HTML 특성을 고려한 트리 편집 거리 측정 알고리즘의 개선

김연정[O] 박제현 최중민
한양대학교 컴퓨터공학과
{yeonjung[O], jhpark, jmchoi}@cse.hanyang.ac.kr

Improvement of an algorithm for tree-editing distance measure regarding the features of HTML

Yeonjung Kim[O] Jeahyun Park Joongmin Choi
Dept. of Computer Science and Engineering Hanyang University, Korea

요 약

웹 문서를 대상으로 하는 정보 추출이나 웹 마이닝에 관한 연구가 활발히 진행되면서 특히, 웹에서 나타나는 구조적 패턴을 이용해 정보를 추출하는 방법에 대한 연구가 이루어 지고 있다. 기존의 연구는 HTML을 단순 문자열로 취급하였으나 연구가 거듭됨에 따라 트리로 접근하는 방안에 대해 논의가 되었으며 성능 또한 우수한 것으로 평가되고 있다. 하지만, 기존의 트리 편집 거리의 기법은 모든 노드가 동일한 값을 가진다는 가정하에 진행되는 것으로 HTML의 특성과는 맞지 않다. HTML은 브라우저에 정보를 보여주기 위한 도구이며 실제 브라우저에 보여지는 내용의 비율이 트리에서의 노드의 비율과 항상 같은 것은 아니기 때문이다. 이 논문에서는 위와 같은 HTML의 특성을 이용하여 노드가 가진 정보의 크기에 따라 서로 다른 비율의 기여도를 부여하고, 이를 고려한 개선된 트리 편집 거리 측정 알고리즘을 이용하여 좀더 나은 패턴 추출 방법을 제안하고자 한다.

1. 서론

월드 와이드 웹(World Wide Web)의 비약적인 발전으로 인해 하루에도 많은 양의 문서들이 웹을 통해 생성 되고 있다. 우리 일상의 많은 부분이 웹과 연결되어 있으며 우리가 필요로 하는 많은 정보들은 웹을 통해서 습득되고있다. 하지만, 웹 상의 문서가 넘쳐날수록 사용자가 정말 필요로 하는 문서의 선별은 쉽지 않다. 그래서 웹 상의 문서를 대상으로 하는 정보추출 기법들이 많이 연구되어 오고 있다. 특히 웹 페이지상에서 어떠한 구조적 패턴을 가지고 반복되어 나타나는 정보에 대해 집중하기 시작하였는데 이는 같은 패턴을 가지고 반복적으로 페이지상에서 나타나는 정보가 각 페이지에서 이야기 하고자 하는 가장 중요한 정보일 것이라는 가정에 의해서이다. 실제 많은 웹 페이지를 보면 상품명, 서비스 목록 등의 핵심 정보들이 같은, 혹은 비슷한 패턴을 가지고 페이지상에 등장함을 알 수 있다. 기존의 연구들은 이러한 패턴을 발견하기 위한 패턴 매칭 기법으로 순수한 자연어에서의 기법인 문자열 편집거리[1]등과 같은 방법을 사용하였다[6,7,8,10,12]. 하지만 HTML은 트리로 접근이 가능하고 단순 문자열로의 접근에 비해 트리의 구조정보를 이용한 패턴 매칭 방법이 훨씬 성능이 좋음을 이후의 연구에서 확인할 수 있다[11,13,14,16]. 기존의 트리 매칭 방법들은 트리의 각 노드의 값이 모두 같다는 가정이 깔려있다. 하지만 HTML 트리의 각 노드는 실제 같은 값을 가지고 있지 않다. 웹 페이지가 동적으로 프로그램에 의해서 자동적으로 실제 브라우저상에 보여지는 정보의 양에 비해 HTML 태그의 양이 엄청나게 많아지는 예도 있다. 그림 1의 경우 브라우저 상에는 비슷한 패턴을 가지는 정보들로 인식된다. 하지만 그림 2에서 나타나는 각 데이터 개체의 HTML 트리 정보는 확실이 다르다는 것을 알 수 있다. 즉, 브라우저상에 나타나는 정보의 양이 전체 패턴에 미치는 비율이 HTML 트리 상에서의 비율과 일치하지 않다는 것이다. 이 논문에서는 이 점에 착안하여 기존의 트리 편집 거리를 HTML의 이러한 특성에 맞게 개선한 알고리즘을 제안하고자 한다.

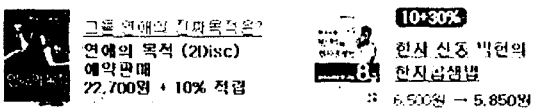


그림 1 비슷한 패턴을 가지는 데이터 오브젝트

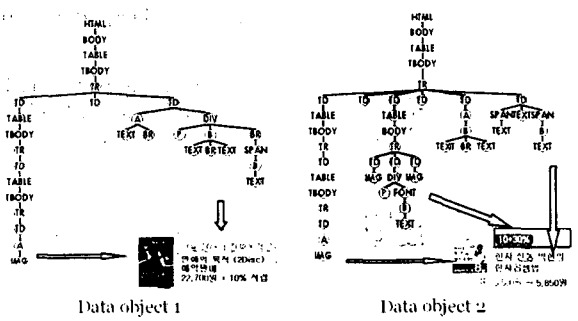


그림 2 그림 1의 HTML 트리 정보

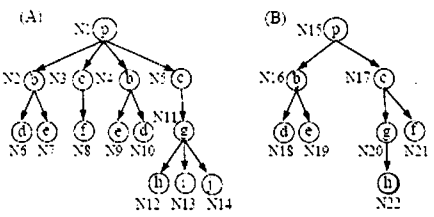
2. 기존의 트리 편집 거리

문자열 편집거리[1]와 유사하게 트리의 편집거리[2,3]또한 두 트리 T_1을 T_2로 변형하기 위한 최소 조작비용을 이야기한다. 그러나 트리 편집 거리는 간혹 같은 조작 정의 하에 매핑이라는 개념으로도 사용된다[3,4]. 이 논문에서는 제한된 매핑 알고리즘인 simple tree matching(STM) [5]을 개선하였다. 그림 3과 그림 4의 simple_Tree_Matching 알고리즘의 핵심은 두 트리의 같은 레벨에서만 매핑이 이루어지며 매핑이 되는 노드의 값은 "1"이라는 것이다. 다시 말해, 모든 노드가 노드 값으로 1을 가짐으로써 전체 트리에서의 비중을 같게 보고 있으며 만약 두

트리의 노드가 매핑이 된다면 각 트리의 노드값의 평균 비용 인 $AVG(1,1)=1$ 이 매핑의 비용이 된다.

```
Algorithm  simple_Tree_Matching(A,B)
1.  if the roots of the two trees A and B contain distinct symbols then return (0)
2.  m =the number of first-level subtrees of A
3.  n =the number of first-level subtrees of B
4.  Initialization, M[i,0] =0 for i = 0, ..., m
                    M[0,j] = 0 for j = 0, ..., n
5.  for i =1 to m do
6.    for j =1 to n do
7.      M[i,j]  max(M[i, j-1], M[i-1,j], M[i-1,j-1]+W[i,j]
8.        where W[i,j] = simple_Tree_Matching( A_i, B_j )
9.          where A_i and B_j are the ith and jth first-level subtrees of A and B, respectively
10.   od
11. od
12. return (M[m,n]+1)
```

그림 3 simple_Tree_Matching 알고리즘



(C) $M_{[i,j]}$

	0	1 (N16)	2 (N16-N17)
0	0	0	0
1 (N2)	0	3	3
2 (N2-N3)	0	3	5
3 (N2-N4)	0	3	5
4 (N2-N5)	0	3	**6**

(D) $W_{[i,j]}$

	1 (N16)	2 (N17)
1 (N2)	**3**	0
2 (N3)	0	2
3 (N4)	2	0
4 (N5)	0	**3**

그림 4 (A) 트리 A ; (B) 트리 B; (C) N1과 N15의 첫번째단계의 서브트리를 위한 M 행렬; (D)N1과 N15의 첫번째 단계의 서브트리를 위한 W 행렬

3. 트리 편집 거리 개선 알고리즘

기존의 트리 편집 거리에서 각 노드의 값은 " 1" 이며 매핑이 될 때는 두 노드의 평균값을 매핑비용으로 부여한다. 하지만 HTML의 특성상 HTML 트리를 구성하는 모든 노드가 같은 값을 가진다고는 볼 수 없다.

3.1 HTML 트리의 정보 기여도의 계산

HTML의 태그 중 실제 정보를 표현하는 태그는 <IMG> 태그와 일반 text이다.(앞으로 일반 text는 <TEXT>태그로 표현하겠다.) 이 태그들에서 나타나는 정보의 사이즈(infosize)의 계산이 먼저 되어야 한다. 그림 5는 정보량의 기여도의 예이다.

3.1.1 <IMG> 태그에서의 정보량 산출 방법

HTML 페이지에서의 이미지는 정보를 표현하는 중요한 수단이다. <IMG> 태그 속성 중 src 속성은 실제 이미지 주소를 가지고 있다. 이 정보를 이용해서 이미지의 가로와 세로의 크기를 알아낼 수도 있다. 하지만 실제 이미지의 크기보다는 <IMG> 태그상의 width, height 속성값이 우선한다.

(A) <img src="http://.../abc.jpg" width="40" height="10">
(B) <img src="http://.../abc.jpg" height="10">
(C) <img src="http://.../abc.jpg" >

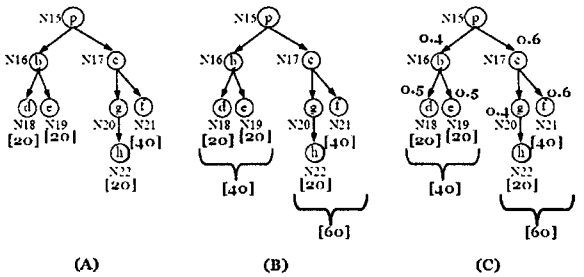


그림 5 정보량의 기여도 (A)각 단말 노드의 정보량. N18, N19, N22의 각각 정보의 양이 20이고 N21은 40이다; (B)N16의 정보의 양은 자식 노드 N18, N19의 합인 40이다; (C)정보의 양을 토대로 각 노드의 정보 기여도는 N16은 0.4, N17은 0.6이다.

위의 코드를 예로 들어보자. (A)의 경우 이미지의 가로크기와 세로크기는 40, 1 이다. (B)의 가로크기와 세로크기는 " abc.gif" 의 가로크기, 10 이고, (C)는 실제 " abc.gif"의 가로, 세로 크기이다. 실제 브라우저상에 얼마만큼의 크기로 표현되고 있느냐는 것이 중요하기 때문이다. 위와 같은 우선순위에 의해 추출된 width, height 값으로 정보의 크기(infoSize)가 구해진다.

$$infoSize = \frac{width * height}{\alpha} (\alpha > 0) \qquad (1)$$

여기서 α 의 값으로 나눠주는 것은 실제 이미지 사이즈와 텍스트간의 크기의 비율을 맞춰주기 위한 것으로 이 논문에서는 " 20" 을 사용하였다.

3.1.2 <TEXT> 태그에서의 정보 산출

<TEXT> 태그에서는 해당 문자열의 바이트 수($B(text)$)를 알 수 있다. 이 길이에 <TEXT> 태그의 글자 크기 fs($fs(text)$)를 곱해준다. 만약 글자 크기에 대한 언급이 없으면 기본 크기인 "12"를 곱해준다.

$$infoSize = B(text) * fs(text) \qquad (2)$$

다음 소스의 경우로 예를 들자면

(A) ....Hello...
(B) ... Hello...

(A)의 경우 정보의 크기는 8*5(Hello의 바이트 수)=40이고 (B) 역시 따로 언급된 텍스트 태그의 사이즈가 없으므로 12*5=60이다.

3.2 정보량 산출에 따른 각 서브트리의 정보의 기여도

그림 6의 (B)에서 <IMG> 노드의 정보의 크기는 20이고 <text> 노드의 크기는 48이다. <BODY> 노드에서 각 A 서브트리와 B 서브트리의 정보의 양은 20,60이므로 정보의 기여도는 29%(20/68), 71%(48/68)이다. 실제 브라우저상의 데이타인 그림 6의 (A)를 보면 각 정보의 기여도가 29%, 71%가 적합한 비율이란 것을 알 수 있을 것이다

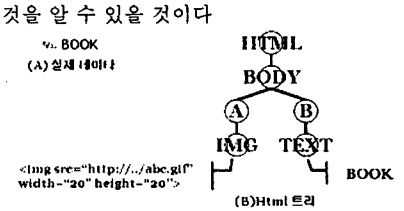
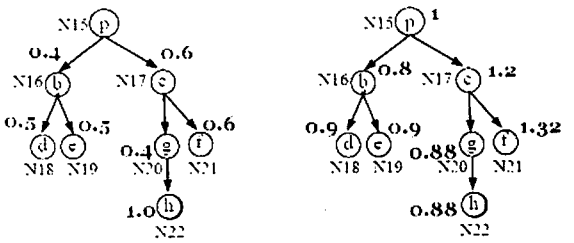


그림 6 각 서브트리별 정보의 기여도의 예제

3.3 정보의 기여도를 근거로 노드값 산출

기존의 트리의 매핑 비용은 같은 레벨의 두 노드가 같을 때 " 1" 이었다. 즉, 각 노드의 값이 " 1" 이고 두 노드가 같을 때 두 노드 값의 평균값인 " 1" 이 매핑의 비용으로 할당되었다고 생각할 수 있다. 하지만 HTML 트리에서의 각 노드는 표현하고자 하는 정보의 기여도에 따라 달라지므로 노드가 갖는 값 또한 달라지게 된다.



(A) 각 노드의 정보 기여도 (B) 각 노드의 노드값

그림 7 (A) 각 노드의 정보의 기여도($W(n)$)의 표시 트리; (B)각 노드의 노드 값($V(n)$) 표시 트리

그림 7 의 (A) 에서 각 노드의 값($V(n)$)을 산출하면 다음과 같다.

$$V(n) = (S(n) + (V(parent) - 1)) * W(n) \qquad (3)$$

식 (3)의 $S(n)$ 은 n노드의 형제 노드의 수 이고, $V(parent)$ 는 노드 n의 부모의 노드 값이며 $W(n)$은 노드의 가중치이다. 루트의 경우 부모 및 형제가 없으므로 노드의 값($V(n)$)은 1이다. 그림 7의 (B)의 N16의 노드 값은 형제 노드의 수가 2이고 부모의 노드값-1은 0, 그리고 $W(N16)$ 은 0.4이므로 (2+0)*0.4=0.8이다. 역시 이와 같은 방법으로 각 노드 N17,N18,...,N22의 노드 값을 산출하면 그림 7의 (B)가 될 것이다.

4. 기존의 트리 편집 거리 알고리즘(STM)과 HTML 성능을 고려한 개선된 알고리즘(HTM)의 비교

STM의 유사도와 HTM의 유사도는

$$S(STM) = \frac{Simple\_Tree\_Matching(A,B)}{AVG(|A|,|B|)} \qquad (4)$$

$$S(HTM) = \frac{HtmlTree\_Matching(A,B)}{AVG(|A_e|,|B_e|)} \qquad (5)$$

이다. 그림 2의 두 데이터 개체를 각각의 알고리즘에 의해 유사도를 비교해 보았다. STM에 의한 $S(STM)$은 0.63인데 반해 HTM에 의한 $S(HTM)$ 은 0.67로 0.04정도의 유사도의 차이를 보였다. 실제 HTML 트리에서 각 서브트리의 유사도를 비교함에 있어 임계치가 0.65였다면 (5)의 식에 의해서만 같은 패턴으로 인정되었을 것이다. 임계치를 낮게 잡게되면 (4),(5)식 모두 같은 패턴으로 인정하게 되겠지만 임계치가 낮아지게 되면 오히려 잡음이 패턴으로 인정될 수도 있다. 따라서 임계치를 낮추는 것만으로는 해결되지 않는 것이다. 대신 이 논문에서 제안하는 HTM 알고리즘을 사용한다면 잡음 없이 유용한 패턴을 추출하는 성능을 개선할 수 있을 것이다. 또한 자동화된 편집 과정에서 생기는 불필요한 태그 정보를 배제함으로써 잡음 감소 효과도 가지게 된다.

5. 결론 및 향후 연구과제

HTML 트리는 기존의 트리 유사도 비교 방식으로 처리되기엔 일반 트리와는 다른 특징을 가진다. 일반 트리는 각 노드의 값이 모두 동일한데 반해 HTML 트리는 각 노드의 값이 그 노드가 나타내려는 정보의 양의 기여도에 따라 달리 적용되어야 한다. 따라서 HTML 트리를 비교하기 위해서는 기존의 트리 비교 알고리즘으로는 적당하지 않다는 의미이다. 이에 이 논문에서는 HTML의 이러한 특성을 반영한 개선된 트리 편집 거리 알고리즘(HTM)을 제안하였다. 이는 기존의 STM보다 향상된 성능을 보이며 HTML의 특성에 적합하고 다양한 분야에 응용될 수 있을 것이다.

6. 참고 문헌

[1] W. J. Masek and M. S. Paterson. " A faster algorithm computing string edit distances" . Journal of Computer and System Sciences, Vol.20: pp.18-31, 1980

[2] valiente, G. Tree edit distance and common subtrees. Research Report LSI-02-20-R, Univ. Politecnica de Catalunya, Barcelona, Spain, 2002

[3] Tai, K. The tree-to-tree correction problem. J. ACM, 26(3):422 433, 1979

[4] Gabriel V. An Efficient Bottom-Up Distance Between Trees, 2001

[5] Yang, W. " Identifying Syntactic Differnces Between Two Programs" Softw. Pract. Exper.,21(7):739 755, 1991.

[6] Embley, D., Jiang, Y and Ng, Y. " Record-Boundary Discovery in Web Documents." , SIGMOD-99

[7] Buttler, D., Liu, L., Pu, C. " A fully Automated Object Extraction System for the World Wide Web" Proc. 21st ICDCS 01, IEEE CS Press, 2001, pp.361-370

[8] Crescenzi, V., Mecca, G. and Merialdo, P. " Roadrunner: Towards Automatic Data Extraction from Large Web Sites" , Proc. Of the 24th VLDB Conference, Roma, Italy, 2001

[9] Ajay H., Stephane B. " Information Extraction-Tree Alignment Approach to Pattern Discovery in Web Documents" EEXA 2002, LNCS 2453, pp.789~798, 2002

[10] Liu, B., Grossman, R. and Zhai, Y. " Mining Data Records in Web Pages." Intelligent Systems(IEEE) 11,12, 2004, pp.49~55

[11] Shiren Y., Tat-Seng C. " Detecting and Partitioning Data Object in Complex Web Pages" Web Intelligence 2004, pp. 669~972

[12] Chia-Hui C., Shih-Chien K. " OLERA: Semisupervised Web-Data Extraction with Visual Support" Intelligent Systems(IEEE)11,12, 2004, pp.56~64

[13] Reis, D. Golgher, P., Silva, A., Laender, A. " Automatic Web News Extraction Using Tree Edit Distance" WWW-04, 2004

[14] Yanhong Z., Bing L. " Web Data Extraction Based on Partial Tree Alignment" ACM 2005

[15] Robert B. D., Oren E., and Eaniel S. W. " A Scalable Comparison-Shopping Agent for the World-Wide Web" Agents-97

[16] Chia-Hui C., Shao-Chen L. " IEPAD: Information Extraction Based on Pattern Discovery" , www10' 01, may 1-5, 2001, HongKong, ACM