독립적인 질의 경로들을 사용하여 이질적인 문서들을 검색하는 XML 문서 검색 모델

(XML Document Retrieval Models for Heterogeneous Data Set using Independent Regular paths)

유 신 재 † 민 경 섭 † 김 형 주 ††

(Shinjae Yoo) (Kyung-Sub Min) (Hyoung-Joo Kim)

요 약 XML 문서는 태그를 가지고 있고 이 태그가 중첩됨에 따라 구조를 나타낼 수 있다. XML 문서가 DTD를 가지지 않거나 여러 곳에서 XML 문서를 모았을 때 그 구조는 비정규적 일 수 있다. 사용자는 이러한 비정규적인 구조에 대해 잘 알기 어려우며 설사 잘 알고 있다고 하더라도 실수하기 쉽다. 특히 비정규적인 구조를 가지는 문서들에 대해 정확한 구조질의를 작성하는 것은 더욱 어렵다. 따라서 사용자는 구조가 없거나 있다 하더라도 적은 양의 구조정보 만을 기술하는 일반적인 질의를 작성하게 된다. 이런 환경에서 구조 정보를 이용하여 문서의 순위결정에 이용하고 사용자 구조 질의와 문서 구조간의 차이에 대해 보상해 주는 검색 모델을 제안한다. 질의 처리를 단순화하기 위하여 질의 경로간의 독립을 가정하였다. 이 가정은 질의 언어의 표현능력의 저하를 가져올 수 있는데 이를 해결하는 질의 모델도 제시한다. 지금까지 XML 문서를 위한 테스트 컬랙션이 없었기 때문에 TIPSTER 컬랙션에서 일부 문서를 추출하여 작은 테스트 컬랙션을 만들고 여기에 구조가 없는 질의를 수행하여 제시한 검색 모델의 유용성을 보였다. 실험 결과 벡터 모델에 비하여 평균 67%의 정확률 개선효과를 얻을 수 있었다.

키워드 : XML, 검색모델, 검색엔진, 정보검색

Abstract An XML document has a structure which may be irregular. It is difficult for end-users to comprehend the irregular document structure exactly. For these XML documents, an end-user has a difficulty in using structured query. Therefore, an end-user formulates no structured query or a query which has a little structure information. In this context, we propose new retrieval models which use the structured information for ranking and compensate the difference between user query structure and document structure. To ease with querying, we assume the independence among querying paths which represent structural constraints. Since this assumption makes degradation of the expression power of a query language, we also propose a model which overcome this problem. As there had been no test collections for XML documents, we made a small test collection from TIPSTER of the TREC and experimented on this collection without a structured query. From this experiment, we showed that our models improve average precision about 67% over conventional Vector-Space model.

Key words : XML, retrieval model, retrieval engine, information retrieval

1. 서 론

XML(eXtended Markup Language)[1]은 서로 다른 시스템간의 데이터 교환이나 문서를 기술하기 위해서 W3C에서 제안한 표준 언어이다. 웹 페이지, 전자 카달로그, 전자 책(e-book), 신문부터 서로 다른 시스템간의 데이터 전송을 위해 점점 많이 사용되고 있다. 특히 한 기관이나 조직의 모든 데이터를 XML로 통합하여 저장, 관리, 검색하는 시스템들까지 속속 등장하고 있다.

이러한 응용 분야를 가지는 XML 문서의 검색과 관련된 특징으로 크게 3가지를 들 수 있다. 첫째, 중첩된 태그(tag)를 사용하여 문서를 기술하게 되면 구조를 가

† 비 회 원 : 서울대학교 전기컴퓨터공학부
sjyoo@oopsla.snu.ac.kr
†† 비 회 원 : 서울대학교 인지과학
ksmin@oopsla.snu.ac.kr
††† 종신회원 : 서울대학교 컴퓨터공학부 교수
hjk@oopsla.snu.ac.kr
논문접수 : 2002년 2월 25일
심사완료 : 2002년 11월 2일

진 문서로 표현될 수 있다. 따라서 구조에 대한 질의와 내용에 대한 질의 또는 이 둘을 동시에 질의할 수 있다. 둘째, XML 문서의 구조는 비정규적이다. XML 문서가 여러 곳에서 다른 종류의 문서들을 수집하였을 경우, 또는 같은 종류의 데이터를 수집하였다 하더라도 서로 다른 DTD(document type definition)를 사용한다면 그 구조는 비정규적이라 할 수 있다. 셋째, 이러한 비정규적인 구조를 가지는 문서들에 대하여 일반 사용자들은 구조 질의를 작성하는데 어려움이 있다. 사용자가 구조 질의를 사용하였다 하더라도 비정규성에 기인한 실수할 가능성이 높다고 하겠다.

검색과 관련된 3가지 특징을 잘 반영하기 위해서 다음 3가지 사항이 필요하다. 첫째로 일반 사용자가 비정규적인 구조를 잘 알지 못하기 때문에 구조가 없거나 있더라도 조금만 있는 구조 질의를 작성하게 된다. 따라서 XML 문서의 구조정보를 단순히 관련성 문서 선별에만 이용하게 되면 좋은 검색 결과를 얻기 어렵기 때문에 XML 문서의 구조정보를 이용하여 구조적인 근접도를 이용하여 순위화에 이용해야한다. 둘째로 비정규적인 구조 때문에 사용자가 충분한 구조 정보를 질의에 포함하였을 때 정확한 질의 결과는 아주 제한적이 된다. 따라서 정확한 질의 결과만을 산출하는 것이 아니라 질의와 가장 부합하는 문서부터 순위화하는 것이 사용자에게는 의미 있는 결과를 줄 수 있겠다. 셋째는 일반 사용자들이 비정규적인 특성을 가지는 XML 문서들에 대하여 XPath[2]나 XQuery[3]에 준하는 구조질의를 사용하기 어렵기 때문에 짧고, 쉽고, 그리고 기술하기 용이한 질의 언어가 필요하다.

본 논문에서는 두 가지 가정을 가진다. 첫째는 XML 문서를 트리로 모델링 하였다. XML 문서는 원소(element), 속성(attribute), 링크(link)로 구성되고 이중 링크를 단순한 텍스트로 취급하면 트리로 모델링 가능하다. 비록 우리가 링크 정보를 단순 텍스트 취급하였지만 본 논문에서 제안하는 모델을 단순히 확장하면 링크에 대한 처리가 가능하다. 둘째로 질의는 적어도 내용에 대한 질의는 포함하여야 한다. XML 문서에 대한 질의는 구조에 대한 질의와 내용에 대한 질의 그리고 이 둘을 동시에 사용하는 질의가 가능하다. 하지만 우리는 이 세 가지 유형의 질의를 모두 처리할 수 있지만 본 논문에서 제안하는 모델들은 단순한 처리를 위해 구조만을 위한 질의는 없다고 가정한다.

우리가 제안하는 XML 문서를 위한 질의 모델은 3가지이다. 3가지 모델의 질의 평가 방법은 [4, 5, 6]에서처럼 상향식으로 접근한다. 문서구조에 대한 질의와 내용에 대한 질의 중, 내용에 대한 질의를 먼저 수행하여 검색하여야할 문서의 수를 줄인 후 선택된 각 문서 트리의 단말 노드에 대해서 구조 질의를 수행한다. 구조 질의를 수행할 때 질의 경로와 문서 트리 상의 루트부터 단말 노드까지의 경로의 차이를 편집거리[7]를 이용하여 얼마나 유사한지 계산하는 방식과 이렇게 계산된 단말 노드들의 가중치가 트리 상에서 얼마나 인접한가에 따라 문서의 가중치를 계산하는 모델을 제안한다. 또한 여러 가지 질의 경로를 질의로 사용했을 때 하나 보다는 여러 가지의 질의 경로가 나타난 것이 인접하여 나타나는 정보를 이용하여 가장 좋은 문단 또는 가장 좋은 구조를 가지고 있는 문서를 검색하는 모델을 제안하였다.

본 논문의 기여한 점은 다음과 같다. XML 문서를 트리로 보고 트리 상에서의 근접도를 정의하였고 이러한 근접도를 이용하여 검색하는 모델을 정의하였다. 편집거리를 이용하여 검색의 질의와 문서의 구조 검색의 유사도를 측정하는 방법을 제안하였으며 질의 트리의 종류가 문서 트리에 얼마나 나타나는지에 관한 정보를 이용하는 모델을 제시하였다. 또한 제시한 모델들을 효율적으로 평가할 수 있는 색인 방법도 제시한다. 또한 실험을 통하여 간접적으로 제시한 모델들의 유용성을 증명한다.

논문의 구성은 2절에서 질의 모델을 위해 필요한 문서, 질의에 대한 모델링을 한다. 추가로 구조 문서에서의 근접도에 대해 정의한다. 3절에서는 2절의 모델 및 근접도에 기반 하여 3가지 검색 모델을 제시하고 4절에서 효율적인 구현을 위한 색인 구조를 제시하고 구현방식에 대해 설명한다. 5절에서 실험에 대해 설명하고 6절에서 관련연구에 대해 언급한다. 마지막으로 7절에서 결론을 맺도록 한다.

2. 문서, 질의 그리고 근접도(proximity)

2.1 XML 문서 모델

XML 문서는 구조가 없는 단어들의 나열, 구조가 있는 트리(Tree) 또는 그래프(Graph)로 모델링 한다. 구조가 없는 단어들의 나열로 모델링 하는 것은 기존의 IR 모델을 XML 문서에 대해 그대로 적용하였을 경우이다. 트리로 모델링 하는 경우는 문서내의 링크구조를 무시하고 원소(element) 간의 포함관계를 트리로 그대로 나타낸 경우이며 [2, 3, 6, 8] 등에서 이와 같이 모델링 한다. 그래프는 트리로 나타내어진 구조상에서 링크구조를

분석하여 모델에 추가하게 됨으로써 모델링 될 수 있다. 그래프로 모델링 하는 경우로 [9, 10, 11] 등에서 사용한다.

XML 문서를 트리로 모델링 할 경우 그래프 보다 쉽게 색인을 만들 수 있고 질의를 처리할 수 있는 반면에 XML 문서가 가지고 있는 정보를 모두 모델링 하지 못한 것이 된다. 하지만 질의 언어가 트리로 모델링 된 문서의 링크정보를 분석하여 처리한다면 XML 문서의 모든 정보를 활용 가능하게 된다.

본 논문에서는 XML 문서를 그림 1과 같이 순서를 가진 트리로 모델링 하였다.

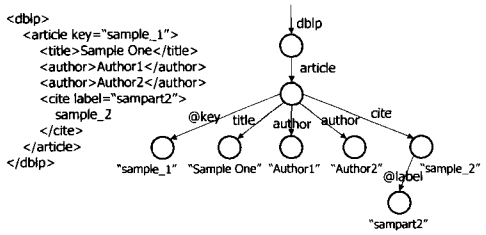

그림 1 XML 문서와 해당 문서 트리의 예

<dblp>와 같은 원소(element)는 간선(edge)에 레이블(label) 된 노드로 나타내었고 속성은 '@'을 앞에 붙인 노드로 모델링 하였다. 따라서 <cite> 노드 자식 노드로 '@label'노드가 생성되었다. PCDATA는 단말 노드의 스트링으로 삽입하였다.

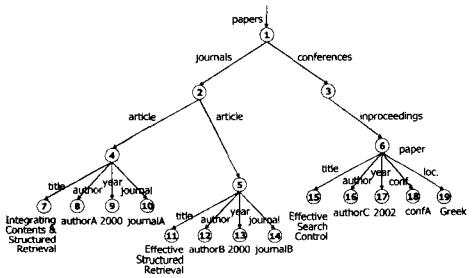

그림 2 XML 문서 예

2.2 XML 질의 모델

트리로 모델링된 문서에 대해서 질의는 트리의 어느 한 노드, 정규 경로들의 집합(Set of Regular Paths, 이후 SRP), 트리로 모델링 가능하다. 내용에 대한 질의만 사용하게 되면 각 질의 단어는 트리의 어느 한 노드에 매핑이 될 것이다. 이와는 다르게 "제목에 contents 라는 단어를 포함하는 문서를 찾아라"는 질의는 XPath[1]로 //title='contents' 나타낼 수 있다. 그림 2에 대하여 이 질의를 수행하면 7번째 노드가 조건을 만족하므로 검색될 것이다. 이전 질의와의 차이점은 내용만이 아니라 구조에 대한 질의가 함께 담겨져 있다는 것이다. 이러한 질의는 정규 경로로 모델링 될 수 있다. 질의가 //title='contents' //journal='journalA'인 경우는 그림 3 같은 SRP로 모델링 될 수 있다. 하지만 "제목에 'contents'라는 단어를 포함하고 journal이 'journalA'인 것과 관련 있는 논문를 찾아라"는 질의는 //paper[/journal='journalA']/title= 'contents'로 표현가능 하고 그림 4와 같이 트리로 나타낼 수 있다. 그림 3의 SRP질의와 차이점은 경로간에 조인이 발생한다는데 있다.

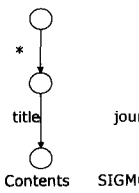

그림 3 SRP 질의 예

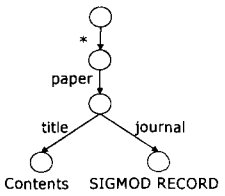

그림 4 질의 트리 예

트리로 모델링된 질의의 경우, 트리는 루트부터 단말 노드까지의 경로의 집합으로 볼 수 있고 이 경로간에 조인이 발생하지 않는다면 SRP 질의로 볼 수 있다.

기존 질의 모델은 질의 표현 능력 면에서 뛰어나다. 하지만, 이는 일반 사용자들이 사용하기에 복잡하고 구조를 잘 알아야 한다는 문제점이 있다. 이에 우리는 기존의 질의 언어 모델과 달리 질의를 SRP로 모델링 한다. SRP로 질의하는 경우는 (1)질의를 하는 문서 집합의 구조에 대해서 잘 알지 못하여 트리 수준의 질의를 기술할 수 없는 경우이거나 (2)사용자가 질의를 잘 구성하지 못하는 경우이다. 이런 경우 SRP 질의는 트리 형태의 질의보다 상대적으로 기술하기 용이하므로 유용하다. 또한 (3) SRP 질의는 기존 검색 모델에서 확장이 용이하다. 반면에 질의 경로간에 독립을 가정하므로 사용자의 질의 경로간에 연관성을 그림 3에서 알 수 있듯이 기술하지 못한다. 이는 질의 표현 능력이 떨어진다는 것을 의미한다. 하지만 본 논문 3.3절에서 제시하는 모델을 사용하면 트리 형태의 질의를 사용하는 것과 유사한 효과를 얻을 수 있다.

2.3 근접도

근접도를 이용하여 검색하면 검색성능을 높일 수 있다는 것은 직관적으로 알 수 있다. [12, 13, 14, 15, 16, 17, 18]

는 근접도를 이용하기 위하여 근접 연산자를 정의하거나, 근접도를 문서의 가중치에 반영한 연구들이다. XML 문서는 논리적인 단위가 태그로 구분되어 있다. 이러한 논리적인 단위를 이용하여 검색하게 되면 논리적 단위 내에 근접하여 가장 많이 나타나는 노드들이 검색되므로 검색의 성능을 높일 수 있을 것이다. 이와 관련된 기존의 연구로 Passage Retrieval [12, 15, 16, 17, 18, 19, 20, 21]이 있다. Passage Retrieval은 문서를 일정한 단어 수 크기 또는 이미 구분된 논리적인 단위들을 하나의 passage로 간주한다. 검색은 각 passage 대한 검색을 수행하거나 검색된 passage들의 가중치 정보를 이용하여 한 문서의 가중치에 반영하는 방법을 사용하고 있다. [15, 21]의 연구에서는 어느 정도의 효과를 보았지만 [12, 16, 17]들에서 논리적 단위를 사용하는 Passage retrieval은 오히려 검색 성능이 떨어지는 결과를 얻었다. 이들 연구들의 실패 원인은 가장 좋은 passage 하나만 선택하거나 또는 검색된 passage들 간의 구조정보를 활용하지 않고 단순히 가중치들을 합하거나 조합하는 방법을 사용한다. 하지만 XML 문서의 구조 정보상의 의미적 관계(부모자식, 조상자손, sibling)가 있고 이를 이용하여 근접도를 정의할 수 있다. 이러한 구조상의 근접도 정보를 이용하여 passage들의 가중치를 합하는데 이용한다면 passage들의 가중치가 근접도에 따라 한 문서의 가중치로 합산될 수 있을 것이다.

근접도를 사용하려면 거리의 개념이 필요하다. 구조를 가진 문서에서 단말 노드의 텍스트 내에서의 단어 거리와 노드 상의 거리가 정의 가능하다. 노드간에 다시 수평 거리와 수직 거리가 정의 가능하다.

수평거리는 XML 데이터를 문서의 관점에서 바라보았을 때 순서를 가진 문서로 볼 수 있다. 이렇게 순서를 가진 문서는 나열되는 순서가 의미를 가지는데 예를 들어 첫 번째 문단과 두 번째 문단이 인접해 있다면 이 두 문단은 비교적 논리적 개연성을 가진다고 볼 수 있고 이들에 대해 근접도를 구하고 근접도에 따른 가중치를 주는 것이 의미가 있다고 할 수 있겠다.

수직거리는 논리적 단위들을 그룹화 하여 이들을 합친 하나의 단위로 취급할 수 있는데 이러한 논리적 단위들의 그룹화의 정도를 수직거리라 생각할 수 있다. 예를 들어 문단들의 합친 단위를 절이라 생각할 수 있고 절들을 합친 것을 장이라 생각할 수 있을 때 문단과 장 간의 수직 거리는 2라고 생각할 수 있다. XML 데이터를 문서의 관점에서 바로 보지 않고 데이터의 관점에서 바라볼 경우 수평거리는 무의미하고 수직거리의 의미는 매우 중요하다고 할 수 있겠다. RDB에서 추출한 데이터를 XML문서로 변환하였을 경우 한 튜플의 속성간의 순서는 무의미할 것이다. 따라서 검색엔진이 XML을 데이터관점과 문서의 관점에서 동시에 바라본다고 하면 이 둘을 동시에 고려하여야 한다.

정의 1 : 거리(Distance)

$$T\text{-}distance(term_{ij}, term_{ik}) = |k-j| \text{ if } Node(term_{ij}) = Node(term_{ik})$$

$$V\text{-}distance(N_{ik}, N_{ij}) = |level(N_{ik}) - level(N_{ij})| \text{ if } N_{ik} \text{ is a descendant or ancestor of } N_{ij}$$

$$H\text{-}distance(N_{ij}, N_{ik}) = |k-j| \text{ if } Parent(N_{ij}) = Parent(N_{ik})$$

$term_{ij}$: 문서 i에서 j번째 단어

N_{ik} : 문서 i에서 k번째 BFS(Breadth First Search) 번호를 가지는 노드

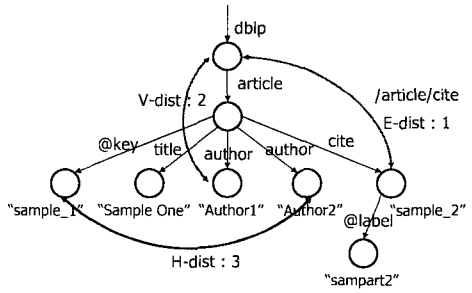

그림 5 거리 예

예를 들어, 그림 5에서 /dblp/article/title에서 Sample과 One의 단말 노드 내에서 단어 거리(word distance)인 T-distance($Sample_{i6}$, One_{i7})는 1이다. 기존의 문서들에서 단어 사이의 거리를 구한 것과 기본적으로 같으나 XML 문서트리의 단말 노드 내에서 정의된 것이 다르다. 수평 거리를 의미하는 H-distance는 같은 부모 노드를 가지는 자식 노드간의 BFS 번호 차이 값이다. 따라서 /dblp/article/@key와 두 번째 /dblp/article/author의 H-distance(N_{i3}, N_{i6})는 3이다. 마지막으로 수직 거리를 의미하는 V-distance는 같은 경로 상에 있는 노드간의 레벨 차이 값이며 /dblp와 /dblp/article/title의 V-distance(N_{i1}, N_{i5})는 2이다. V와 H-distance는 XML 문서를 트리로 가정함에 따라 새로이 생겨난 거리 개념이다.

거리에 따른 근접도는 얼마나 서로 다른 두 노드가 인접해 있는가 의미한다. 거리와 마찬가지로 수평과 수

직 근접도의 정의가 가능하고 우선 수평 근접도의 정의는 다음과 같다.

정의 2 : H-proximity(수평 근접도)

$$H-proximity(N_{ij}, N_{ik}) = \{f(v+(1-v)C)\}^D$$
$$D : H-distance(N_{ij}, N_{ik})$$
$$C : \text{level factor}$$
$$f, v : 0 \le f, v \le 1$$
$$C = \frac{level(N_{ik})}{\max V(i)}$$
$$\max V(i) : \text{문서 i의 최대 깊이}$$

정의 2에서 C는 트리의 레벨에 따라 H-distance의 의미를 다르게 부여하기 위해서 사용된다. v는 C의 반영 정도를 나타내기 위한 매개변수이며 f는 H-distance에 따른 감소 요소(decreasing factor)이다. f^D으로 H-distance에 따른 영향을 크게 준 이유는 거리가 멀어 짐에 따라 근접도의 의미가 지수적으로 감소한다고 가정했기 때문이다. 또한 실험을 통하여 f와 D의 관계를 합, 곱 등 여러 가지로 실험하여 보았지만 지수적으로 감소하는 함수가 가장 좋은 결과를 산출했다.

정의 3 : V-proximity(수직 근접도)

정의 3.1 단순 V-proximity

$$V-proximity(N_{ij}, N_{ik}) = t^V$$

정의 3.2 정규화된 V-proximity

$$V-proximity(N_{ij}, N_{ik}) = \frac{level(N_{ij})}{level(N_{ik})}$$
$$t : V-proximity\ factor (0 < t \le 1)$$
$$V : V-distance(N_{ij}, N_{ik})$$

정의 3.1은 V-distance는 단순히 상수배 하여 V-proximity를 적용하는 방법이고 정의 3.2는 각 문서의 레벨로 정규화 시킨 방법이다. 정의 3.1을 사용한 V-proximity의 효과는 루트로부터 멀리 떨어진 결과에 대해서는 낮은 가중치를 주고 루트로부터 가까운 결과 노드에 대해서는 보다 높은 가중치를 주는 효과가 있다. 정의 3.2도 같은 문서에 대해서 같은 효과를 나타내지만 다른 문서들과의 깊이 차이에 대한 정규화가 가미된 방법이다. 정의 2와 같이 정의 3에도 레벨에 따른 효과를 다르게 줄 수도 있다.

추가로, 근접도를 기반으로 하여, 두 노드의 가중치를 정의할 수 있다. 다음은 수평과 수직 가중치에 대한 정의이다.

정의 4 : V-weight(수직 가중치)

$$V-weight: W_{ij} = W_{ij} + V-proximity(N_{ij}, N_{ik}) \cdot W_{ik} \text{ if } W_{ij} \text{ is an ancestor of } W_{ik}$$
$$W_{ij} : N_{ij}\text{의 가중치}$$

정의 5 : H-weight(수평 가중치)

$$H-weight: W_{iq} = \max(W_{ij}, W_{ik}) + H-proximity(N_{ij}, N_{ik}) \cdot \min(W_{ij}, W_{ik}),$$
$$\text{where } q = \frac{j * W_{ij}}{(W_{ij}+W_{ik})} + \frac{k * W_{ik}}{(W_{ij}+W_{ik})}$$

지금까지, 문서 내의 거리와 거리에 따른 근접도와 가중치를 정의하였다. 추가로, 질의와 문서와의 차이가 발생할 경우에 대한 개념 정립이 필요하다. XML 문서 트리와 정규 질의 경로와의 정확한 매치가 발생하지 않은 경우 본 논문에서는 편집 거리(edit distance)[7]를 사용하여 질의와 문서간의 매치를 시도한다. 편집 거리란 질의와 데이터간에 정확한 매치가 발생하지 않았을 경우 최소 비용 편집 연산(삽입, 삭제, 변경)을 통하여 질의와 문서 트리간의 매치를 발생하게 하는 최소 비용을 말한다. 편집 연산은 원소(element) 단위로 이루어진다. 그림 5에서 질의 경로가 /article/cite이고 문서에 /dblp/article/cite가 존재한다고 하자. 편집 비용 중 삽입 비용이 1이라고 하면 이 경우의 편집 거리는 1이 된다.

3. 검색 모델

이 절에서는 앞서 정의한 XML 문서, 질의, 근접도, 편집거리를 기반으로 세 종류의 모델을 제안한다. 제안하는 모델의 기본 접근 방향은 먼저 질의에 사용된 내용에 관한 질의들을 수행 하여 검색되어야 할 문서의 범위를 줄인 후, 검색된 문서의 트리 상에서의 노드들에 대해 구조 질의를 수행한다. 이는 [4, 5, 6]에서 접근한 방법과 같은 상향식 접근법이며 [23]에서 V-색인과 L-색인을 사용하여 접근하는 방법과도 유사하다. 이후 문서에 대해 가중치를 매기는 방법은 모델별로 상이하다.

3.1 ED 모델

만약 어떤 사용자가 Search control과 관련된 논문을 찾기 위해서 //article/title = 'Search Control'을 찾는 질의를 그림 2에 수행하면 정확한 매치가 없기 때문에 이 문서에서는 결과를 찾을 수 없다. /conferences/inproceedings/title = 'Search Control'과 같이 정확히 질의를 하기 위해서는 질의 대상이 되는 문서집합의 정확한 구조를 알고 있을 때 가능하다. 하지만 사용자가 방대한 양의 XML문서의 구조를 모두 알고 있기는 어렵고, 설사 알고 있다고 하더라도 article과 inproceedings처럼 충분히 실수 할 수 있으며, 다양한 데이터가 합쳐져 있을 경우에는 정확한 질의를 위해서 복잡한 질의를 던져야 한다. 이러한 문제를 해결하기 위해서는 사용자 구조 질의와 문서 구조간의 차이에 대해서 적절한 비용을 지불하고 찾아주는 모델이 필요하다.

ED 모델은 SRP 질의 모델에서 경로의 정확한 매칭만을 찾는 것이 아니라 편집 거리(edit distance)를 고려하여 질의와 데이터간의 차이에 대해서 편집 거리에 따라 비용을 지불하고 검색하는 모델이다.

질의 수행 방법은 각 질의 경로마다 내용에 대한 질의를 수행한다. 이때 가중치로 벡터 모델을 사용한다면 TF*IDF 가중치를 사용할 수 있다. 이 가중치에 대해 구조 질의를 수행하여 구조 질의와 문서 구조간의 차이에 대하여 편집 거리를 구한다. 편집 거리에 따른 감소 함수를 이용하여 TF*IDF 가중치를 감소시키면 해당 질의 경로에 대한 단말 노드의 가중치($W_{ik,j}$)가 된다.

$$W_{ik,j} = idf_{ij} \cdot tf_{N_{ik},j}\, e^{d_{ik,j}}$$

idf_{ij} : 문서 i에서 j번째 질의 경로의 질의 단어의 idf

$tf_{N_{ik},j}$: j번째 질의 경로의 단말 노드 N_{ik}에서 출현 빈도 수

$d_{ik,j}$: j번째 질의 경로와 루트부터 단말 노드 N_{ik}까지의 편집거리

e : 편집 거리 factor (단 $0 \le e \le 1$)

$W_{ik,j}$: 단말 노드 N_{ik}에서 j번째 질의 경로의 가중치

편집 거리에 따른 감소 함수는 편집 거리가 증가함에 따라 지수적으로 감소하는 함수를 이용하였고 단말 노드의 가중치를 위해서 다른 모델 [24, 25]을 사용해도 무방하다. $W_{ik,j}$는 각 질의 경로에 대한 가중치이므로 단말노드의 가중치(W_{ik})는 $W_{ik} = \sum_{j=1}^{|SRP|} W_{ik,j}$이고 이에 따른 각 문서별 가중치는 $W_i = \sum_k W_{ik}$이다.

e가 1이면 단순히 단말 노드에 같은 질의 용어가 나타나는 모든 노드들의 가중치의 합이되며 질의에 사용된 단어가 경로에 나타나지 않는다면 기존 검색모델과 같은 결과를 산출하게 된다. e가 0에 가까워지면 질의와 정확한 매치가 발생한 경우만 고려하게 된다. e가 0과 1 사이의 값이면 기존 검색모델과 정확한 매치간의 중간의 결과를 얻게 된다.

예를 들어 그림 2에 //article/author='author3' //article/year= '2000' 와 같은 질의를 e=0.5이고 각 단어의 idf가 1이라고 하면 9, 12, 13번 노드의 가중치는 1이고 16번 노드는 경로가 하나 틀렸으므로 0.5의 가중치를 가지게 되어서 총 3.5의 가중치를 얻게 된다. e가 1이면 4의 가중치를 가지게 되고 0이면 3이 된다.

문서 길이에 따른 정규화 방법은 각 단말 노드 별 정규화 하는 방법과 본 연구에서 벡터 모델을 사용했으므로 벡터 모델의 문서 길이 정규화 방법이 사용가능 하다.

3.2 PE 모델

만약 어떤 사용자가 "제목이 contents를 포함하고 1996년에 출간된 논문이나 책 등 관련 있는 문서를 찾아라"라는 질의는 //title=contents //year=1996으로 간단히 질의로 표현할 수 있다. 이런 질의일 경우 title과 year에 대해서 매치된 노드들은 같은 부모노드를 가질 경우는 중요한 의미를 가질 것이고 트리 상에서 멀리 떨어져 있다면 낮은 의미를 가질 것이다.

이에, 우리는 ED 모델의 가중치 방법과 함께, V-proximity 와 H-proximity를 이용하여 트리 내에서의 상대적인 근접도를 계산하고 문서의 가중치를 구하는 방법인 PE 모델을 제안하였다.

이 모델은 ED 모델에서 단말 노드들의 가중치를 이용하여 근접도에 따른 문서의 가중치를 계산한다.

다음은 PE 모델의 알고리즘이다.

Algorithm 1 Evaluation algorithm for PE model

```
1: Input: i : doc number, result leaf nodes from ED
   model in doc. i
2: Output: weight of document i
3:
4: insert result leaf nodes into heap
5: level ← get depth of document i
6: repeat
7:    start_bfs ← get_start_bfs(i, level)
8:    end_bfs ← get_end_bfs(i, level)
9:    node ← remove max bfs node from heap
10:   while start_bfs ≤ node ≤ end_bfs and heap ≠∅
      do
```

```
11:    if while loop is the first time then
12:      last_node ← node
13:    else if last_node is sibling with node then
14:          calculate H-distance, H-proximity and
           H-weight
15:        adjust weight and bfs of the last_node with
           H-weight
16:    else
17:      to_parent(heap, last_node)
18:      last_node ← node
19:    end if
20:    node ← remove max bfs node from heap
21:  end while
22:  if while loop is performed then
23:    to_parent(heap, last_node)
24:  else
25:    to_parent(heap, node)
26:  end if
27:  level ← level - 1
28: until level is 1
29: return weight of the node in the heap
30:
31: procedure to_parent(heap, node)
32: calculate V-weight with 1 H-distance
33: adjust weight and bfs of the node with V-weight
34: insert node into heap
35: end procedure
```

알고리즘 1에서는 단말노드 레벨부터 루트노드까지 각 레벨별 형제 노드가 존재하는 노드들은 H-distance에 따른 H-proximity를 계산하고 H-proximity를 이용하여 두 노드를 합친 노드의 가중치와 BFS 번호를 계산한다. 만약 더 이상 형제 노드가 존재하지 않는 노드들에 대해서는 그 노드의 가중치를 부모 노드로 보낸다. 다음, V-distance '1'에 따른 V-proximity를 계산하고 이에 따라 V-weight를 계산한다. 계산된 가중치는 V-weight의 정의에 따라 부모노드의 가중치가 된다. 부모노드의 계산 결과는 다시 힙에 저장된다. V-proximity와 H-proximity에 따른 가중치가 계속적으로 합산이 되어서 루트노드에 이르게 되면 루트노드의 가중치가 해당 문서의 가중치가 된다.

PE 모델 결과 가중치는 모든 결과 노드가 인접하여 있다면 벡터 모델의 가중치와 같게 된다. 하지만 검색 결과 노드들이 근접하지 않을수록 낮은 가중치를 갖게 된다. 이는 검색 결과 노드들이 인접하여 있을 때 높은 가중치를 주어 근접도를 검색 결과에 반영하게 된다.

$f=1$, $v=0$이고 정의 2와 정의 3.2를 사용하여 알고리즘 1을 ED모델의 예제에 똑같이 수행할 경우 단말 노드의 가중치는 변함이 없다. 5번 노드의 가중치는 1.75이고 2번 노드는 2.25 1번 노드는 0.2375의 가중치를 가지게 된다. 3,6번 노드의 가중치는 단말 노드의 가중치와 같다.

3.3 PH 모델

만약 사용자가 //article/author= 'authorB' //article/year= '2000' 과 같은 질의를 사용자가 요청하였을 때 //article/author='authorB'나 //article/year= '2000' 둘 중 하나만 나타난 문서와 두 가지의 질의 경로가 모두 나타난 문서가 있다면 두 가지 질의 경로가 모두 나타난 문서가 보다 더 의미가 있을 것이다. 두 가지 질의 경로가 모두 나타난 문서들 중에서도 XPath 질의인 //article[/author='authorB']/year= '2000' 을 만족하는 문서가 있다면 보다 더 의미 있는 문서로 볼 수 있다.

PH 모델은 SRP 질의 수행 결과가 문서 트리에서 질의 정규 경로의 종류의 수가 많으면 많을수록 그리고 각 경로가 문서 트리에 질의와 보다 정확히 매치하면서 다른 경로와 인접하면 할수록 보다 높은 의미를 가지는 요소를 이질성(Heterogeneity)라 정의하자. PH 모델은 SRP 질의의 이질성을 PE 모델의 결과에 반영하는 모델이다.

다음은 임의의 단말 노드 N_{ik}의 j번째 질의에 대한 이질성 $H^T_{ik,j}$, 임의의 노드 N_{ik}의 이질성 벡터 H^T_{ik} 그리고 임의의 노드 N_{ik}의 이질성 H_{ik}는 다음과 같이 정의된다.

정의 6 : $H^T_{ik,j}$, H^T_{ik}, H_{ik}

$$H^T_{ik,j} = e'^{\,H^S_{ik,j}}$$
$$H^S_{ik,j} = level(N_{ik})$$
$$H^D_{ik,j} = d_{ik,j}$$

N_{ik} : 단말 노드

e' : 이질성을 위한 편집거리 factor (단 $0 \le e' \le 1$)

$$H_{ik} = \sum_{j=1}^{|SRP|} H^T_{ik,j}$$

$$H^T_{ik} = \begin{pmatrix} H^T_{ik,1} & H^T_{ik,2} & \cdots & H^T_{ik,|SRP|} \\ H^S_{ik,1} & H^S_{ik,2} & \cdots & H^S_{ik,|SRP|} \\ H^D_{ik,1} & H^D_{ik,2} & \cdots & H^D_{ik,|SRP|} \end{pmatrix}$$

$H^S_{ik,j}$는 단말노드의 시작 레벨을 기억하여 서로 다른 시작레벨에 다른 가중치를 부여하기 위해 사용되어진다.

예를 들어 /article/author와 /article/ref/book/author를 서로 다르게 취급하기 위해서는 $H^S_{ik,j}$가 사용되어진다. $H^D_{ik,j}$는 N_{ik}의 j번째 질의 경로의 편집 거리를 저장한다. 만약 j번째 질의 경로가 정확히 정합 되었다면 $H^T_{ik,j}$는 1을 가질 것이고 그렇지 않다면 정의 6의 식에 따라 1보다 작은 값을 가지게 될 것이다.

중간 노드 N_{ik}와 임의의 자식 노드 N_{iq}와의 이질성 합 $H^T_{ik} = H^T_{ik} \oplus H^T_{iq}$은 다음과 같다.

정의 7 : $H^T_{ik} = H^T_{ik} \oplus H^T_{iq}$

각 j에 대하여

$$\left.\begin{array}{l} H^T_{ik,j} = e^{\prime\, H^D_{iq,j}} \cdot \left(\dfrac{level(N_{ik})}{H^S_{ik,j}}\right)^h \\ H^S_{ik,j} H^S_{iq,j} \\ H^D_{ik,j} H^D_{iq,j} \end{array}\right\} \quad \text{if } H^T_{ik,j} < e^{\prime\, H^D_{iq,j}} \cdot \left(\frac{level(N_{ik})}{H^S_{iq,j}}\right)^h$$

$$\left.\begin{array}{l} H^T_{ik,j} = H^T_{ik,j} \\ H^S_{ik,j} = H^S_{ik,j} \\ H^D_{ik,j} = H^D_{ik,j} \end{array}\right\} \text{otherwise}$$

N_{ik} : 임의의 한 자식 노드의 이질성이 할당된 중간 노드

N_{iq} : N_{ik}의 자식 노드

Algorithm 2 Compute the heterogeneity of an internal node N_{ii}

1: Input: i : $H^T_{ij}, H^T_{ik}, \cdots, H^T_{in}$: the heterogeneity of child nodes of N_{ii}

2: Output: H^T_{ii} : the heterogeneity of N_{ii}

3:

4: $H^T_{ii} = H^T_{ik}$ where k is m of

$$H_{im} = MAX\{H_{ij}, H_{ik}, \cdots, H_{in}\}$$

5: **for** each $H^T_{iq} \in \{H^T_{ij}, H^T_{ik}, \cdots, H^T_{in}\}$ do

6: $\quad H^T_{ii} = H^T_{ii} \oplus H^T_{iq}$

7: **end for**

정의 7의 h가 0이면 단지 유일한 질의 경로의 개수만 고려하게 되고 h가 1이거나 1보다 크면 *V-distance*가 이질성에 영향을 미칠 것이다. 알고리즘 2는 내부 노드의 이질성을 계산한다. PE 모델과 마찬가지로 이질성은 단말 노드부터 루트 노드로 계산해 나간다. 이질성에 따른 PH 모델의 가중치는 다음과 같다.

$$W_i^{PH} = \frac{|SRP| + k \cdot H_{i1}}{|SRP|} W_i^{PE}$$

W_i^{PH}는 k가 0이면 PE모델과 같은 가중치를 가진다. 가중치에 따른 순위화 방법은 W_i^{PH}를 그대로 사용하거나 이질성 H_{i1}에 따라서 먼저 순위화 한뒤 W_i^{PH}에 따라 순위화 할 수 있다. 이중 두 번째 방법을 메달 순위화(medal rank)라 한다.

정리 3.1 임의의 한 XPath 질의 트리를 SRP 질의로 변환하였을 때 다음 세가지 조건을 만족한다면 항상 보다 정확한 정합이 보다 부정확한 정합 보다 높은 이질성 값을 가진다.

1. 임의의 문서 트리의 루트부터 단말까지 중복된 태그가 나타나지 않는다.
2. $e' < \left(\dfrac{1}{\max V(i)}\right)^h$ and $h \geq 1$
3. $e' < \dfrac{1}{|SRP|}$

증명 : 생략 ■

조건 1은 문서 트리에서 태그이름이 일치하였다면 그 위치는 항상 맞다는 것을 보장하여 준다. 조건 2는 얻을 수 있는 가장 작은 이질성 값이 임의의 한 태그와 정합이 발생하지 않았을 때보다 크다는 것을 보장해 준다. 조건 3은 부정확한 경로들의 합이 하나의 정확한 경로의 이질성 값보다 작다는 것을 보장하여 준다.

정리 3.1에 의하면 이질성에 따른 순위화를 하게 되면 XPath질의의 대부분을 수용할 수 있다는 것을 알 수 있다. 따라서 SRP로 질의를 사용하더라도 어느 정도의 XPath의 표현능력을 가진다는 것을 보장할 수 있다.

이질성 값이 크면 클수록 독립적인 정규 경로들이 인접하여 정확하게 정합이 발생했다는 것을 의미하고 작으면 그 반대이다. 이질성 값을 구조정보가 없는 문서에 대해 적용하면 각 단말 노드에 대한 이질성 값은 가장 좋은 단말 노드 또는 문단을 알려주는 것이 되고 문서에 대해서는 가장 좋은 단말 노드들의 집합 또는 가장 좋은 문단들의 집합이 어느 정도 인지 알려주는 기준이 된다. 구조 정보가 있는 문서에 대해서 이질성 값은 가장 좋은 매칭을 가지고 있는 문서에 대해 알려준다. PE 모델의 가중치만을 사용할 경우 어느 한 질의 경로만

많이 나타나는 문서도 높은 가중치를 가지지만 PH 모델의 이질성을 같이 고려하면 인접하여 나타나면서 이질성의 값이 높은 문서가 높은 가중치를 가지게 되어 보다 관련성이 높은 문서를 높은 가중치를 가지게 한다.

//article/author='authorB' //article/year= '2000' 질의를 $e'=0.3$, $k=1$, $h=1$ 값을 가지고 이질성을 계산하면 1,2,5노드는 1.75의 이질성을 가지고 나머지 노드의 이질성은 단말 노드의 이질성과 같다. W_i^{PH} = 2.375 * 3.75 / 2 = 4.45이다.

3.4 모델간의 비교

e값이 1이고 벡터 모델의 문서길이 정규화 기법을 적용하면 벡터 모델이 된다. e값이 0이면 정확한 매칭만을 고려하는 검색모델이 된다. PE모델은 ED 모델의 결과 단말 노드들의 가중치를 근접도에 따라 문서의 가중치를 낮게 만드는 모델이다. 모든 결과 노드들이 정확히 매칭되면서 하나의 단말 노드에 나타난 다면 항상 ED 모델의 결과와 같다. PH 모델은 PE 모델 계산 방식에 H-proximity와 중복해서 나타난 정규 질의 경로를 무시하고 각 질의 경로마다 1의 가중치를 갖는다고 가정하고 계산하면 PH모델의 이질성 값을 얻을 수 있다.

4. 구 현

4.1 인덱스 구조

내용에 대한 색인으로서 트리의 단말 노드의 단어들에 대한 역색인을 사용한다. 이는 기존의 정보검색 시스템에서 사용하는 것과 동일하다.

구조에 대한 색인으로서 두 가지 색인 구조를 사용한다. 루트부터 단말 노드까지의 경로 정보를 얻기 위한 역색인인 절대 경로 역색인과 XML 문서 트리에서 위치를 빨리 찾기 위한 k차 완전 트리 색인[25] 사용한다.

절대 경로 역색인을 이용함으로써 단말 노드부터 루트 노드까지의 경로를 알아내기 위해서 조인이 발생하는 것을 막고 쉽게 편집 거리를 구할 수 있게 한다. 특히 절대 경로 역색인을 RDB에 저장할 경우에도 전용색인 구조를 사용할 경우에 비해서 성능이 크게 뒤떨어지지 않는다[26]. 하지만 절대 경로 인덱스는 루트부터 단말 노드까지의 모든 경로 정보를 가지고 있기 때문에 저장할 양이 많다는 부담과 트리 구조상에서 어느 위치에 해당하는가에 대한 정보가 없다. 우선 저장할 양이 많아질 것이라는 부담은 XML 문서를 트리 형태로 보았기 때문에 문서별로 트리의 단말 노드의 개수만큼 이상 증가하지 않는다. 그리고 실제로 저장된 XML 문서들을 살펴보면 같은 그룹의 문서들은 유사한 문서구조를 가지기 때문에 원소(element) 단위로 저장하는 것보다 색인 저장 비용이 절감된다. 두 번째로 위상에 대한 정보의 결여 문제를 해결하기 위해서 [4,6]에서 사용된 k차 완전 트리 색인[26]을 사용한다. k차 완전 트리 색인은 XML 문서를 k차 완전 트리로 보고 루트부터 단말 노드까지 BFS(Breadth First Search) 번호를 각 노드에 할당한다. k차 완전 트리이므로 어느 한 노드의 부모와 j번째 자식 노드를 수식 계산에 의해 쉽게 계산 될 수 있다. k차 완전 트리 색인만을 사용하게 되면 트리 상에서의 위상 정보를 쉽게 구할 수 있지만 해당 노드와 관련된 태그의 이름은 알아낼 수 없다. 따라서 색인을 찾아야 하고 새로운 노드를 찾을 때마다 같은 작업을 반복해야 한다. 하지만 절대 경로 역색인과 같이 사용하게 되면 루트부터 단말 노드까지의 경로 정보를 한번에 얻을 수 있고 트리 상에서의 위상 정보도 한번에 얻을 수 있기 때문에 추가적인 계산을 통해서 나머지 모든 정보를 얻을 수 있다.

4.2 구현

단어 역색인, 절대 경로 색인, k차 완전 트리 색인은 RDB 위에 구현되었다. 색인 정보를 RDB 위에 구현함으로써 얻을 수 있는 이점은 안정된 RDB의 기술의 이용 가능성(병행제어, 회복, ...), XML질의와 기존 RDB의 데이터에 대한 통합질의, 기존의 RDB 재활용, 검색 시스템의 호환성, 유연성 및 확장성 증가, 정보 검색 시스템 구현의 용이성 등이 있다. RDB를 사용함으로써 발생 가능한 문제점으로 정보 검색 시스템의 성능 저하를 생각해 볼 수 있다. 그러나 [27,28]에서 실험한 결과와 같이 크게 차이 나지 않고 특히 절대 경로 역색인을 이용한 [27] 실험에서는 B+트리 색인을 직접 운용한 것과 성능 차이가 별로 없었다.

그림 6의 term 테이블은 term에 대한 기본적인 정보를 가지고 있다. term 문자열과 각 term별로 유일하게 식별해 주는 termid그리고 idf(inverted document frequency) 정보를 가지고 있다. termindex 테이블은 특정 term이 나타나는 모든 위치에 관한 정보를 가지고 있으며 그에 대한 경로정보(pathid)와 구조정보(eid)를 가지고 있다. 여기서 eid는 k 완전 트리의 BFS 번호이다. path 테이블은 각 경로를 유일하게 식별하는 정보와 절대 경로를 저장하고 있고 각 경로의 idf값도 가지고 있다. pathindex 테이블은 각 문서별 절대 경로의 시작 위치와 끝 위치 그리고 eid정보를 가지고 있다. termindex 테이블에 있는 eid정보는 없어도 된다. 만약 termindex 테이블에서 eid를 제거하면 20%가량의 저장

공간이 줄어들게 된다. 하지만 질의 처리 시에 term index 테이블과 pathindex테이블간에 조인이 발생하여야 한다. docs 테이블은 해당 문서의 id와 실제 파일 위치 그리고 해당 문서의 최대 깊이(maxV)와 최대 자식 노드수(maxH) 정보를 가지고 있다.

term(term, termid, idf)
termindex(termid, docid, pathid, position, eid)
path(pathid, path, idf)
pathindex(pathid, docid, start_position, end_position, eid)
docs(docid, doc_path, maxV, maxH)

그림 6 RDB 테이블 스키마

5. 실 험

5.1 질의 집합

실험을 통하여 PE 모델의 유용성과 그에 따른 PH 모델의 정확성을 증명하고자 한다. 우선 PH 모델의 유용성을 보이기 위해서는 다양한 종류의 대량의 XML 문서가 있어야 하고 이 문서들에 대한 질의 집합과 관련 문서집합이 필요하다. 하지만 XML 문서와 관련된 테스트 컬렉션이 없는 상황에서 직접적인 유용성을 보이기는 어렵다. 또한 PE 모델의 유용성을 보이기 위해서도 구조가 충분한 XML 문서 컬렉션이 필요하지만 현재 구할 수 없는 상황이다. 따라서 XML 문서를 위한 테스트 컬렉션을 활용한 직접적인 성능비교는 어려운 상황이므로 TREC의 TIPSTER 3의 SGML로 구성된 특허 문서를 이용하여 구조 질의 없이 내용 질의만으로 유용함을 증명하는 간접적 방법을 [4]과 같이 선택하게 되었다. 간접적인 방법을 통해 PE모델과 PH모델의 근접도에 따른 유용성을 증명하도록 하겠다. 특허 문서를 선택한 이유는 [4,6]에서와 같이 TIPSTER 문서 집합에서 가장 복잡한 DTD 구조를 가지고 있기 때문이다.

[4]에서 처럼 TREC의 51번부터 150번까지의 토픽 중 특허문서와 관련성을 가지고 있는 총 토픽이 20개였고 관련문서의 개수는 275개이었다. 725개의 문서를 나머지 특허 문서에서 무작위 추출하여 1000개의 문서집합을 생성하였다(약 51MB).

5.2 질의 결과

그림 7의 기준이 되는 벡터 모델은 단순한 TF*IDF를 사용하여 cosine 정규화를 한 것과 정규화된 tf $\left\{0.5+0.5\frac{tf_{N_{doc}}}{tf_{\max}}\right\}$와 $\frac{W_i}{\sqrt{\text{단어개수}}}$를 사용한 2가지 방법을 적용하였다. PE모델은 정의 2와 3.2를 사용하여 계산하였고 PH 모델은 W_i^{PH}가중치를 이용하여 계산하였다. 그림 8은 PH 모델 상에서 k 값의 변화를 정규화 방법간에 비교한 것이다. 그림 9는 정의 3.1의 t값의 변화에 따른 평균 정확률 변화 그래프이고 그림 10은 정의 2의 각 f값에 따른 v 값의 변화의 평균 정확률 변화를 나타내는 그래프이다.

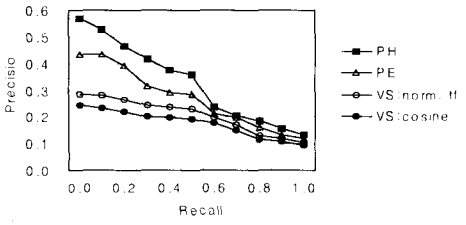

그림 7 모델별 정확률-재현율 그래프

표 3 모델별 평균 정확률

모 델	평균 정확률
PH	0.2989
PE	0.2413
VS:정규화된 tf	0.1927
VS:cosine정규화	0.1788

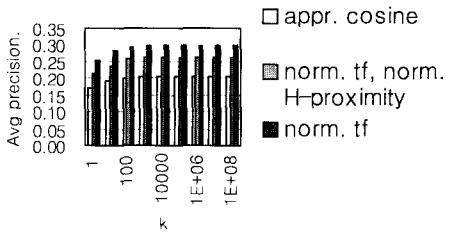

그림 8 k변화 그래프

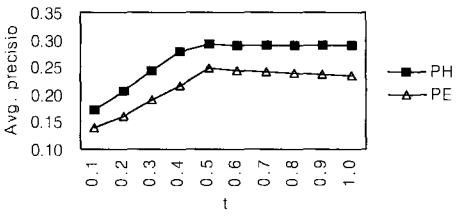

그림 9 t변화 그래프

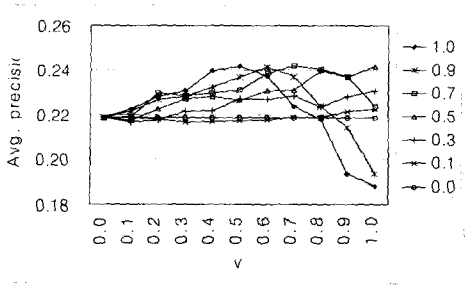

그림 10 f에 따른 v값의 변화

표 1과 그림 7을 보면 PE모델과 PH모델이 상당히 좋은 결과를 산출함을 알 수 있다. 특히 PE 모델의 결과는 문서의 구조상의 근접도 정보를 활용하여 순위화 이용할 경우 좋은 결과를 얻을 수 있다는 것을 간접적으로 나타내어 주는 것이며 PH 모델의 이질성은 가장 좋은 근접도를 가지는 문단를 선택하는 기준으로 작용--구조 정보가 없으므로--하여 PE 모델의 가중치 결과에 반영하여 보다 높은 정확도를 얻게 해준다. 따라서 PE와 PH모델의 근접도가 구조가 없는 내용에 대한 질의에 대해서도 평균 약 67%의 최대 약 128%의 정확도 향상 효과를 가져온다는 것을 실험을 통해서 알 수 있다. 그림 8을 통해서 k값이 커짐에 따라 정확률이 향상되다가 k값이 1000이상이 되면 수렴하는 것을 알 수 있다. 이는 PH 모델의 선택이 거의 대부분의 경우 정확하며 같은 이질성 값을 가지는 것들 중에서 보다 높은 PE 가중치를 가지는 문서들에 대해 순위화하는 것이 의미있는 작업이라는 것을 보여준다. 또한 정규화 방법에 따른 평균 정확률 변화가 크다는 것도 보여준다. 그림 9는 t값이 0.5 정도에서 가장 좋은 결과를 산출하나 0.5이상의 t값을 가진다면 거의 유사한 결과를 얻을 수 있다는 것을 알려준다. 이는 실험을 수행한 테스트 컬렉션의 문서들의 깊이가 거의 유사하여 깊이에 따른 차이가 나타나지 않았기 때문으로 분석된다. 그림 10은 H-distance 당 50% 비율로 감산하여 더하는 것이 효율적이라는 것을 알려주며 문서들의 구조가 유사한 부분에서 질의 단어가 나타나서 레벨에 따른 효과는 효율적으로 반영되지 못했다.

6. 관련연구

구조와 내용을 통합하여 검색하는 것에 대한 조사한 논문으로 [29]이 있다. [29]에서는 기존의 검색 기법과 새로운 접근법들에 대해서 조사하였다. 이 중 XML 문서에 적용할 수 있는 모델로 [5, 30, 31]가 있다. 이들 각각에 대한 설명과 비교는 [29]에 잘 되어 있으므로 다시 언급하지 않겠다. [5, 30, 31] 기법들은 공통적으로 어떻게 가중치를 계산 할 것 인지와 결과 노드들의 가중치를 근접도에 따라 혼합하여 해당 문서의 가중치로 만들 것인가에 대해서는 언급하고 있지 않다. 또한 질의에 대해 정확한 매칭만을 고려하고 있다.

[6]는 상향식 접근법을 [5]에 이어서 이용하였고 문서에 가중치를 주는 방법에 대해서도 제안하였다. 하지만 가중치를 주는데 있어서 단순히 단말 노드들의 가중치를 합산하는 수준에 불과하며 DTD가 존재하는 문서들에 대해서 정확한 매치만을 고려하였다.

[4]는 추론망에 기반한 SGML문서에 대한 검색 모델을 제시하였다. 하지만 추론망에 기반 한다는 것은 한 부모 노드의 자식 노드들간에 독립을 가정하므로 인접도에 따른 가중치를 무시하는 결과가 되는 것이며, 역시 정확한 질의 매칭에 대해서만 언급하고 있다. 또한 중간 노드의 가중치가 내용 질의에 따른 가중치가 아니라 자식 노드들의 총 텍스트 양에 따라 결정된다는 것이 항상 맞는지는 증명되지 않았다. 마지막으로 XML의 tag 중에서 중요한 태그에 보다 높은 가중치를 준다는 것은 문서의 구조를 알아야 한다는 것이므로 이 또한 XML 검색환경에 적합하지 않다.

색인 구조와 관련하여 [26]에서 k차 완전 트리 색인 이용을 제안하였고 [4, 6]에서 역시 사용하였다. 하지만 루트부터 단말 노드까지의 경로 정보를 얻기 위해서는 원소(element) 단위의 접근이 필요하다. [26]에서는 절대경로 색인의 기본 아이디어를 제공하였지만 정보검색의 색인구조 측면이 아니라 XML 문서를 어떻게 효율적으로 저장하고 찾을 것인가에 대해서만 언급하고 있다.

7. 결론 및 향후과제

XML 문서 검색의 특징은 "(1) 문서에 구조가 있다. (2) 비정규적인 문서 구조 가능하다. (3) 검색의 유용성을 높이기 위해서 구조질의를 사용하여야 한다."로 요약할 수 있다. 하지만 현재 제시된 검색 모델들은 이러한 특성을 제대로 반영하지 못하였다. 첫째로 문서에 구조가 있다는 성질에 대해서 관련 문서 선별에만 사용하지 순위화에 사용하지 못했다. 이러한 문제점을 우리는 PE와 PH모델을 이용하여 본 논문에서 제안한 근접도와 이질성으로 해결하였다. 둘째로 비정규적인 문서 구조가 가능하여 사용자가 질의를 구성하기 어려웠던 문제는 ED와 PH 모델의 편집 거리와 이질성을 이용하여 사용자 질의와 가장 유사한 문서 구조부터 순위화 되도록

하는 검색 모델을 제시하였다. 마지막으로 구조 질의의 구성의 어려움과 기존 검색 모델의 확장을 위해서 질의를 SRP(Set of Regular Paths)로 모델링 하였고 이것이 가지는 단점을 PH모델의 이질성을 사용하여 극복하였다. 추가로, 제시한 모델을 효율적으로 수행하기 위한 색인 구조를 제시하였고 제시된 색인 구조의 유용성은 [27]을 통해서 알 수 있다.

XML 문서 검색을 위한 테스트 컬렉션이 없는 현실에서 구조가 없는 내용 질의를 사용하더라도 XML 문서의 구조정보를 활용한 PE와 PH모델이 평균 약 67% 최대 128%의 정확률 개선 효과를 가져온다는 사실을 알게되었고, 본 논문에서 제시한 PE와 PH모델의 유용성을 간접적으로 증명하였다.

향후 과제로는 우선 실험을 수행한 테스트 컬렉션이 작았기 때문에 보다 큰 테스트 컬렉션을 만들어 보다 많은 실험이 필요하다. 실험을 통하여 제안한 모델별 각 매개변수별 특성을 추가로 분석하고 필요하다면 평가하는 수식을 개선하고자 한다. 그 예로 문서길이 정규화 기법에 따라 모델별 성능의 차이가 많이 나타났다. 따라서 각 모델별 타당한 문서길이 정규화 기법에 대한 연구가 더 필요하다. SRP 질의시 전체 경로들의 집합을 하나의 이질성 계산의 단위로 활용하였다. 하지만 SRP 질의 중에서 그룹핑이 가능하다면 하나 이상의 이질성이 계산될 수 있고 이들을 p-norm모델과 같은 모델을 적용하여 이질성의 합을 계산해 낼 수 있을 것이다. 마지막으로 제안된 색인 구조를 사용하여 검색하게 되면 효율적으로 계산할 수 있다. 하지만 검색엔진에 부하가 클 경우 단말 노드가 특정 조건을 만족할 때만 가중치 계산에 이용하도록 하여 부하에 따른 정확률을 조정할 수 있는 시스템을 구성 가능하다.

참 고 문 헌

[1] Neil Bradley. The XML Companion, 2nd Edition. Addison-Wesley, 1999.

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

[3] http://www.w3.org/TR/xquery.

[4] S. H. Myaeng and et. al. A Flexible Model for Retrieval of SGML Documents. SIGIR, pages 138-145, 1998.

[5] Gonzalo Navarro and Ricardo Baeza-Yates. Proximal nodes: a model to query document databases by content and structure. TOIS, 15(4):400-435, 1997.

[6] D. Shin, H. Jang, and H. Jin. BUS: An Effective Indexing and Retrieval Scheme in Structured Documents. journal of Digital Library, pages 235-243, 1998.

[7] V.I. Levvenshtein. Binary codes capable of correcting deletions, insertions, and reversals. Sov. Phys. Dokl., pages 707-710, 1966.

[8] Takeyuki Shimura, Masatoshi Yoshikawa, and Shunsuke Uemura. Storage and Retrieval of XML Documents using Object-Relational Databases. DEXA, pages 206-217, 1999.

[9] Alin Deutsch, Mary F. Fernandez, and Dan Suciu. Storing Semistructured Data with STORED. SIGMOD, pages 431-442, 1999.

[10] Daniela Florescu and Donald Kossmann. Storing and Querying XML Data Using an RDBMS. Data Engineering Bulletin, 22(3), 1999.

[11] Roy Goldman, Jason McHugh, and Jennifer Widom. From Semistructured Data to XML: Migrating the Lore Data Model and Query Language. WebDB, pages 25-30, 1999.

[12] J.P. Callan. Passage-Level Evidence in Document Retrieval. In W. Bruce. Croft and C.J. van Rijsbergen, editors, Proceedings of the Seventeenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 302 310, Dublin, Ireland, July 1994. Spring-Verlag.

[13] Charles L. A. Clarke and Gordon V. Cormack. Shortest-substring retrieval and ranking. TOIS, 18(1):44-78, January 2000.

[14] D. Hawking and P. Thistlewaite. Proximity operators - so near and yet so far.

[15] G. Salton and C. Buckley. Automatic text structuring and retrieval: Experiments in automatic encyclopedia searching. In Proceedings of the 14th Annual International ACM/SIGIR Conference, pages 21-31, 1991.

[16] Ross Wilkinson. Effective retrieval of structured documents. In Proceedings of the 17th Annual International ACM Conference on Research and Development in Information Retrieval (SIGIR '94, Dublin, Ireland, July 3-6), pages 311 317, 1994.

[17] Justin Zobel, Alistair Moffat, Ross Wilkinson, and Ron Sacks-Davis. Efficient retrieval of partial documents. Information Processing and Management, 31(3):361-377, 1995.

[18] C. Clarke, G. Cormack, and F. Burkowski.

Shortest substring ranking (MultiText ex-periments for TREC-4). In D. K. Harman, editor, Proceedings of the 4th Text Retrieval Conference(TREC-4, Washington, D.C., Nov.), pages 295-304, 1995.

[19] M. Kaszkiel and J. Zobel. Passage retrieval revisited. In Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '97, Philadelphia, PA, USA, July 27-31), pages 178-185, 1997.

[20] M. Kaszkiel, J. Zobel, and R. Sacks-Davis. Efficient passage ranking for document databases. TOIS, 17(4):406-439, 2000.

[21] G. Salton, J. Allan, and C. Buckley. Automatic structuring and retrieval of large text files. CACM, 37(2):97-108, 1994.

[22] C. Stanfill and D. L. Waltz. Statistical methods, artificial intelligence, and information retrieval. In P. S. Jacobs, editor, Text-Based Intelligent Systems: Current Research and Practice in Information Extraction and Retrieval, pages 215-225. Lawrence Erlbaum Associates, Inc., 1992.

[23] J. McHugh, S. Abiteboul, R. Goldman, D. Quass, and J. Widom. "Lore: A Database Management System for Semistructured Data", SIGMOD Record, Vol.26 No.3, pp. 54-66, 1997.

[24] S. E. Robertson and S. Walker. Some simple effective approximations to the 2-Poisson model for probabilistic weighted retrieval. SIGIR, pages 232-241, 1994.

[25] Howard R. Turtle and W. Bruce Croft. Evaluation of an Inference Network-Based Retrieval Model. TOIS, 9(3):187-222, 1991.

[26] Yong Kyu Lee, Seong-Joon Yoo, Kyoungro Yoon, and P. Bruce Berra. Index Structures for Structured Documents. journal of Digital Library, pages 91-99, 1996.

[27] Chi young Seo and Hyung-joo Kim. An Efficient Inverted Index Technique Using RDBMS for Supporting Containment Queries. technical report, 2001.

[28] Chun Zhang, Jeffrey F. Naughton, David J. DeWitt, and Guy M. Lohman Qiong Luo. On Supporting Containment Queries in Relational Database Management Systems. SIGMOD, 2001.

[29] Ricardo A. Baeza-Yates and Gonzalo Navarro. Integrating Contents and Structure in Text Retrieval. SIGMOD Record, 25(1):67-79, 1996.

[30] Pekka Kilpelainen and Heikki Mannila. Retrieval from Hierarchical Texts by Partial Patterns. SIGIR, pages 214-222, 1993.

[31] I. MacLeod. A query language for retrieving information from hierarchic text structures. The Computer Journal, 34(3):254-264, 1991.

유 신 재

200년 2월 숭실대학교 컴퓨터학부 학사. 2002년 2월 서울대학교 전기 컴퓨터공학부 석사. 관심분야는 IR, XML & Web DB, Digital libary

민 경 섭

1995년 2월 한국항공대학교 컴퓨터공학과 학사. 1997년 2월 서울대학교 컴퓨터공학과 석사. 1997년~현재 : 서울대학교 인지과학 박사과정. 관심분야는 XML, IR, Database

김 형 주

1982년 서울대학교 전자계산학교(학사). 1985년 Univ. of Texas at Asution(석사). 1988년 5월~1988년 9월 Univ. of Texas at Austin. Post-Doc. 1988년 9월~1990년 12월 Georgia Institute of Technology(부교수). 1991년 1월~현재 서울대학교 컴퓨터공학부 교수