XML 공유 구조 발견을 위한 변형 순차패턴 마이닝 알고리즘

이정원[0] 이기호
이화여자대학교 과학기술대학원 컴퓨터학과
{jungwony[0], khlee}@ewha.ac.kr

Adapted Sequential Pattern Mining Algorithms for Finding XML Common Structure

Jung-Won Lee[0] Kiho Lee
Dept. of Computer Science, Ewha Institute of Science and Technology

요 약

최근 XML 관련 연구가 급증하면서 저장 기법, 질의 최적화, 인덱싱 등의 기법이 활발히 개발되고 있다. 그러나 하나의 DTD나 XML Schema를 공유하는 문서집합이 아닌 다양한 구조를 가진 문서들을 대상으로 하는 경우, 다중 문서간의 구조적 유사성이나 차이 등을 파악할 필요가 있다. 특히 서로 다른 사이트나 문서관리 시스템에서 도출된 문서들을 병합하거나 분류할 필요가 있을 때, 두 문서의 병합 가능성 및 분류 기준을 파악하는 것은 매우 중요하다. 따라서 본 연구에서는 여러 문서들의 구조를 구성하는 경로들간의 유사성을 파악하기 위해 기존의 순차패턴 마이닝 알고리즘을 변형하였다. 변형된 순차패턴 마이닝 알고리즘[1]을 통해 두 문서간의 정확한 공유 경로를 찾을 수 있었다.

1. 서 론

최근 XML 문서는 웹상에서 표현 및 교환의 수단으로 널리 확산되고 있다. XML은 사용자가 임의로 엘리먼트를 정의할 수 있고 엘리먼트는 하위 엘리먼트를 가짐으로써 계층적인 구조를 형성한다[2]. 이러한 XML의 특징은 정보검색, 문서관리 시스템, 그리고 데이터 마이닝에 커다란 영향을 미쳐 왔다. 따라서 XML 관련 저장 기법, 인덱싱 기법, 질의어, 그리고 질의 최적화에 이르기까지 많은 연구가 진행되고 있다.

그러나 이러한 연구들은 비슷한 문서를 대상으로 하나의 DTD를 공유하거나 XML Schema를 공유하는 것이 대부분이다. 따라서 다양한 구조를 가진 문서를 대상으로 하는 경우, 바로 인덱싱이나 저장 기법을 설계할 것이 아니라 문서들의 구조의 공유 정도를 미리 파악하는 일은 매우 중요하다. 더욱이 다양한 사이트나 데이터베이스로부터 도출된 문서들을 하나의 시스템에 적용하기 위해 병합하거나 저장하기 위해서는 먼저 문서들의 구조적 유사성을 파악해 볼 필요가 있다. 또한 문서의 유사한 정도에 따라 문서를 분류하거나 클러스터링 하고자 하는 경우에도 XML의 가장 큰 장점인 내재된 구조를 feature로 하여 문서간의 공유 feature를 추출하고 이를 토대로 분류 알고리즘을 적용할 필요가 있다.

따라서 본 논문에서는 순서 개념이 있는 아이템의 시퀀스를 방대한 데이터베이스에서 선형시간에 찾을 수 있는 순차 패턴 마이닝 알고리즘[1]을 변형하여 문서간의 구조의 공유정도를 파악한다. 본래 순차 패턴 마이닝 알고리즘이 가지는 5단계를 모두 XML의 구조의 공유 정보를 추출해 낼 수 있게 변형하고 이를 통해 최종적으로 두 문서 구조의 공유 경로를 산출해 낸다.

2. 관련 연구

2.1 XML 구조 발견

XML은 다른 비구조적인 문서와는 달리 구조정보를 내재하고 있다. 따라서 문서의 구조를 발견하는 것은 모든 XML 관련 연구에서 필수적이다. XML 문서의 데이터를 관계형 데이터 베이스로 옮기거나 체계적인 구조로 저장하기 위한 구조 발견을 목적으로 하는 연구들로서 다중 문서를 대표할 수 있는 하나의 구조 추출을 목표로 한다[3][4][5]. 여기에서는 하나의 구조 정의를 공유하는 문서집합을 대상으로 약간의 구조 정보의 손실을 감수하고서라도 최적화된 구조를 추출하고 있다. 그러나 하나의 구조를 공유하고 있더라도 실제 문서 인스턴스에 나타나는 구조는 크게 차이가 날 수 있다. 또 완전히 다른 엘리먼트에 다른 구조를 사용하는 문서의 공유 구조를 파악하기는 더욱 어렵다.

한 편, 발견된 구조에서 매칭되는 패턴을 찾는 문제는 컴파일러의 최적화 과정에서 트리 패턴 매칭 문제로 오랫동안 연구 되어 왔다[6]. 그러나 하나의 트리가 아닌 여러 트리간의 공유 구조를 밝혀야 한다. 또한 트리의 포함 관계를 따지는 연구[7][8]도 있으나 그 복잡도가 매우 크므로 방대한 문서데이타에 적용하기는 매우 어렵다.

2.2 순차 패턴 마이닝 알고리즘

본래 순차 패턴 마이닝 알고리즘[1]은 데이터베이스에서 사용자-정의 최소 지지도를 만족하는 트랜잭션의 시퀀스들 가운데 최대 시퀀스를 찾는 알고리즘이다. 연관

규칙과는 달리 트랜잭션의 발생 횟수만이 아닌 발생 순서(sequence)를 고려한다는 점에서 구조를 이루는 경로상에서 요소들의 순차를 고려 할 수 있다. 이 때 트랜잭션 시퀀스는 ' 우유' , ' 기저귀' , ' 맥주' 와 같이 고객이 구입한 물건에 시간 개념을 부과한 것이다. 따라서 방대한 고객정보 데이터베이스에서 각각의 고객이 구입한 물품들이 시간에 의해 시퀀스를 형성하고 이 정보들을 기반으로 순차 패턴 마이닝 알고리즘을 적용하여 고객들이 구매하는 물품과 그 순서 정보를 알아 낼 수 있다. 이러한 정보는 개인화(personalization)나 추천시스템(recommendation system)과 같이 개인의 구매성향을 파악하는데 이용되고 있다.

3. 변형 순차 패턴 마이닝 알고리즘

XML 구조간의 최대 유사 경로를 구하기 위하여 변형된 순차 패턴 마이닝 알고리즘을 제안한다. 연관 규칙과는 달리 트랜잭션의 발생 횟수만이 아닌 발생 순서, 즉 시퀀스를 고려한다는 점에서 구조를 이루는 경로상에서 요소들의 순차를 고려 할 수 있다. 다음 표 1 본래의 순차 패턴 마이닝 알고리즘의 5 단계-Sort, Litemset, Transformation, Sequence, Maximal-를 기준으로 변형된 개념을 요약하였다.

표 1. 변형된 개념

단계	본래 순차 패턴 알고리즘	변형된 순차 패턴 알고리즘
정렬 (Sort)	데이터베이스는 고객-아이디와 트랜잭션 시간으로 정렬된다. 따라서 본래의 트랜잭션 데이터베이스는 고객-시퀀스로 구성된 데이터베이스로 변경된다.	XML문서 구조에서 경로 표현(path expression)을 추출한다.
빈발항목 찾기 (Litemset)	최소지지도를 만족하는 빈발항목을 찾는다.	문서간에 유사하게 사용된 XML elelement를 식별한다.
변경 (Transform-ation)	고객 시퀀스는 빠른 검색을 위한 형태로 변경된다.	모든 추출된 경로들이 정수로 교체되는데 이때 유사하게 사용된 빈발 element는 같은 숫자로 재명명된다.
시퀀스화 (Sequence)	빈발항목 1, 2, ..., n 시퀀스를 최소 지지도에 입각하여 찾는다.	길이 1, 2, ..., n의 빈발 경로를 최소 지지도 (유사성)에 의해 찾는다.
최대 유사 경로 추출 (Maximal)	발견된 빈발 시퀀스 중 최대 빈발 시퀀스를 발견한다.	최대 유사 경로를 발견한다.

변형된 개념을 보면 먼저 경로의 중복을 제거한 최소화된 구조를 추출한다. 데이터베이스에서 트랜잭션을 시간으로 정렬하듯이 경로의 각 요소들은 구조의 트리의 레벨로 정렬화 한다. 다음으로 루트에서 단말 노드에 이르는 경로들을 탐색함으로써 두 문서에 모두 나타나는 1-시퀀스, 2-시퀀스 순으로 최대 n-시퀀스를 찾아 낸다. 이 때 시퀀스를 구성하는 요소간의 유사 행렬을 기반으로 유사한 트랜잭션을 인식할 수 있다. 빈발의 기준인 최소 지지도는 두 문서에 모두 나타나는 경로여야 빈발 경로라고 말할 수 있으므로 항상 100%가 된다. 그리고 포함관계가 있는 시퀀스를 제거하여 최대 시퀀스를 추출할 수 있다. 계산의 편의를 위해 추출된 경로를 유사한 엘리먼트를 고려하여 정수로 재명명하는 변경 단계까지를 전처리로 보고 다음 장에서는 표 1을 토대로 알고리즘을 설명한다.

4. 전처리

4.1 정렬 단계

설명의 편의를 위해 다음 표 2의 original path와 같은 예를 든다. 두 문서 a와 b의 경로를 추출하였다. 추출된 경로 집합을 기준(base) 문서를 a.xml로 보았을 때 PE_B 로, 이에 대해 비교(query) 문서를 b.xml로 보았을 때 PE_{Q1}로 보았을 때 모두 레벨로 정렬되어 있다. 최소화된 XML의 구조를 찾고 이로부터 경로를 추출해 내는 작업은 [8]에 자세히 설명되어 있다. a와 b 문서 모두 온라인서점의 문서로서 b문서가 좀 더 간단한 구조로 구성되어 있다. transformed path에 대해서는 다음 4.3 절에서 설명한다.

표 2. 경로 추출 및 변경의 예

Doc.	Original paths	Transformed paths
a.xml PE_B	book.title.⊥	<{1} {2} >
	book.bookinfo.page.⊥	<{1} {3} {4}>
	book.bookinfo.paperback.⊥	<{1} {3} {5}>
	book.bookinfo.edition.⊥	<{1} {3} {6}>
	book.bookinfo.date.⊥	<{1} {3} {7}>
	book.bookinfo.publisher.⊥	<{1} {3} {8}>
	book.bookinfo.ISBN.⊥	<{1} {3} {9}>
	book.bookinfo.size.⊥	<{1} {3} {10}>
	book.buyinginfo.price.list.⊥	<{1} {11} {12} {13} >
	book.buyinginfo.price.our.⊥	<{1} {11} {12} {14} >
	book.buyinginfo.price.save.⊥	<{1} {11} {12} {15} >
	book.contents.section.chap.⊥	<{1} {16} {17} {18} >
	book.reviews.customer.rating.⊥	<{1} {19} {20} {21} >
	book.writer.name.⊥	<{1} {22} {23} >
b.xml PE_{Q1}	bookinfo.title.⊥	< {3} {2} >
	bookinfo.price.list.⊥	< {3} {12} {13} >
	bookinfo.price.our.⊥	< {3} {12} {14} >
	bookinfo.author.name.⊥	< {3} {22} {23} >
	bookinfo.tableofcontents.section.chap.⊥	< {3} {16} {17} {18} >
	bookinfo.reviews.reviews_no. ⊥	< {3} {19} {24} >
	bookinfo.reviews.avg_rating.⊥	< {3} {19} {21} >

4.2 빈발 항목 찾기 단계

여기에서 빈발 항목이란 두 문서 모두에 나타나는 항목으로, 즉 유사하게 사용된 엘리먼트를 의미한다. 따라서 빈발 항목(유사 엘리먼트)을 찾기 위해서 다음 표3과 같은 매핑 테이블을 구축한다.

표 3. 매핑 테이블

no	element	No	element	no	Element
1	Book	9	ISBN	17	Section
2	Title	10	size	18	Chap
3	bookinfo	11	buyinginfo	19	Reviews
4	Page	12	price	20	Customer
5	paperback	13	list	21	rating, avg_rating
6	Edition	14	our	22	writer, author
7	date	15	save	23	Name
8	publisher	16	contents, tableofcontents		

매핑 테이블은 WordNet이나 사용자-정의 사전을 구축하여 자동화 할 수 있다. 표에서 보듯이 같은 번호 16

으로 매겨진 <contents>와 <tableofcontents>는 유사 엘리먼트로, 또한 동일한 엘리먼트 <bookinfo>는 3으로 표 2의 transformed path 표현으로 재명명된다.

4.3 변경 단계

이 단계에서는 앞 단계에서 서로 유사한 것으로 인식된 엘리먼트를 같은 수로 재명명한다. 따라서 공유 경로를 찾기 위해 계속해서 엘리먼트 스트링을 검색하는 대신 수의 비교로 매칭 시간을 단축 시킬 수 있다. 다음 알고리즘은 최소화된 XML 구조로부터 추출된 경로를 입력으로 하여 두 문서에서 유사한 빈발 항목을 찾고 이 항목을 토대로 경로 표현을 재명명하는 과정을 기술 하였다.

```
procedure Litemset&Transform ( PE_B: path expressions of a base
  document, PE_Q1..n : path expressions of query documents, SM :
  similarity matrix between elements)
returns L_1^{B•Q1..n}; // all large 1-paths between PE_B and PE_Q1..n
begin
// build a mapping table
    initialize MappingTable;
    for (k=1; PE_B ≠ nil or PE_Q1..n ≠ nil; k++) do
    begin
      E = read (one element from PE_B or PE_Q1..n);
      if (E ∉ MappingTable)
      then if (similar element of E in SM ∉ MappingTable)
           then insert MappingTable(k , E);
           else append MappingTable(similar element of E, E)
                // with the same index of similar element of E
    end
    // elements in path expressions are replaced by integers
    map elements of PE_B and PE_Q1..n to integers in MappingTable;
    generate new path expressions in which elements are
    represented in integers;
    // find all large 1-paths L_1^{B•Q1..n}
    for (k=1; k <= n ; k++) do
      foreach element E in PE_Qk do
          if (E ∈ PE_B)
            // minimum support 100% between PE_B and PE_Qk
          then insert E into L_1^{B•Qk};
      end
    end
end
```

그림 1. 빈발항목찾기 및 변경 알고리즘

5. 시퀀스화 및 최대 유사 경로 추출 단계

전단계에서 유사한 엘리먼트를 같은 인덱스로 재명명된 경로 표현을 생성하였다. 이제 매핑테이블의 길이 1의 빈발항목을 L_1이라 한다면 이 단계에서는 다음 알고리즘을 통해 길이 증가시키면서 L_2, L_3 ... L_n을 찾는다. 이 때 최소 지지도는 엘리먼트의 출현 횟수와 상관없이 두 문서에 모두 나타난 엘리먼트를 대상으로 한다.

```
procedure Sequence&Maximal (L_1^{B•Q1..n} : all large 1-path
expressions between a base document B and each query document,
PE_B: path expressions of a base document, PE_Q1..n : path
expressions of query documents)
returns ML^{B•Q1..n}; // all maximal large similar paths between PE_B
                                                and PE_Q1..n
begin
  for (i=1; i <= n; i++) do
  begin
    for (k=2; L_{k-1}^{B•Qi} ≠ ∅; k++) do
    begin
    C_k = New candidate-paths generated from L_{k-1}^{B•Qi};
    foreach path expressions in PE_B and PE_Qi do // pruning
      if (C_k ∈ PE_B and C_k ∈ PE_Qk) // minimum support 100%
          then L_k^{B•Qi} = C_k ;
      end
          length = k; // longest length of large paths L_k^{B•Qi}
    end
    for (j = length; j >= 1; j--) do   // maximal phase
      foreach j-large paths, L_j^{B•Qi} do
          delete all sub-paths of L_j^{B•Qi} ;
      end
    ML^{B•Qi} = { remained large paths in L^{B•Qi} }
  end
end
```

그림 2. 시퀀스화 및 최대 유사 경로 추출 알고리즘

따라서 표 2의 transformed path 표현과 매핑 테이블 정보를 기반으로 알고리즘을 적용하면 다음 유사 경로의 시퀀스를 얻을 수 있다. 그리고 밑줄 친 시퀀스가 최대 유사 경로가 된다.

표 4. 최대 유사 경로

Large 1-paths	Large 2-paths	Large 3-paths
<u><{2}></u>, <u><{3}></u>, <{12}>, <{13}>, <{14}>, <{16}>, <{17}>, <{18}>, <{19}>, <{21}>, <{22}>, <{23}>	<u><{12} {13}></u> <u><{12} {14}></u> <{16} {17}> <{16} {18}> <{17} {18}> <u><{19} {21}></u> <u><{22} {23}></u>	<u><{16} {17} {18}></u>

이렇게 얻어진 최대 유사 경로 기반으로 표 2의 transformed path에 적용해 보면 bold하게 표시된 경로의 시퀀스들이 두 문서의 공통적인 경로 시퀀스가 된다. 따라서 b.xml은 상당한 경로들이 a.xml에 포함되고 있음을 알 수 있다.

6. 결론 및 향후 연구

전혀 다른 구조 정의에 의해 작성된 문서라 하더라도 그 공유 정도를 파악하기 위해서 최소화된 구조로부터 경로 표현을 추출하고 이를 순차 패턴 마이닝 알고리즘의 개념을 변형하여 최대 유사 경로를 구할 수 있었다. 본 알고리즘은 XML 문서 마이닝의 전처리 과정에서 활용될 예정이다.

참고문헌

[1] R. Srikant and R. Agrawal. "Mining Sequential Patterns:Generalizations and Performance Improvements", In *Proc. of EDBT,* France, March 1996

[2] http://www.w3.org/

[3] Deutsch, Fernandez and Suciu. "Storing Semistructured Data with STORED", *In Proc. of SIGMOD*, pages 431-442, 1999

[4] Nestorov, Abiteboul, Motwani. "Extracting Schema from Semistructured Data", *In Proc. of SIGMOD*, pages 295-306, 1998

[5] Ke Wang and Huiqing Liu. "Discovering Typical Structures of Documents: a Road Map Approach", In *the Proc. of SIGIR*, pages 146-154, 1998

[6] Christoph M. Hoffmann and Michael J. O'Donnell. "Pattern Matching in Trees", *Journal of ACM 29(1)*, pages 68–95, Jan. 1982.

[7] Ira D.Baxter, Andrew Yahin, Leonardo Moura, Marcelo Sant'Anna, and Lorraine Bier. "Clone Detection using Abstract Syntax Tree", In *Proc. of the ICSM'98*, Nov. 1998

[8] Jung-Won Lee, Kiho Lee, Won Kim, "Preparations for Semantics-based XML Mining", In *Proc. of the ICDM '01*, Nov. 2001.