_x가 6인 *LPS*도 자식으로 보고 비교 연산을 수행하게 된다. 그리고 suffix 트리의 suffix가 루트 엘리먼트부터 현재 엘리먼트까지의 경로 정보를 나타낸다는 특성을 고려하지 않고 질의를 처리하기 때문에 그림 3처럼 불필요한 노드를 접근하는 문제점이 발생한다. 예를 들어, 그림 3에서 (*v1*, *PSL*)은 자신의 suffix인 *PSL*을 통해 자신의 상위 노드로 (*P*, *ε*), (*S*, *P*) 그리고 (*L*, *PS*)를 포함하고 있다는 알 수 있다. 그러므로 (*P*, *ε*), (*S*, *P*) 그리고 (*L*, *PS*)에 대한 연산은 불필요하다.



(그림 3) ViST에서 질의 처리 문제

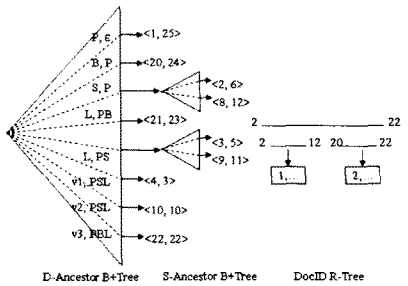
또한, 분기 질의 처리 시 문서와 질의의 구조가 틀려도 질의의 structure-encoded 경로가 문서에 포함되어 결과로 나타나는 False Alarm이 발생할 수 있다. 예를 들면, 그림 4의 질의를 보면 문서와는 다른 구조이다. 하지만 문서의 structure-encoded가 질의 Q의 structure-encoded를 포함하기 때문에 질의에 부합된 문서로 보고 결과로 나타나는 문제가 발생한다.



(그림 4) ViST에서 False Alarm 문제

3.2 제안하는 상향식 질의 처리 기법

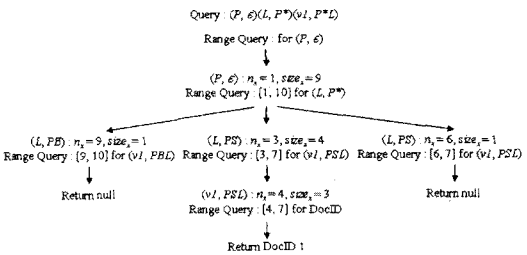
제안하는 색인 구조는 ViST와 유사하다. D-Ancestor B^+ 트리는 ViST와 같은 방법으로 (*symbol, prefix*) 쌍을 키 값으로 하여 삽입한다. S-Ancestor B+ 트리는 ViST와 같은 방법으로 삽입하지만 키 값을 Durable 번호 부여 기법에서 사용하고 있는 (*order, size*)로 변경하여 삽입한다.



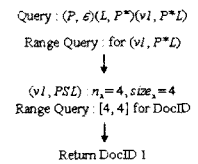
(그림 5) 제안하는 색인 구조

이는 Durable 번호 부여 기법[6]을 통해 ViST의 문제인 자식이 아닌 노드 접근 문제를 해결하기 위함이다. 하지만 번호 부여 기법이 달라져 질의 대상이 되는 경로의 마지막 노드에 대한 n_x를 가지고 DocID B^+에서 질의를 포함하는 문서를 찾을 수 없기에, 영역 질의를 제공하는 DocID R 트리를 사용한다. 그림 5는 그림 1에 대한 제안하는 색인 구조를 보여준다.

각 노드의 suffix는 현재 노드까지의 경로를 나타내므로, 이 정보를 사용하면 ViST에서 질의 처리 시 요구되는 중간 노드 검색 비용을 줄일 수 있다. 따라서 질의를 상향식으로 처리하는 기법을 사용하게 되면 단일 경로 질의의 경우는 마지막 노드에 대한 비교 연산만으로 질의를 처리할 수 있다. 예를 들어, 그림 6과 같이 와일드-카드가 포함된 질의를 하향식으로 처리하면 (*P, ε*)에 대한 검색 후, 그림 5의 D-Ancestor B^+ 트리에서 (*L, P**)에 대한 범위 검색을 통해 (*L, PB*)와 (*L, PS*)에 대한 결과를 얻는다. 그리고 그 각각에 대하여 나머지 질의를 처리하게 되므로 많은 비용이 요구된다.



(a) 하향식 질의 처리 방식



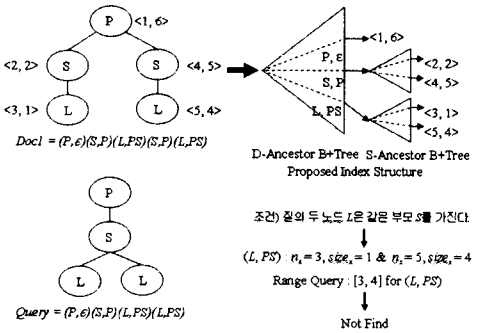
(b) 상향식 질의 처리 방식

(그림 6) ViST에서 False Alarm 문제

하지만 같은 질의를 상향식으로 처리하면 먼저 (*v1, P*L*)를 D-Ancestor B^+ 트리에서 범위 질의하여 (*v1, PSL*)에 대한 결과를 획득하고 이에 대한 질의 수행을 통해 최종 질의 결과를 얻을 수 있다.

또한 상향식으로 질의를 처리하므로 그림 7에서와 같이 질의의 마지막부터 필요한 노드들만을 접근하여 처리하므로 구조적으로 틀린 문서를 판별해 나갈 수 있다. 즉, False Alarm이 해결된다. 예를 들면, 그림 7의 질의 처리 시 질의의 structure-

encoded에서 (*L*, *PS*)이 연속되어 나타나므로 하나의 (*S*, *P*)를 부모 노드로 가지는 것을 알 수 있다. 따라서 문서의 structure-encoded에서 하나의 (*S*, *P*)에 대해 두 (*L*, *PS*)이 포함되는지를 비교하면 된다. 그림 7의 문서는 질의의 structure-encoded를 포함하지만, 상향식 질의 처리 조건을 만족하지 못하므로 결과로 나타나지 않는다.



(그림 7) 제안하는 색인 구조에서의 분기 질의 처리

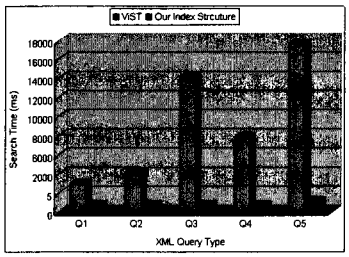
4. 구현 및 성능 평가

성능 평가에 사용된 시스템은 3GHz Pentium Ⅳ 프로세서에 512Mbyte의 메모리를 가지며, 운영체제는 Windows XP를 사용하였다. 사용한 언어는 C++를 사용하였으며, 실험에 사용된 데이터 셋은 NIAGARA에서 사용된 데이터 셋을 사용하였다. 표 1은 실험에서 사용한 다양한 종류의 질의 유형을 보여준다.

<표 1> 실험에 사용된 질의 유형

질의	질의 표현식
Q_1	/W4F_DOC/Actor/Name/FirstName/Robert
Q_2	/W4F_DOC/Actor/Name/*/Robert
Q_3	//Name/*/Robert
Q_4	//Name[FirstName/Claudio]/LastName/Alfonsi
Q_5	//*/Name[FirstName/Claudio]/LastName/Alfonsi

그림 8은 제안하는 색인 구조가 ViST보다 질의 처리시간이 확연히 감소함을 통해 성능이 우수함을 보여주고 있다. 깊이가 낮은 질의의 경우는 비슷한 성능을 보이고 있지만, 상향식 질의 처리의 장점으로 인해 깊이가 높아질수록 더 좋은 성능을 나타내는 것을 알 수 있다. 그리고 제안하는 번호 부여 기법이 불필요한 노드의 접근을 피하고, 동시에 상향식으로 질의를 처리함으로 중간 처리 과정에 요구되는 노드 접근 비용을 감소시켜 와일드-카드가 포함된 질의 및 분기 질의 처리에 있어서도 우수한 성능을 보여주고 있다. 또한 ViST에서는 False Alarm 문제로 인해 질의에 대한 잘못된 결과가 나타나는데 반해 제안하는 색인 구조에서는 전혀 False Alarm 문제가 발생하지 않는 것을 확인할 수 있었다.



(그림 8) ViST와 제안하는 색인 구조의 질의 처리 시간

5. 결론

본 논문에서는 경로 질의를 효율적으로 처리하기 위한 새로운 색인 구조를 제안하고 이에 적합한 질의 처리 기법을 설계하고 구현하였다. 이 기법은 suffix 트리에 적합한 Durable 번호 부여 기법을 사용하고, 질의를 상향식으로 처리함으로써 ViST에서 발생하는 많은 중간 검색 과정과 불필요한 노드 접근을 감소시킴으로써 질의 처리 시간을 줄일 수 있었다. 또한 상향식 질의를 통해 분기 질의 처리 시 발생할 수 있는 False Alarm 문제 또한 자연스럽게 해결할 수 있음을 증명하였다. 향후 연구는 다양한 질의를 더욱 효율적으로 처리할 수 있는 기법들에 대한 연구들을 진행할 예정이다.

참고문헌

[1] T. Bray, J. Paoli, C. M. et. al., "Extensible Markup Language (XML) 1.0 3rd", http://www.w3.org/TR/REC-xml (2004)

[2] M. Fernandez and D. Suciu, "Optimizing regular path expressions using graph schema", In ICDE (1998) 14-23

[3] C. W. Chung, J. K. Min, K. S. Shim, "APEX: An Adaptive Path Index for XML Data", In SIGOD (2002) 121-132

[4] H. Jiang, H. Lu, W. Wang, and B. C. Ooi, "XR-Tree: Indexing XML Data for Efficient Structural Joins", In IEEE International Conference on Data Engineering (2003) 253-264

[5] H. Wang, S. Park, W. Fan, and P. S. Yu, "ViST: A Dynamic Index Method for Querying XML Data by Tree Structures", In SIGMOD (2003) 110-121

[6] Q. Li and B. Moon, "Indexing and querying XML data for regular path expressions", In VLDB (2001) 361-370