XML 문서의 효율적인 검색과 관리를 위한 SCOF 모델

Service-centric Object Fragmentation Model for Efficient Retrieval and Management of XML Documents

정창후
한국과학기술정보연구원

Chang-Hoo Jeong
Korea Institute of Science and Technology Information

요약

XML 문서가 기하급수적으로 증가하면서 XML 문서를 처리하는 방법론에 대한 많은 논의가 있어왔다. 본 논문에서는 두 가지 중요한 목적을 가지고 XML 정보 검색 및 관리 시스템을 개발하는데, 첫 번째는 질의에 적합한 내용을 쉽고 빠르게 검색해서 제공하는 것이고, 두 번째는 시스템의 부담을 최소화하면서 효율적이고 안정적인 관리 기능을 제공하는 것이다. 이렇게 실용적인 시스템을 개발하는 핵심 기술은 XML 문서를 어떻게 효과적으로 분할하여 구조적으로 서비스하는가에 달려 있다. 이러한 목적을 달성하기 위하여 본 논문에서는 SCOF (Service-centric Object Fragmentation) 모델을 제안한다. SCOF 모델은 XML 데이터베이스 관리자에 의해서 정의되는 변환 규칙(conversion rule)을 이용하여 문서를 분할하는 준분할(semi-decomposition) 저장 방식이다. SCOF 모델을 사용한 키워드 기반 검색은 전형적인 XML 질의 언어처럼 문서의 특정 엘리먼트나 속성값을 이용하여 검색을 수행할 수 있다. 비록 이러한 접근법이 XML 문서 컬렉션에 대한 관리자의 지식을 필요로 한다고 하더라도, 개별 문서의 크기나 전체 문서의 양에 상관없이 검색과 관리를 효율적으로 수행할 수 있기 때문에 실용적인 시스템을 구축할 수 있다는 장점이 있다.

Abstract

Vast amount of XML documents raise interests in how they will be used and how far their usage can be expanded. This paper has two central goals: 1) easy and fast retrieval of XML documents or relevant elements; and 2) efficient and stable management of large-size XML documents. The keys to develop such a practical system are how to segment a large XML document to smaller fragments and how to store them. In order to achieve these goals, we designed SCOF(Service-centric Object Fragmentation) model, which is a semi-decomposition method based on conversion rules provided by XML database managers. Keyword-based search using SCOF model then retrieves the specific elements or attributes of XML documents, just as typical XML query language does. Even though this approach needs the wisdom of managers in XML document collection, SCOF model makes it efficient both retrieval and management of massive XML documents.

I. 서 론

XML 정보 검색은 일반적으로 XML 문서의 계층적인 구조에 기반을 두고 있다. 명확하게 정의된 계층 구조가 문서에 존재하기 때문에 문서의 내용뿐만 아니라 구조에 대해서도 검색을 수행할 수 있다. 이렇게 세분화된 검색을 명시하기 위해서 XPath나 XQuery와 같은 XML 질의 표현식을 사용하는데, 이러한 질의는 사용자가 XML 문서상의 계층 구조(XML DTD 또는 XML 스키마)를 어느 정도 알고 있어야만 사용이 가능하다. 이와 같이 표현력이 풍부한 구조적 질의가 XML 문서의 검색 성능 향상에 기여할 수 있다는 부분은 잘 알려져 왔으나, 구조 검색을 표현하는 사용자의 미숙성 혹은 관련 있는 정보를 어떤 구조로 찾아야하는지를 모르는 데이터에 대한 비전문성으로 인해 실제적으로 그 효과를 보는 경우는 드물다. 더욱이 최근에는 질의에 포함되어 있는 구조적 힌트가 검색의 정확도를 향상시키지 못한다는 연구 결과도 발표되고 있다[1]. 그럼에도 불구하고 대부분의 사용자는 검색 결과를 브라우징할 때 XML 문서 전체보다는 그 문서를 구성하는 일부 중요한 영역만을 보기를 바라고 있다[2, 3]. 사용자들은 전체 문서보다 특정 엘리먼트에서 많은 유용한 정보를 쉽게 찾을 수 있기 때문에 엘리먼트가 그들의 작업을 위해서 좀 더 유용하다고 생각한다[2, 4, 5]. 따라서 XML 정보 검색 시스템은 XML 문서로부터 질의에 적합한 엘리먼트를 검색하고 브라우징해줄 필요가 있다. 이와 같은 사용자의 복잡한 요구사항으로 인해 XML 정보 검색 시스템을 개발하는 작업은 많은 어려움에 직면하게 된다.

의미 있는 정보를 포함하면서 계층적으로 잘 구조화된 엘리먼트는 문서 컬렉션의 중요한 기능이지만 복잡한 문서 구조를 세세하게 파악하여 자신이 원하는 정보를 찾아내야 하는 것은 사용자 질의의 역할이 아닐 수 있다[1]. 따라서 관리자가 XML 문서의 의미 있는 내용이 쉽게 검색될 수 있도록 사용자 친화적인 시스템을 구축하는 것이 복잡한 질의 언어에 대한 적절한 대안일 수 있다. 그래서 키워드를 이용하여 XML 문서를 구조적으로 검색하는 다양한 연구가 이루어지고 있다[2, 6, 7, 8, 9].

본 논문에서는 XML 문서 컬렉션 관리자가 전처리 과정으로 서비스에 적합한 구조를 미리 정의해 놓고, XML 문서의 계층 관계와 해당 내용을 검색 필드로 지정해서 키워드 검색을 가능하도록 하는 SCOF 모델을 제안한다. 대부분의 정보 서비스 시스템이 주어진 XML 문서 컬렉션 내에서 구조가 고정되어 있기 때문에 이러한 방법을 쉽게 적용할 수 있다. 그 결과 최종 사용자에게 복잡한 XML 질의 언어 없이 키워드 방식의 검색 서비스를 제공할 수 있을 뿐만 아니라, XML 데이터베이스 관리자에 의해서 의미 있게 재구성된 문서의 구조적 검색 결과도 함께 제공할 수 있다.

II. SCOF 모델

1. SCOF 모델 정의

XML 문서를 효과적으로 검색하기 위한 보편화된 모델이나 데이터 구조에 대한 의견일치가 아직까지는 명확하게 이루어지고 있지 않다[10]. 또한 고전적인 정보 검색과 달리, XML 검색은 명확하게 문서나 색인 단위를 결정하기가 쉽지 않다. 구조화된 문서 검색을 제공하기 위해서 시스템은 질의에 부합하는 가장 적합한 문서의 부분을 검색해야 하는데, 이러한 종류의 시스템을 알고리즘적으로 구현하기에는 많은 어려움이 있다. 그래서 본 논문에서는 XML 문서를 단편 노드로 분할하여 처리하는 SCOF 모델을 제안한다.

SCOF(Service-centric Object Fragmentation) 모델은 XML 문서의 부모-자식 관계, 형제 관계 등의 계층 정보를 유지하면서 의미 있는 엘리먼트를 중심으로 여러 개의 중복된 영역이 없는 단편 노드(document fragment)로 분할하는 준분할(semi-decomposition) 저장 방식이다. 이때 XML 문서의 단편 노드는 계층적으로 구조화된 트리 형태를 가지는데 이것을 FOT(Fragment Object Tree)라고 부른다. 분할 저장 방식이 모든 엘리먼트 노드의 구조 정보를 저장하는 반면에 준분할 저장 방식은 단편 노드 간의 구조 정보만을 저장한다. 그리고 단편 노드 내에 존재하는 엘리먼트 노드들은 가상 분할 저장 방식처럼 있는 그대로 저장한다. 하지만 위치 정보를 따로 추출하지는 않고 검색에 사용될 엘리먼트나 속성에 대해서 검색 필드를 미리 지정하여 향후에 키워드 검색에 이용될 수 있도록 한다.

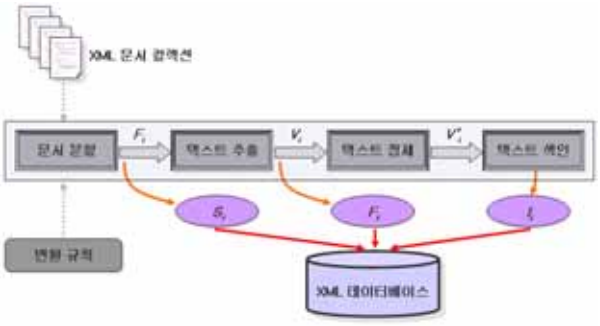
SCOF 모델에서 XML 문서는 변환 규칙(conversion rule)을 통하여 단편 노드로 분할된다. 문서의 내용과 형식을 잘 알고 있는 관리자가 변환 규칙을 이용하여 XML 문서의 효율적인 서비스 구조를 기술함으로써, 최종 사용자는 해당 문서에 가장 적합한 구조로 XML 문서의 검색 및 관리 서비스를 이용할 수 있다.

SCOF 모델을 도식화하면 다음과 같이 표현된다.

$$f(D, R) = \{(F_i, S_i, I_i) \mid 1 <= i <= n\}$$

f : XML 문서를 단편 노드로 분할하는 함수
D : XML 문서
R : 관리자에 의해서 정의된 변환 규칙
n : $f(D, R)$에 의해서 분할된 단편 노드의 개수
F_i : i번째 단편 노드
S_i : F_i의 구조 정보(주변 단편 노드와의 관계 설정을 위한 정보)
I_i : F_i의 검색을 위한 색인 정보

SCOF 모델을 이용하여 XML 문서를 처리하는 과정을 살펴보면 그림 1과 같다.

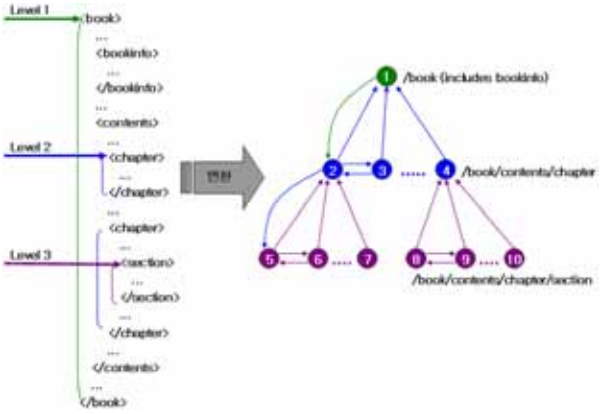


▶▶ 그림 1. XML 문서를 데이터베이스에 저장하는 과정

그림 1에서 보이는 것과 같이 XML 문서를 서비스하기 위해서는 XML 문서와 해당 문서를 분할하기 위한 변환 규칙이 필요하다. 서비스할 XML 문서에 대한 변환 규칙이 정의가 되면 XML 문서를 분할하여 단편 노드(F_i)와 구조 정보(S_i)를 생성한다. 다음 작업으로 단편 노드로부터 검색에 사용할 엘리먼트나 속성 값(V_i)을 추출한다. 추출된 값은 정규 표현식과 같은 문자열 처리 기술을 이용하여 사용자가 원하는 포맷(V'_i)으로 정제가 된다. 정제된 데이터는 관리자가 지정한 색인 방식에 따라서 데이터 특성에 맞는 색인 정보(I_i)를 생성하게 된다. 그리고 최종적으로 단편 노드(F_i)와 구조 정보(S_i), 색인 정보(I_i)는 데이터베이스에 함께 저장되어 검색 및 관리 서비스에 사용된다.

일반적으로 사이즈가 큰 XML 문서들을 다루는 데이터베이스는 검색과 관리 양 측면에서 많은 어려움을 겪게 된다. 그러나 SCOF 모델에서는 XML 문서가 데이터베이스 관리자에 의해서 작성된 변환 규칙을 이용하여 크기가 작은 단편 노드들로 분할되기 때문에, XML 문서의 크기에 상관없이 효과적으로 검색 및 관리 서비스를 제공할 수 있다.

SCOF 모델에서 XML 문서는 단편 노드의 구조 정보를 이용하여 그 자체보다 처리하기 쉬운 FOT로 재구성된다. FOT는 단편 노드의 순회나 관리와 같은 다양한 기능들을 제공하는데, 이러한 기능들은 단편 노드를 조작하는 것을 쉽게 만든다. 이러한 측면에서 FOT는 비록 모든 기능들이 완전히 일치하지는 않더라도 구조적으로 DOM 트리와 유사하다.



▶▶ 그림 2. XML 문서와 FOT

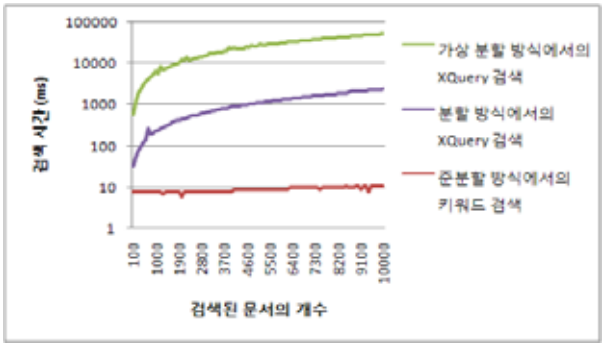
그림 2에서 XML 문서의 book, chapter, 그리고 section이 단편 노드로 지정되었다. 이것들은 서로간의 데이터의 중복을 허용하지 않는다. 그림 2를 통해 알 수 있듯이 SCOF 모델은 원래 XML 문서의 계층 관계를 있는 그대로 유지하면서 단편 노드로 분할하기 때문에 새롭게 생성된 트리도 원래 문서의 구조적인 형태를 그대로 가지고 있다. FOT는 의미 있는 엘리먼트를 중심으로 구성된 레벨이 낮고 간단한 구조를 가지는데, 트리의 노드(node)들은 단편 노드(F_i)를 나타내고 트리의 간선(edge)들은 구조 정보(S_i)를 나타낸다.

2. SCOF 모델을 이용한 문서 관리의 특징

SCOF 모델을 이용한 문서 관리의 특징은 단편 노드의 삽입, 수정, 삭제뿐만 아니라 단편 노드의 이동, 병합, 재생성이 가능하다는 것이다. 단편 노드의 삽입은 FOT의 특정 위치에 단편 노드 혹은 노드 그룹을 추가하는 기능을 말한다. 단편 노드의 삭제는 FOT의 특정 위치에 있는 단편 노드 혹은 노드 그룹을 삭제하는 기능을 말한다. 단편 노드의 수정은 FOT에 있는 특정 단편 노드의 내용을 수정하는 기능을 말한다. 단편 노드의 이동은 FOT의 특정 위치에 있는 단편 노드 혹은 노드 그룹을 FOT 내의 다른 위치로 이동하는 기능을 말한다. 이때 원본 문서의 전체 구조를 위배하지 않는 경우에만 이동이 허용된다. 단편 노드의 병합은 하나의 FOT에 또 다른 FOT를 병합하는 기능을 말한다. 이것은 다른 말로 하나의 XML 문서에 또 다른 XML 문서를 결합시켜서 서비스할 수 있다는 것을 의미한다. 병합 시에는 병합 키를 사용하는데, 병합 키에 따라서 FOT의 다양한 레벨에서 통합 작업이 이루어질 수 있다. 노드의 재생성은 FOT를 XML 문서로 재생성하는 기능을 말한다.

III. 실험 및 고찰

본 논문에서는 SCOF 모델을 사용한 XML 정보 검색 및 관리 시스템의 성능을 평가하기 위해서 문서 검색 시간과 문서 입력 시간을 측정해보았다.

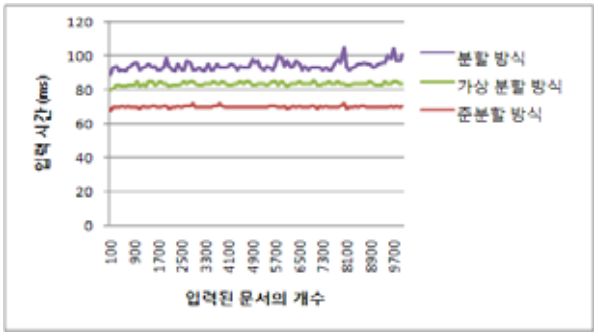


▶▶ 그림 3. 검색 시간 비교

그림 3은 XQuery 기반 검색과 SCOF 모델을 사용한 키워드 기반 검색의 검색 시간을 보여준다. 순수한 검색 시간만을 나타내기 위해서 검색 결과를 가져오는 시간은 제외하였다. 준분할 저장 방식을 사용한 키워드 검색이 분할 저장 방식 혹은 가상 분할 저장 방식을 사용한 XQuery 검색보다 성능이 훨씬 우수함을 알 수 있다. 더욱이 SCOF 기반의 키워드 검색의 성능은 검색 결과 문서의 수가 증가하더라도 크게 영향을 받지 않는 반면에, XQuery를 이용한 검색의 성능은 검색 결과 문서의 수가 증가함에 따라서 급격하게 저하되는 것을 볼 수 있다. 이러한 측면에서 SCOF 접근법이 대용량의 XML 문서를 처리하는데 있어서 훨씬 효율적이라는 것을 알 수 있다.

그림 4는 각 방법의 문서 입력 시간을 보여준다. 그림 4에서 보이는 바와 같이, 준분할 저장 방식이 분할 저장 방식이나 비분할 저장 방식보다 문서 입력 시간이 적게 걸리는 것을 볼 수 있다. 준분할 저장 방식의 입력 시간이 다른 방법들보다 빠른 이유는 XML 문서를 저장하는 방법과 밀접하게 관련되어 있다. 분할 저장 방식이나 비분할 저장 방식이 XML 문서의 모

든 엘리먼트를 일일이 고려해서 저장해야 하는 반면에, 준분할 저장 방식은 의미 있게 분할된 단편 노드만을 고려해서 저장하면 되기 때문이다.



▶▶ 그림 4. 입력 시간 비교

비록 XQuery가 XML 문서를 다루는 강력한 기능을 가지고 있다고 하더라도, 대용량 XML 데이터를 다루는 데 있어서 XQuery의 성능 테스팅은 별로 이루어져 있지 않은 상태이다. 그것의 복잡성 때문에, 사용자들이 기대하고 만족할 만한 성능을 가진 XQuery 기반의 시스템을 구현하는 것은 아직까지는 어려운 상황이라고 볼 수 있다. 이것에 대한 실용적 대안으로 SCOF 접근법은 검색 및 관리 측면에 있어서 대용량의 XML 문서를 처리하는데 훨씬 적합한 방법론임을 알 수 있다.

IV. 결론 및 향후 연구

지금까지 우리는 SCOF 모델에 대해서 살펴보았다. 이러한 접근법의 장점은 서비스 공급자인 관리자와 서비스 수요자인 사용자 입장에서 살펴볼 수 있다. 먼저 관리자 입장에서는 복잡한 계층 구조의 XML 문서를 서비스의 목적에 따라서 중요하다고 판단되는 단편노드의 단위로 구분하고 단편노드 안에 존재하는 세부 엘리먼트 및 속성에 대해서 별칭(alias)을 사용하는 검색 필드를 미리 구성해 놓음으로써 보다 효과적으로 검색 및 관리 서비스를 제공할 수 있다. 사용자 입장에서는 XML 문서의 구조적 특성을 명확하게 알지 못하거나 질의어 작성에 익숙하지 않더라도, 키워드 방식의 인터페이스를 사용해 쉽게 검색 서비스를 이용할 수 있다.

그리고 시스템 측면에서 얻을 수 있는 장점은 문서의 중요한 엘리먼트를 중심으로 구조 정보를 그대로 유지하면서도 검색 결과 재구성 시에 엘리먼트 재조합을 위한 오버헤드가 거의 발생하지 않는다는 것이다. 또한 문서 관리 작업이 주로 단편 노드 기반으로 이루어지기 때문에 대용량의 문서를 처리함에 있어 오버헤드를 분산시키는 이점을 얻을 수 있다. 이것은 실제적으로 XML 정보 검색 관리 시스템을 실용화 시키는데 있어서 가장 중요한 두 가지 요소라고 생각된다. 실제로 "역사정보통합시스템"[11]이나 "조선왕조실록"[12]과 같이 대용량의 XML 문서를 다루거나 "향토문화전자대전"[13]과 같이 실시간 문서 편찬 작업을 수행하는 시스템에서 SCOF 모델이 효과적으로 활용되고 있다.

향후 연구로는 사용자의 다양하고 복잡한 요구 사항을 충분히 반영할 수 있도록 변환 규칙에 대한 보다 상세하고 다양한 명세 작업이 필요하다. 대부분의 시스템들이 검색 시에 XML 질의 표현식을 사용하는 반면에 본 시스템은 사용자 변환 규칙을 통하여 질의에 관련된 작업을 처리하기 때문에, 변환 규칙에 좀 더 융통성을 부여할 필요가 있다.

▌참 고 문 헌▐

[1] A. Trotman, and M. Lalmas, "Why Structural Hints in Queries do not Help XML-Retrieval", SIGIR'06, Seattle, Washington, USA, pp. 711-712, 2006.
[2] B. Larsen, A. Tombros, and S. Malik, "Is XML Retrieval Meaningful to Users? Searcher Preferences for Full Documents vs. Elements", SIGIR'06, Seattle, Washington, USA, pp. 663-664, 2006.
[3] J. Kamps, M. Koolen, and M. Lalmas, "Where to Start Reading a Textual XML Document?", SIGIR'07, Amsterdam, The Netherlands, pp. 723-724, 2007.
[4] H. S. Kim, and H. J. Son, "Users Interaction with the Hierarchically Structured Presentation in XML Document Retrieval", INEX 2005, Glasgow, Scotland, pp. 422-431, 2005.
[5] S. Betsi, M. Lalmas, A. Tombros, and T. Tsikrika, "User Expectations from XML Element Retrieval", SIGIR'06, Seattle, Washington, USA, pp. 611-612, 2006.
[6] D. Florescu, D. Kossmann, and I. Manolescu, "Integrating Keyword Search into XML Query Processing", WWW 2000, Amsterdam, The Netherlands, 2000.
[7] T. Shimizu, N. Terada, and M. Yoshikawa, "Kikori-KS: An Effective and Efficient Keyword Search System for Digital Libraries in XML", ICADL 2006, Kyoto, Japan, 2006.
[8] L. Guo, F. Shao, C. Botev, and J. Shanmugasundaram, "XRANK: Ranked Keyword Search over XML Documents", SIGMOD 2003, San Diego, CA, pp. 16-27, 2003.
[9] R. Schenkel, and M. Theobald, "Structural Feedback for Keyword-Based XML Retrieval", ECIR 2006, London, England, pp. 326-337, 2006.
[10] C. D. Manning, P. Raghavan, and H. Schutze, "Introduction to Information Retrieval", Online Book, pp. 149-164, 2007.
[11] Korean History On-line, "http://www.koreanhistory.or.kr"
[12] The Annals of The Choson Dynasty, "http://sillok.history.go.kr"
[13] Digital Encyclopedia for Local Areas, "http://www.grandculture.net"