함수 변환 모델링에 의한 XML 문서의 유사성 비교에 대한 연구

이 호 석
호서대학교 공과대학 뉴미디어공학과
hslee@office.hoseo.ac.kr

A Study on XML Document Similarity based on Function Modeling

Ho Suk Lee
New Media Department, College of Engineering, Hoseo University

요 약

근래에 XML 문서가 인터넷에서 정보 교환의 방법으로 표준화됨에 따라, 많은 양의 데이터가 XML 문서 포맷으로 저장되고 있다. XML 문서의 유사성 연구는 크게 edit-distance를 이용하는 방법, 문서의 그래프 모델을 이용하는 방법, 문서의 매트릭스 모델을 이용하는 방법 등이 있다. 최근에는 문서를 encoding 하고 푸리에 변환을 이용하는 방법이 보고되었다. 본 논문에서는 XML 문서를 함수로 변환하여 모델링하여 문서의 구조적 유사성을 비교하는 방법을 제안한다. 제안된 방법을 사용하여 XML 문서를 함수로 모델링하였으며 XML 문서 간의 유사성을 비교해 보았다.

1. 서론

1.1 연구 동기

웹은 정보를 제공하고 공유하는데 있어서 최적의 수단이 되었다. XML(eXtensible Markup Language) 문서는 사용자 자료, 시스템 자료, 네트워크 자료, 그리고 웹 서비스를 나타내는데 사용되고 있다[1][2]. XML 문서의 유사성을 파악하는 것은 XML 문서의 저장과 검색에 매우 유용하여 많은 연구가 진행되었다. 유사성 비교 연구에는 edit-distance 방법, 문서의 그래프 모델링 방법, 문서의 매트릭스 모델링 방법, 그리고 최근에는 문서의 encoding 방법 등이 보고되었다. 본 논문에서는 XML 문서의 구조적 유사성을 효율적으로 비교할 수 있도록 XML 문서의 함수 모델링 방법을 제안한다.

1.2 관련 연구

참고문헌 [3]에서는 이산 푸리에 변환을 사용하여 질의어에 대한 time series를 주파수 영역으로 변환하여 질의어의 유사성을 비교하였다. 푸리에 변환을 사용하는 중요한 이유는 소수의 저주파 성분이 중요한 요소를 표현하기 때문에 질의어 비교가 용이하기 때문이다. 참고문헌 [4]에서는 어휘뿐만이 아니고 구(phrase)를 사용하여 웹 문서의 유사성을 비교하였다. 또한 이 방법에서는 DIG(Document Indexing Graph)를 사용한 그래프 비교 방법을 사용하였다. 참고문헌 [5]에서는 S-GRACE 라고 불리는 계층적인 알고리즘을 제안하였다. 이 알고리즘은 s-graph 라고 불리는 구조 그래프(structure graph)를 사용하여 XML 문서간의 거리를 측정하였으며 이를 이용하여 문서들을 클러스터링하였다. 참고문헌 [6]에서는 edit-distance를 사용하여 XML 문서의 유사성을 비교하였다. 참고문헌 [7]에서는 이전 버전과 현재 버전의 XML 문서 사이의 차이를 계산하는 방법을 edit script 라는 개념을 사용하여 제시하였다. 참고문헌 [8]에서는 XML 문서에 대한 새로운 클러스터링 알고리즘을 제안하였다. 참고문헌 [8]에서는 XML 문서의 태그(tag)들을 encoding하여 숫자로 표현한 다음에 숫자들의 발생 빈도를 조사하여 문서의 유사성을 측정하는 방법을 제안하였다. 즉, [8]에서는 XML 문서의 태그인 엘리먼트와 에트리뷰트를 encoding 함수를 정의하여 숫자로 변환하였다. Encoding 함수에는 direct encoding, pairwise encoding, 그리고 nested encoding 함수가 있다. 이 함수들은 XML 문서에서 사용하는 엘리먼트와 에트리뷰트들을 신호처리에서 사용하는 개념인 impulse로 간주하여 숫자로 변환한다. 이렇게 하면 XML 문서는 일련의 숫자들의 순서가 되며 이 숫자들을 디지털 신호의 신호 값으로 간주한다. 다음에 이 신호 값들에 대하여 이산 푸리에 변환을 수행하여 주파수 영역에서 XML 문서의 구조적 유사성을 조사하였다. 우선 XML 문서에서 태그들의 집합을 구한다. 다음에 XML 문서를 읽으면서 발견하는 태그들에 대하여 집합에 있는 태그 순서를 부여한다. 이 방법을 direct encoding 이라고 한다. Pairwise encoding 방법은 연속하는 두 개의 태그를 함께 고려하여 숫자를 부여한다. 이 경우 숫자는 두 개의 연속하는 태그의 위치에 따라 숫자가 부여된다. Nested encoding은 루트에서 태그에 이르는 경로인 path에 따라

숫자를 부여한다. 또한 문서 encoding 이란 문서에 대하여 숫자들의 순서(sequence)를 부여하는 함수이다. 이때 문서 encoding 함수는 문서의 구조를 손실이 없이 encoding 할 수 있어야 한다. 가장 간단한 문서 encoding 함수는 문서의 skeleton에 나타나는 태그에 대하여 태그 encoding 함수를 적용하는 것이다. 여기서 skeleton이라는 것은 문서에 존재하는 모든 태그들을 순서대로 정리한 것이다. 다음에는 선형 문서 encoding 함수를 생각할 수 있다. 여기서 선형이라는 것은 태그 encoding 함수 값을 계속 누적하여 문서 encoding 함수 값으로 하는 것을 의미한다. 또한 태그의 네스팅 레벨을 고려한 멀티레벨 문서 encoding 함수를 생각할 수 있다. 마지막으로 이산 푸리에 변환을 사용한 문서 간의 비교는 다음 수식으로 정의된다.

$$\mathrm{dist}(d_1,d_2) = \left(\sum_{k=1}^{M/2}(|[DFT(h_1)](k)|-|[DFT(h_2)](k)|)^2\right)^{1/2}$$

여기서, dist는 문서 간의 차이를 나타낸다. d_1과 d_2는 문서를 나타낸다. 최종 DFT는 원래 DFT와 DFT의 보간 값까지를 포함한다. h_1 = enc(d_1)이고 h_2 = enc(d_2)이다. enc는 문서 encoding 함수이다. M은 보간 값까지를 포함한 최종 모든 신호 값이며 $M=N_{di} + N_{dj} -1$ 이다. dist(d_1,d_2)는 삼각 부등식을 만족하는 metric distance이다.

2. 본론

2.1 문서 모델링 함수

XML 문서 간의 유사성을 비교하기 위해서는 적합한 문서 모델링이 필요하다. 문서의 모델링에는 어휘 모델링, 구조 모델링, 의미 모델링을 생각할 수 있다. 참고문헌 [8]에서 사용한 태그와 문서 encoding 함수는 각각 어휘와 구조 모델링으로서 문서를 time series로 변환하였다고 할 수 있다. 그리고 푸리에 변환을 사용하여 time series를 다시 주파수 영역으로 변환하고 소수의 저주파 성분을 비교하여 문서의 유사성을 비교하였다고 할 수 있다.

이러한 방법이외에 문서를 함수로 모델링하는 방법을 생각해 보자. 문서를 time series로 모델링하는 방법보다는 개념적으로 더 적합하다고 할 수 있다. 왜냐하면 XML 문서를 구성하는 태그들은 시간 개념을 포함하지 않기 때문이다. 우선 문서의 어휘 모델링으로서는 엘리먼트의 집합과 시퀀스 그리고 에트리뷰트의 집합을 생각한다. 여기서, 집합은 중복된 원소를 포함하지 않고 오직 시작 태그(start tag)만을 포함한다. 시퀀스는 원소의 나열로서 중복을 허용하며 또한 시작 태그와 끝 태그(end tag)를 포함하는 것으로 정의한다. 엘리먼트가 에트리뷰트를 가지고 있을 때에는 엘리먼트 시퀀스에 있는 에트리뷰트를 가리키는 주소를 표기하여 에트리뷰트를 가리킬 수 있도록 한다. 구조 모델링으로서는 엘리먼트의 시퀀스를 함수로 변환하는 방법을 생각한다. 예를 들어, 엘리먼트 시퀀스를 X 축으로 하고 시퀀스에 포함된 각 엘리먼트를 X 축의 값으로 간주하고 이 값으로부터 Y 값을 구하여 문서를 함수로 변환하는 것이다.

문서를 함수로 변환하는 방법은 다음과 같다. (1) 시퀀스에 있는 첫 번째 엘리먼트를 X_1 위치에 놓는다. 이 엘리먼트의 Y 값을 Y_1으로 한다. (2) X_1 에 위치한 엘리먼트와 관련된 에트리뷰트의 개수를 조사한다. 개수가 n 이면 n 만큼 X_1 위치에 있는 엘리먼트를 X 축에 연속하여 위치시킨다. (3) 개수가 0이면 다음 엘리먼트를 조사하여 X_2 에 위치시킨다. 만약 다음 엘리먼트의 내포 레벨(nesting level)이 하나 증가되어 있으면 X_2 의 Y 값은 Y_1 + 1로 한다. 만약 증가되어 있지 않으면 같은 값을 사용한다. (4) 시퀀스에서 끝 태그를 만나면 해당 시작 태그를 시작할 때의 Y 값을 해당 이번 X 위치의 Y 값으로 기록한다. (5) 마지막으로 엘리먼트의 시퀀스를 모두 사용하면 종료한다. 다음은 위의 과정을 C 형태의 코드로 표현한 것이다.

```
#define   N   100    // NUMBER_ELEMENT

int x[N], y[N];  // X:X축, Y:Y축
int c_nlevel, p_nlevel =1 ;

void XML_Func_Modeling(int element_set[], int element_seq[],
int attr_set[])
{
    int  i, j, k, m, n, p;
    int  y_save, count;

    for (i=1; i<number_element_sequence; i++) {
        x[i] = element_seq[i];
        if (start_tag(x[i])) then {
            j = i; k = nesting_level(x[i], i);
            if (increased(k)) then y[i] = j + 1;
                              else y[i] = j;
            y_save = y[i];
            count = number_attr(x[i]);
            for (m=0; m<count; m++) {
                p = i + m + 1; x[p] = x[i]; y[p] = i;
            }
        } else {   // x[i] is the end tag
            y[i] = y_save - 1; y_save = y_save - 1;
        }
    }
}

int nesting_level(x, j)
{
    int  j, k;

    if (j == 1) then {
        c_nlevel = p_nlevel = 1;
        return (c_nlevel);
    } else k = j - 1;
    if (start_tag(x[k])) then {
```

```
        p_nlevel = c_nlevel;
        c_nlevel = c_nlevel + 1;
        return (c_nlevel);
    }
}

boolean increased(k)
{
   int k;
   if (c_nlevel > p_nlevel) then return true;
                            else return false;
}
```

문서의 의미는 엘리먼트 태그와 태그가 포함한 문자열 그리고 에트리뷰트와 에트리뷰트가 나타내는 문자열을 비교하는 것으로 모델링할 수 있다. 비교 대상이 되는 XML 문서에 대하여 태그 배열과 태그 문자열 그리고 에트리뷰트 배열과 에트리뷰트 문자열을 구성하여 비교함으로서 문서 의미의 유사성을 비교한다.

2.2 실험

두 개의 비슷한 XML 문서와 서로 다른 XML 문서를 대상으로 실험을 수행하였다. 다음은 실험 대상 XML 문서이다[8].

```
<xml document="A">
   <book year="1999">
      <title>XML Bible</title>
      <author>Elliotte Rusty Harold</author>
      <publisher>IDG Books</publisher>
   </book>
</xml>

<xml document="B">
   <book year="2003">
      <title>XML 1.1 Bible</title>
      <author>Elliotte Rusty Harold</author>
      <publisher>John Wiley</publisher>
   </book>
</xml>

<xml document="C">
   <article>
        <title>XML Document Similarity</title>
      <author>John Smith</author>
      <journal>The XML Journal
         <s_page>100</s_page>
         <e_page>500</e_page>
      </journal>
   </article>
</xml>
```

다음은 함수로 변환된 XML 문서를 나타낸다. 이 결과는 위에서 제안된 알고리즘을 적용하여 생성하였다.

f(a) X	Y	f(b) X	Y	f(c) X	Y
1	1	1	1	1	1
2	2	2	2	2	2
3	3	3	3	3	3
4	3	4	3	4	3
5	3	5	3	5	3
6	2	6	2	6	4
7	1	7	1	7	4
				8	3
				9	2
				10	1

변환된 함수 f(a), f(b), f(c)를 비교하여 보자. 함수들은 예로 사용한 XML 문서 A, B, C의 구조를 매우 잘 반영하고 있음을 알 수 있다. 그리고 문서 A와 B는 동일하기 때문에 f(a)와 f(b)도 동일한 모양을 보여주고 있다. 반면에 문서 C는 구조가 다르기 때문에 함수 f(c)는 f(a)와 서로 다른 모양을 보여주고 있다. 이러한 결과로부터 XML 문서의 함수 변환 모델링은 XML 문서의 구조를 비교하는데 적합함을 알 수 있다. 또한 이 결과는 참고문헌 [8]의 Fig.4에서 제시한 결과보다 더 우수하다고 할 수 있다. 그리고 최종적으로 문서 간의 유사성 비교는 어휘 모델링, 함수 변환에 의한 구조 모델링, 그리고 의미 모델링 간의 비교를 기반으로 구할 수 있다. 어휘 모델링의 비교는 배열 연산에 의하여 그리고 의미 모델링의 비교는 문자열 연산에 의하여 구할 수 있다.

3. 결론

본 논문에서는 XML 문서의 유사성 비교를 위하여 문서를 함수로 변환하여 모델링하고 비교하는 새로운 방법을 제안하였다. 제안된 함수 변환 모델링 방법은 XML 문서의 구조를 잘 표현하여 문서의 유사성 비교를 용이하게 한다. 다음 연구로는 orthogonal 함수를 사용하여 모델링 함수를 주파수 영역에서 비교해 보는 것이다.

참고문헌

[1] Elliotte Rusty Harold, XML 1.1 Bible (3rd ed.), John Wiley & Sons, 2003.

[2] Elliotte Rusty Harold, Processing XML with Java, Addison-Wesley, 2003.

[3] Rakesh Agrawal, Christos Faloutsos, Arun Swami, "Efficient Similarity Search in Sequence Databases," Proc. of the 4th Int'l Conf. of Foundations of Data Organization, pp.69-84, 1993.

[4] Khaled M. Hammouda, Mohamed S. Kamel, "Efficient Phrase-Based Document Indexing for Web Document Clustering, IEEE Trans. on Knowledge and Data Engineering, Vol. 16, No. 10, pp.1279-1296, October 2004.

[5] Wang Lian, David Wai-lok Cheung, Nikos Mamoulis, Siu-Ming Yiu, "An Efficient and Scalable Algorithm for Clustering XML Documents by Structure," IEEE Trans. on Knowledge and Data Engineering, Vol. 16, No. 1, pp82-96, January 2004.

[6] Andrew Nierman, H. V. Jagadish, "Evaluating Structural Similarity in XML Documents," Proc. of the 5th Int'l Workshop on Web and Databases, 2002.

[7] Kyong-Ho Lee, Yoon-Chul Choy, Sung-Bae Cho, "An Efficient Algorithm to Compute Differences between Structured Documents," IEEE Trans. on Knowledge and Data Engineering, Vol. 16, No. 8, pp.965-979, August 2004.

[8] Sergio Flesca, Giuseppe Manco, Elio Masciari, Luigi Pontieri, Andrea Pugliese, "Fast Detection of XML Structural Similarity," IEEE Trans. on Knowledge and Data Engineering, Vol. 17, No. 2, pp.160-175, February 2005.