지능정보연구 제17권 제3호 2011년 9월(pp.63~77)

FCA 기반 계층적 구조를 이용한 문서 통합 기법

김태환
한양대학교 컴퓨터공학과
(kimth@islab.hanyang.ac.kr)

전호철
한양대학교 컴퓨터공학과
(hcjeon@cse.hanyang.ac.kr)

최종민
한양대학교 컴퓨터공학과
(jmchoi@hanyang.ac.kr)

월드와이드웹(World Wide Web)은 인터넷에 연결된 컴퓨터를 통해 사람들이 정보를 공유할 수 있는 매우 큰 분산된 정보 공간이다. 웹은 1991년에 시작되어 개인 홈페이지, 온라인 도서관, 가상 박물관 등 다양한 정보 자원들을 웹으로 표현하면서 성장하였다. 이러한 웹은 현재 5천억 페이지 이상 존재할 것이라고 추정한다. 대용량 정보에서 정보를 효과적이며 효율적으로 검색하는 기술을 적용할 수 있다. 현재 존재하는 몇몇 검색 도구들은 초 단위로 gigabyte 크기의 웹을 검사하여 사용자에게 검색 정보를 제공한다. 그러나 검색의 효율성은 검색 시간과는 다른 문제이다. 현재 검색 도구들은 사용자의 질의에 적합한 정보가 적음에도 불구하고 많은 문서들을 사용자에게 검색해준다. 그러므로 대부분의 적합한 문서들은 검색 상위에 존재하지 않는다. 또한 현재 검색 도구들은 사용자가 찾은 문서와 관련된 문서를 찾을 수 없다. 현재 많은 검색 시스템들의 가장 중요한 문제는 검색의 질을 증가 시키는 것이다. 그것은 검색된 결과로 관련 있는 문서를 증가시키고, 관련 없는 문서를 감소시켜 사용자에게 제공하는 것이다.

이러한 문제를 해결하기 위해 CiteSeer는 월드와이드웹에 존재하는 논문에 대해 한정하여 ACI(Autonomous Citation Indexing)기법을 제안하였다. “Citaion Index”는 연구자가 자신의 논문에 다른 논문을 인용한 정보를 기술하는데 이렇게 기술된 논문과 자신의 논문을 연결하여 색인한다. “Citation Index”는 논문 검색이나 논문 분석 등에 매우 유용하다. 그러나 “Citation Index”는 논문의 저자가 다른 논문을 인용한 논문에 대해서만 자신의 논문을 연결하여 색인했기 때문에 논문의 저자가 다른 논문을 인용하지 않은 논문에 대해서는 관련 있는 논문이라 할 지라도 저자의 논문과 연결하여 색인할 수 없다. 또한 인용되지 않은 다른 논문과 연결하여 색인할 수 없기 때문에 확장성이 용이하지 못하다.

이러한 문제를 해결하기 위해 본 논문에서는 검색된 문서에서 단락별 명사와 동사 및 목적어를 추출하여 해당 동사가 명사 및 목적어를 취할 수 있는 가능한 값을 고려하여 하나의 문서를 formal context 형태로 변환한다. 이 표를 이용하여 문서의 계층적 그래프를 구성하고, 문서의 그래프를 이용하여 문서 간 그래프를 통합한다. 이렇게 만들어진 문서의 그래프들은 그래프의 구조를 보고 각각의 문서의 영역을 구하고 그 영역에 포함관계를 계산하여 문서와 문서 간의 관계를 표시할 수 있다. 또한 검색된 문서를 트리 형식으로 보여주어 사용자가 원하는 정보를 보다 쉽게 검색할 수 있는 문서의 구조적 통합 방법에 대해 제안한다. 제안한 방법은 루씬 검색엔진이 가지고 있는 순위 계산 공식을 이용하여 문서가 가지는 중요한 단어를 문서의 참조 관계에 적용하여 비교하였다. 제안한 방법이 루씬 검색엔진보다 15% 정도 높은 성능을 나타내었다.

논문접수일 : 2011년 07월 22일 논문수정일 : 2011년 08월 03일 게재확정일 : 2011년 08월 16일
투고유형 : 국문일반 교신저자 : 최중민

1. 서론

웹 페이지의 증가와 함께 검색엔진의 발전은 기존의 검색에 대한 개념을 바꾸어 놓고 있다. 연구자들은 더 이상 도서관에서 지면으로 발생된 논문지를 찾지 않고, 자신의 책상에서 인터넷으로 간단한 질의어를 입력하여 필요한 자료를 찾는다. 여기에, 대학의 도서관들도 검색엔진과 도서 자료를 유기적으로 조합시켜 공개하고 있다. 도서관에서 적접 연구 자료를 찾는 것과 비교하면, 인터넷을 통한 검색은 많은 양의 정보를 훨씬 편리하게 전달한다. 그러나 정보가 많아지면 많아질수록 실제로 관련 있는 연구 문서를 찾기 어려울 뿐만 아니라 찾았다 하더라도 찾고자 하는 연구 문서와 관련된 문서를 다시 연계해서 찾기는 더 어렵다. 이러한 문제를 해결하기 위해 CiteSeer는 월드와이드웹에 존재하는 논문에 대해 한정하여 ACI(Autonomous Citation Indexing)기법을 제안하였다(Giles, 1998). ACI 기법은 연구자가 자신의 논문에 다른 논문을 인용한 정보를 기술하는데 이렇게 기술된 논문과 자신의 논문을 연결하여 색인하는 방법이다. ACI 기법은 논문 검색이나 논문 분석 등에 매우 유용하다. 그러나 논문의 저자가 다른 논문을 인용한 논문에 대해서만 자신의 논문을 연결하여 색인 하기 때문에 논문의 저자가 인용하지 않은 다른 논문에 대해서는 관련있는 논문이라 할 지라도 저자의 논문과 연결하여 색인할 수 없다. 또한 인용되지 않은 다른 논문과 연결하여 색인할 수 없기 때문에 확장성이 용이하지 못하다.

본 논문에서는 이러한 단점들을 해결하기 위해 데이터 분석을 위해 사용되었던 FCA(Formal Concept Analysis)를 이용하여 문서를 하나의 계층적 개념 그래프로 표현한다. 문서 간 개념 그래프를 병합하여 만든 통합 문서 그래프에 각각의 문서가 가지는 영역을 구한다. 포함 관계를 통해 문서의 참조 관계가 있는지 판별하여 각각이 가지고 있는 문서에 참조 관계를 가지는 문서를 연결하여 참조 문헌 정보가 없어도 참조 관계가 형성되는 문서 간의 계층적 구조를 이용한 문서의 통합 방법을 제안하려 한다.

2. 관련 연구

이 장에서는 정보 검색 기술과 CiteSeer에서 사용하고 있는 검색 기법 및 데이터 분석을 위해 사용되었던 FCA의 개념에 대하여 설명한다.

2.1 정보검색기술

전통적인 방식의 검색 기술은 4가지로 요약된다. 첫 번째는 전문주사 방식(Full text scanning)으로 찾고자 하는 모든 대상 문서에 대해 원하는 단어가 포함되어 있는지를 문자열 검색으로 찾아보는 방식이다. 이 방식의 성능은 문자열 검색 알고리즘에 좌우된다(Boyer, 1977). 이 방식이 다른 방식과 비교하여 유리한 점은 색인 정보를 필요로 하지 않기 때문에 별도의 저장 공간을 필요로 하지 않고, 대상 문서의 추가와 삭제에 부담이 없다는 것이다. 그러나 대상 문서가 많을 경우 성능이 저하된다는 단점을 갖는다. 따라서 대부분 제한된 영역 내에서 활용된다. 이 검색 방법은 문서에 원하는 단어만을 고려하여 검색함으로 단어에 의존한 문서 검색이기 때문에 문서와 문서의 참조 관계를 알아내려는 방법에 사용할 수 없다.

두 번째는 시그니춰 비교 방식(Signature matching)으로 각 문서는 그에 포함된 단어들을 대상으로 해싱 등의 방법을 이용하여 비트열(Signature)를 만든 다음 별도의 파일(Signature file)을 생성하

여 이들을 비교하는 방법이다(Meadow, 1992). 시그니춰 파일은 원문보다 훨씬 작은 공간을 차지하며, 검색 성능도 비교적 빠르다. 이 방식은 문서의 크기가 큰 경우 성능이 저하되는 단점이 있으나, 구현이 용이하고 대상 문서의 추가와 불완전한 질의어 처리에 용이한 장점이 있다. 이러한 방법으로 문서의 참조 관계를 형성하기 위해서는 문서 자체를 비트화하여 비교해야 하는데 문서 자체를 비트화하기 어려울 뿐만 아니라 비트화 해서 타 문서를 검색했다 하더라도 유사한 문서라고 판단 할 수 없다.

세 번째는 가장 널리 사용되고 있는 방식인 역색인 검색(Inverted file search) 방식으로서, 각 문서는 그 문서를 대표하는 키워드로 표현되고, 키워드를 중심으로 그 키워드가 사용되고 있는 문서 번호를 저장하는 색인 파일을 생성한다(Salton, 1989). 이 방식의 단점은 색인 파일의 과도한 크기로 인한 저장 장소의 과중과 검색 대상 문서가 수시로 변하는 동적인 환경에서 색인(키워드)의 추가 및 삭제에 따른 관리 부담 등을 들 수 있다. 이 검색 방법은 문서를 대표하는 키워드를 이용하여 문서를 검색하는 방식이기 때문에 이것을 문서의 참조 관계를 가지고 있는 그 대상을 찾았다 하더라도 참조 관계라고 할 수 없다. 의미적 관계를 기반으로 검색한 것이 아니라 키워드 매칭을 통해 검색된 데이터이기 때문이다.

네 번째는 벡터 공간 모델(Vector space model) 방식으로서, 각 문서와 질의어는 선정된 단어들의 용어 벡터(Term vector)로 표현되어 유사성을 검사하는 방식이다(Mingjun, 2002). 모델이 간단하고 가중치에 따른 검색 순위 매김이 용이하며 또한 관련 반응(Relevance feedback)과 같은 동적 환경에 잘 적응하는 장점이 있다. 이 검색 방법을 이용해 문서를 하나의 문서 벡터로 만들어 낼 수 없기 때문에 문서의 검색에는 사용할 수 없다.

2.2 FCA(Formal Concept Analysis)

Formal Context는 문서가 가지고 있는 문장들 내에서 객체(주어)와 속성(동사)를 추출해낸 결과의 집합을 이야기한다. 정형적으로 Formal Concept K는 $K = (G, M, I)$로 정의되며, 객체들의 집합 G와 속성들의 집합 M, 그리고 G와 M사이의 이항관계 $I \subseteq G \times M$으로 구성된다. 이항관계 I는, "G의 원소 g과 M의 원소 m을 가지고 있다"라는 의미이다. 이 정의를 이용하여 두 집합 A', B'를 다음과 같이 정의하였다.

$$A' := \{m \in M \mid \forall g \in A : (g, m) \in I\} \quad (1)$$

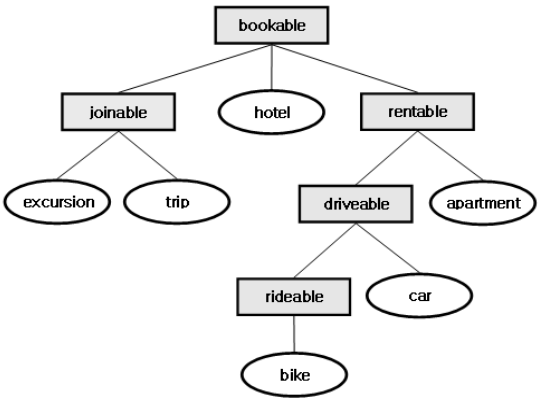
$$B' := \{g \in G \mid \forall m \in B : (g, m) \in I\} \quad (2)$$

이 때 집합 A는 $A \subseteq G$이고, 집합 B는 $B \subseteq M$이다. 여기서 $(A = B')$와 $(B = A')$를 만족시킬 때, A는 extent라고 부르고 B는 intent라고 부른다. 이와 같은 정의를 바탕으로 문장으로부터 다양한 개념 (g, m)을 추출할 수 있다. 이러한 개념들 사이에는 일종의 상-하위 관계에 따른 순서가 존재한다. 즉, 임의의 개념$(A1, B1)$과 $(A2, B2)$에 대하여 $A1 \subseteq A2$이거나 $B1 \subseteq B2$일 때, 개념$(A1, B1)$은 개념$(A2, B2)$의 하위개념이라고 하며, 반대로 개념$(A2, B2)$은 개념$(A1, B1)$의 상위 개념이라고 한다.

<표 1> Formal Context

	Bookable	Rentable	Driveable	Rideable	Joinable
Hotel	X				
Apartment	X	X			
Car	X	X	X		
Bike	X	X	X	X	
Excursion	X				X
Trip	X				X

이와 같이 주어진 문장으로부터 개념들을 추출하여 상위개념-하위개념 관계를 구성함으로써 격자구조를 구축할 수 있다. 또한 추출된 개념들은 자연스럽게 객체집합이나 속성집합에 의한 계층적 관계가 형성이 되며 이를 통해 개념격자(Concept Lattice)를 구축할 수 있고, 개념격자는 개념을 나타내는 정점들과 상 하위 개념 관계를 나타내는 변으로 구성되고 이것을 <표 1>과 같은 형태로 만든다. <표 1>에서 가로는 속성을 나타내고 세로는 개체를 나타낸다. 객체가 가질 수 있는 속성의 수가 가장 많은 것이 상위 노드가 되고, 적은 것이 하위 노드가 된다. 이러한 방법으로 <표 1>는 <그림 1>과 같은 문서 개념 트리로 변환될 수 있다. 이 트리의 각 노드에는 여러 개념을 갖는 extent들과 intent들에 대한 정보가 레이블로 표시된다(Eijck, 2004).

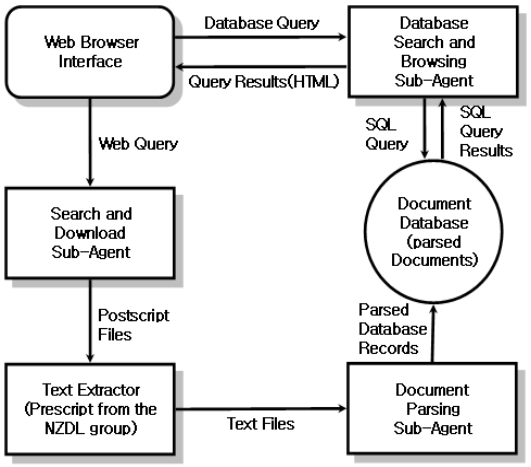


<그림 1> Formal Concept Tree

2.3 CiteSeer

CiteSeer는 과학관련 논문들의 전자도서관으로서 기본적인 논문자료 열람과 검색기능 이외의 다양한 추가기능을 제공한다(CiteSeer, 1998). CiteSeer의 대표적인 서비스인 ACI는 <그림 2>와 같이 한 논문에서 인용된 다른 논문들의 목록을 논문원문에서 자동으로 색인화하여 연결을 생성해주는 기능이다. 또한 CiteSeer는 인용된 논문들을 계속해서 갱신하여 연결하기 때문에, 한 논문과 관련된 최근의 연구결과들을 찾아볼 수 있다. 이외에도 검색시 질의어와 관련된 부분만을 본문에 요약해서 보여주는 기능, 논문의 인용빈도를 나타내는 그래프 작성 기능을 제공하고 있다(Giles, 2004).

그런데, CiteSeer는 논문에서 인용된 다른 논문들의 목록만을 보여주기 때문에 저자가 인용하지 않은 다른 논문들에 대한 정보는 표기하지 않는다. 또한 저자가 인용하지 않은 다른 논문들에 관한 참조관계는 표현할 수 없다. CiteSeer는 논문간의 인용관계에 한정하였기 때문에 논문이 아닌 타문서간의 연관관계나 인용되지 않은 다른 논문에 대해 확장하여 표현할 수 없다.



<그림 2> CiteSeer Agent Architecture

이 논문에서는 문서가 가지고 있는 그래프들을 통합하여 문서 A에 가지고 있는 객체와 속성들의 영역을 구하고, 문서 B에 가지고 있는 객체와 속성들의 영역을 구하여 그영역이 문서 B가 문서 A

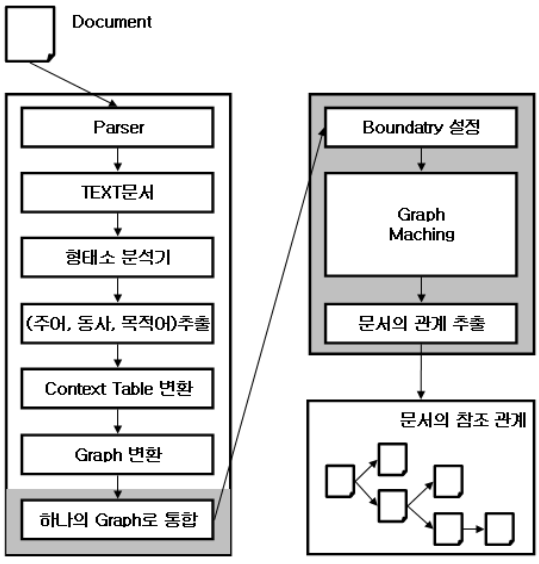
에 포함되었다면 문서 A는 문서 B를 참조한다고 가정하였다. 그래서 명시적인 참조관계가 존재하지 않아도 병합트리를 이용하여 참조관계를 파악할 수 있다.

3. 시스템 절차도

<그림 3> 본 논문의 시스템 절차도를 나타낸 그림이다. 시스템 절차도는 <그림 3>과 같이 크게 두 가지 단계로 구성된다. 첫 번째는 문서와 문서가 나타내고 있는 그래프 통합이고, 두 번째는 통합된 문서를 비교하여 각각의 문서의 영역을 구한다. 이렇게 구해진 영역을 비교하여 상-하위 관계를 정의하고 정의된 내용에 따라 참조 관계를 나타내는 단계이다.

3.1 시스템 구조

첫 번째 문서의 통합 과정은 다음과 같다.



<그림 3> 시스템절차도

3.1.1 Formal Context 정보 변환

알고리즘 1은 문서들을 formal context로 변환하는 알고리즘이다.

라인 2~3은 문서의 문장들을 분석하기 위한 반복구문이다. 라인 4은 스탠포드 파서(Manning, 2002)를 이용하여 문장이 가지고 있는 구문을 분석하여 주어 또는 목적어를 가지는 단어를 분석한다. 라인 5는 스탠포드 파서를 이용하여 문장이 가지고 있는 구문을 분석하여 동사를 가지는 단어를 분석한다. 라인 6는 문장이 가지는 주어 또는 목적어와 동사를 하나의 쌍으로 나타낸다. 라인 7~9은 문장이 하나 이상의 쌍으로 나타나는 경우 formal_context_check() 함수를 이용하여 formal context 정보로 변환한다.

Algorithm 1 Formal Context

D = {set of documents}
D_i = {the i^{th} document in D}
W_{ij} = {W_{ij} | W_{ij} is the j^{th} sentence in document D_i }
C_{ij} = {c|c is a subject or an object in the i^{th} sentence, W_{ij}}
A_{ij} = {a|a is a predicate in the sentence, W_{ij}}
P_{ij} = {<c, a> | c∈C_{ij}, a∈A_{ij}}

input : D_i **output** : Formal Context

```
1 : function ConvertDocument(doc_file D)
    // doc_file : file name
2 :   for all documents Di do
3 :     for all sentences Wij do
4 :       Cij ← CSParser(Wij)
5 :         Aij ← ASParser(Wij)
6 :           P ←< Cij, Aij >// set of pair for the
                              subject and object
7 :         for all sizeof(P) do
8 :       formal_context_check(sizeof(P))
9 :     endfor
10 :   endfor
11 :  endfor
12 : end function
```

3.1.2 문서간의 관계 정의

정형적으로 Formal Context K는 $K = (G, M, I)$로 정의되며, 객체들의 집합 G와 속성들의 집합 M, 그리고 G와 M 사이의 이항관계 $I \subseteq G \times M$으로 구성된다. 집합 D는 문서 집합을 타나내며, FCA에 적용하기 위하여 다음과 같이 정의한다.

$$D = (C, A, P) \tag{3}$$

이때 C는 문서에서 나타내고 있는 객체, A는 속성을 말한다. P는 문장의 정보를 가지고 있는 2차원 배열이라 하며 이항관계로서 $P \subseteq I$이다.

정의 1 : 불일치관계

통합된 문서가 가지고 있는 객체와 속성의 그래프에서 각각의 문서가 가지고 있는 영역이 서로 매칭이 되지 않는 경우

$$A_{ij} = \{a | (c_i, a) \in P_j, c_i \in C\} \tag{4}$$

$$C_{ij} = \{c | (c, a_i) \in P_j, a_i \in A\} \tag{5}$$

식 (6)을 만족하면 A_{ij}와 A_{ik}는 서로 불일치 관계에 있다고 한다. P_j는 j개의 문장 내에 객체와 속성을 가지는 이차배열의 집합이다.

$$(A_{ij} \cap A_{ik} = \emptyset) \wedge (C_{ij} \cap C_{ik} = \emptyset) \tag{6}$$

이때 두 문서 d_i와 d_k는 관련이 없는 문서라 정의한다.

정의 2 : 포함관계

통합된 문서가 가지고 있는 객체와 속성의 그래프에서 문서 A와 문서 B가 가지고 있는 영역이 객체와 동일하게 매칭이 되고, 속성에는 문서 A가 문서 B에 속한다면 문서 B는 의미의 확장이 있었다고 말하며 문서 B는 문서 A를 참조하는 경우이다. 즉, 두 문서 d_i와 d_k에 대하여 식 (7)을 만족하면 두 문서 d_k는 문서 d_i를 참조한다고 정의한다.

$$(A_{ij} \subset A_{ik}) \wedge (C_{ij} = C_{ik}) \tag{7}$$

정의 3 : 유사관계

통합된 문서가 가지고 있는 객체와 속성의 그래프에서 문서 A와 문서 B가 가지고 있는 영역이 객체와 속성 모두 똑같이 매칭 된다면 두 문서는 유사한 문서이고, 참조 문서와 다른 의미를 가진다. 즉, 두 문서 d_i와 d_k에 대하여 식 (8)을 만족하면 두 문서 d_i와 문서 d_k는 유사한 문서로 정의한다.

$$(A_{ij} = A_{ik}) \wedge (C_{ij} = C_{ik}) \tag{8}$$

정의 4 : 의미적 유사관계

통합된 문서가 가지고 있는 객체와 속성의 그래프에서 문서 A와 문서 B가 가지고 있는 영역이 객체와 속성이 동일하게 매칭이 되지 않는다. 사용된 객체는 다르지만 객체가 사용하고 있는 속성이 문서 A가 문서 B에 속한다면 문서 A와 문서 B는 의미적으로 유사하다고 정의한다. 사용되는 속성은 객체가 행하고자 하는 행위에 대한 의미이기 때문에 의미적으로 유사하다. 즉, 두 문서 d_i와 d_k에 대하여 식 (9)을 만족하면 두 문서 d_k는 문서 d_i를 참조 한다고 정의한다.

$$(A_{ij} = A_{ik}) \wedge (C_{ij} = C_{ik}) \tag{9}$$

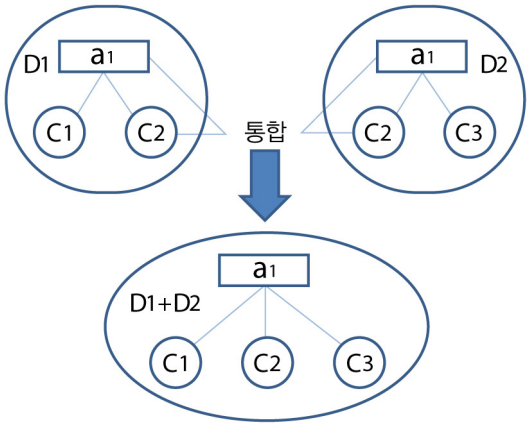
정의 5 : 부분참조관계

통합된 문서가 가지고 있는 객체와 속성의 그래프에서 문서 A와 문서 B가 가지고 있는 영역이 객체가 나타내는 범위와 속성이 나타내는 범위가 부분적으로 같다면 두 문서는 부분 참조 관계에 있다고 한다. 즉, 두 문서 d_i와 d_k에 대하여 식 (10)을 만족하면 두 문서 d_i와 d_k는 부분 참조 관계에 있다고 정의한다.

$$(A_{ij} \cap A_{ik} \neq \emptyset) \wedge (C_{ij} \cap C_{ik} \neq \emptyset) \qquad (10)$$

3.1.3 수평적 계층의 통합

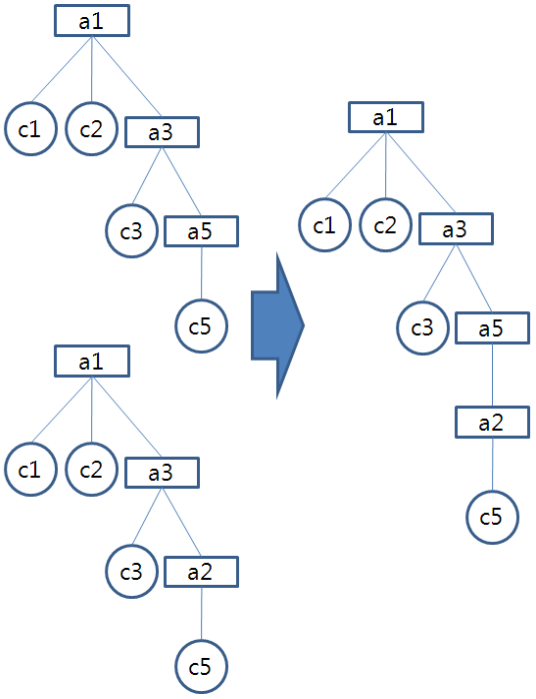
객체와 속성이 두 개의 문서에 같이 존재하고 트리의 구조가 다를 때 <그림 4>와 같이 문서의 통합이 이루어진다. 여기서 D는 문서, a는 속성, c는 객체를 나타낸다.



<그림 4> 수평적 계층의 통합 예

3.1.4 수직적 계층의 통합

문서 간 상위 속성이 같고 그 문서 간 사용된 객체가 존재한다면 그 객체가 사용된 속성이 다른 것에 상관없이 <그림 5>와 같이 통합 될 수 있다.



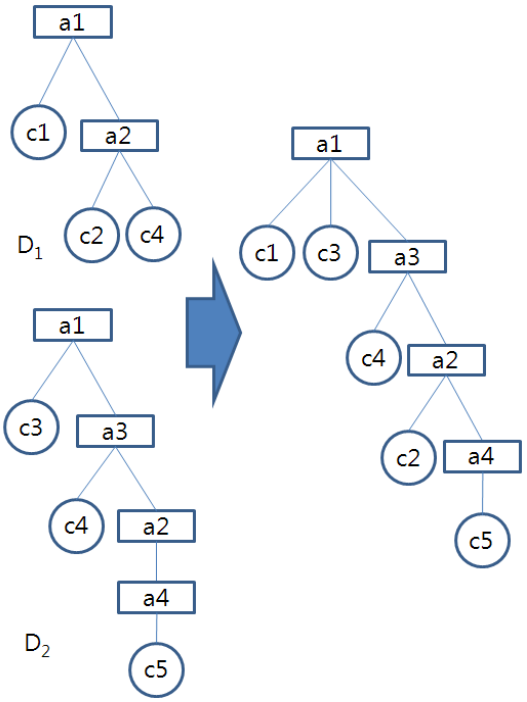
<그림 5> 수직적 계층의 통합 예

3.2 통합된 그래프에서 문서의 영역 설정

두 번째 단계는 통합된 문서 내에서 비교하고자 하는 문서를 대비 시켜 그 문서가 가지는 영역을 구한다. 이렇게 구해진 영역을 이용하여 문서의 상하위 관계를 결정한다.

<그림 6>은 문서의 영역을 설정하는 예이다. 문서 D_1에서 나타난 객체는 c_1, c_2, c_4이고, 객체가 사용하고 있는 속성은 a_1, a_2이다. 하지만 문서의 통합이 이루어진 이후에 D_1에서 나타난 객체는 c_1, c_2, c_4로 똑같지만, 사용된 속성은 a_1, a_2, a_3로 문서 D_1이 가지고 있던 속성 a_1, a_2에서 a_1, a_2, a_3로 확장되었다. 문서 D_2에서 나타난 객체는 c_3, c_4, c_5이고, 객체가 사용하고 있는 속성은 a_1, a_2, a_3, a_4이다. 본 논문의 제3.1.2절의 정의 4에 의하여 D_1은 D_2 문서를 상위 문서로 가지고 있다. 즉, D_1 문서는 D_2 문

서보다 더 상세하게 기술하고 있고, D_2는 D_1보다 일반적으로 기술되었다. 결과적으로 D_1 문서보다 D_2 문서가 더 상위에 있다고 결정한다.



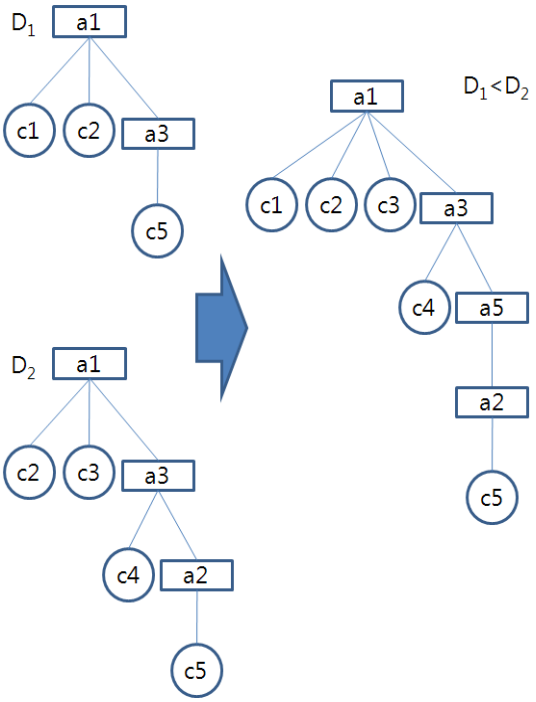
<그림 6> 문서의 영역설정 예

4. 서비스

통합된 문서를 이용한 서비스는 크게 두 가지가 있다. 첫 번째는 임의의 키워드에 대한 문서의 순위를 표현해 주는 서비스이며, 두 번째는 선택된 문서에 대해서 더 구체적(Specific)으로 설명한 문서와 더 일반적(General)으로 설명한 문서는 어떤 것들이 있는지 계층적으로 표현 해주는 서비스이다.

4.1 키워드를 이용한 문서의 순위

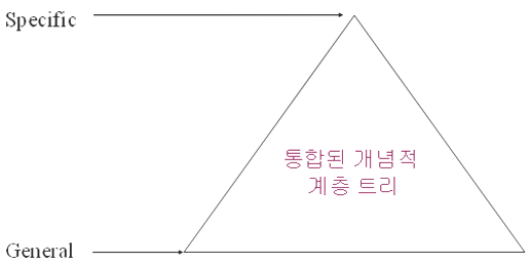
문서 D_1에서 나타난 객체가 사용하고 있는 속성의 수가 문서 D_2의 같은 객체가 사용하고 있는 속성의 수보다 적을 때 해당 객체에 대하여 D_1보다 D_2가 더 비중이 있다고 한다. 이를 이용하여 <그림 7>과 같이 문서 순위를 표현할 수 있다. c_5로 문서를 검색할 때 D_1과 D_2에서, c_5가 가질 수 있는 속성의 수로써 검색 순위를 $D_1 < D_2$로 표현할 수 있다.



<그림 7> 문서간 관계 예

4.2 선택된 문서를 이용한 계층적 문서 구조

계층적 문서 구조를 표현하기 위해서는 문서 간의 상위, 하위, 및 동등 관계를 이용한다. 현재 문서의 root 노드와 leaf 노드의 객체를 타 문서와 비교하여 현재 문서의 leaf 노드의 객체의 깊이가 타 문서보다 더 깊고 같은 트리 상에 위치해 있다면 현재 문서는 타 문서 보다 더 일반적이다.



<그림 8> 통합된 트리상의 General and Specific

만일 현재 문서의 깊이가 타 문서의 깊이와 유사하고 문서 간 같이 사용된 객체와 속성이 존재한다면 두 문서는 서로 동등 관계에 있다. 이러한 관계를 바탕으로 선택된 문서의 계층 구조로 문서를 표현할 수 있게 된다.

5. 실험 및 평가

5.1 실험 방법

현재 존재하는 검색 엔진 중 선택된 문서를 이용하여 계층적 관계를 표현해 주는 연구는 없다. 본 연구의 성능을 평가하기 위해 루씬 검색엔진을 이용하여 비교 평가하였다. 루씬 검색엔진이 가지고 있는 순위 계산 공식을 이용하여 문서가 가지는 중요한 단어를 문서의 참조 관계에 적용하였다. 이렇게 적용된 실험 데이터와 제안한 논문의 데이터를 비교하였다. 비교 방법은 문서의 참조 관계가 있는 대상 문서를 가지고 참조 관계를 얼마나 복원해 주는지 정확률과 재현율 그리고 F-지수로 나타내었다.

5.1.1 비교 평가 대상

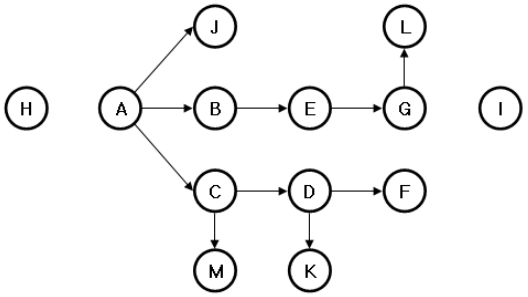
루씬의 순위 점수 계산 알고리즘은 단어의 개수가 적고 짧은 필드가 상대적으로 높은 점수를 얻는다. 다음은 루씬에서 사용하는 질의 q에 대한 문서 d의 점수를 계산하는 공식이다.

$$score(q,\ d) = \sum_{t \in q} tf(t \in d) \times lengthNorm(t.field \in d) \times coord(q,\ d) \times queryNorm(q) \quad (11)$$

tf(*t* in *d*)는 문서 *d*안에서 단어 *t*가 몇 번 나타나는지 표시하는 빈도수 이다. *idf*(*t*)는 역 파일 색인에서 단어 t를 포함하는 문서의 빈도수의 역수이며, 해당 단어를 포함하는 문서가 많을수록 단어 t는 흔히 사용하는 단어라 판단하여 유사도가 떨어진다고 판단한다. *lengthNorm*(*t.field in d*)는 필드에 포함된 단어 개수를 정규화한 값으로 색인 할 때 계산해 저장된다. *Coord*(*q, d*)는 문서에 포함된 개별 검색어의 가중치의 제곱의 합을 정규화한 값을 나타낸다. 이 공식을 이용하여 각각의 문서가 가지고 있는 가장 중요한 키워드를 이용하여 주변의 문서를 검색하고 검색된 관계를 설정한다.

5.1.2 참조 관계 비교를 위한 데이터

인터넷 상에서 문서의 참조 관계를 나타내고 있는 문서는 연구 논문 밖에 없기 때문에 이 실험에서 사용하는 데이터는 연구 논문이다. 각각의 연구 논문들의 참조 관계는 그림 x와 같은 형태로 나타난다.



<그림 9> 데이터 참조 관계의 예

참조 관계가 있는 데이터들을 참조 관계가 없는 문서로 바꾸어 루씬과 제안하는 방법으로 참조 관계를 복구 시켰을 때 각각의 정확률, 재현율, F 지수를 이용하여 비교 평가한다. 정확률은 전체 참조 관계가 얼마나 정확하게 복구 되었는지의 비율이며 재현율은 전체 적합한 참조 관계에 대한 추출된 참조 관계의 비율이다. 정확률과 재현율은 반비례적인 경향을 가지고 있기 때문에 정확률을 높이면 재현율이 낮아지고 재현율을 높이면 정확률이 낮아진다. 상반된 두 지수를 종합하여 시스템의 성능을 객관적으로 수치화 한 지수가 F-지수이다. F-지수는 정확률과 재현율을 모두 고려한 방법으로 각각의 식은 다음과 같다.

$$\text{평균재현율} = \frac{\text{검색된 상위적합문서수} + \text{검색된 하위적합문서수}}{\text{상위적합문서수} + \text{하위적합문서수}} \quad (12)$$

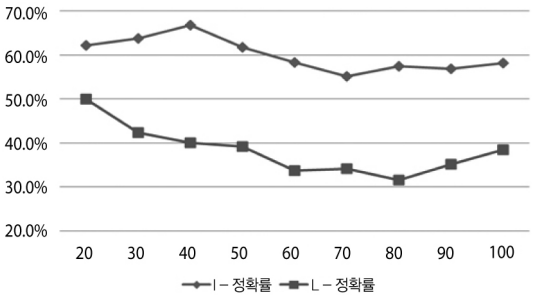
$$\text{정확률} = \frac{\text{검색된 상위적합문서수} + \text{검색된 하위적합문서수}}{\text{검색된 상위문서} + \text{검색된 하위문서수}} \quad (13)$$

$$F-\text{지수}(P, R) = \frac{2PR}{P+R} \quad (14)$$

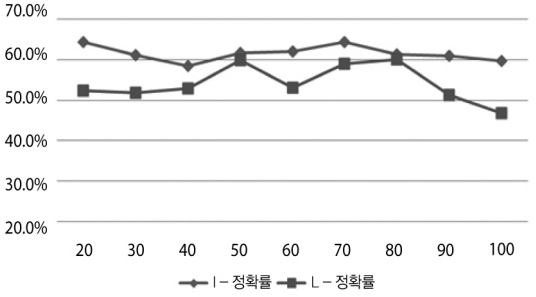
실험은 문서와 문서간 참조 관계를 형성하고 있는 논문 67개와 참조 관계와 관련이 없는 문서 23개 총 100개를 가지고 실험하였다.

5.2 실험 평가

<그림 10>에서 I는 문서의 통합 기법을 이용한 문서의 참조관계를 가리키고, L은 루씬 검색엔진을 통한 문서의 검색을 가리킨다. 루씬의 순위 계산 공식을 이용하여 문서의 대표단어를 추출하고 추출된 단어를 이용하여 문서를 검색한다. 검색된 문서는 추출된 단어의 문서가 관련이 있다고 정의하고 문서의 통합 기법과 비교 평가 하였다. <그림 10>은 실험 평가에 대한 정확률을 나타낸다. 문서의 수가 증가할수록 문서와 문서가 가지고 있는 방향이 맞는 링크를 얼마나 복원해 주는가에 대한 결과이다. I는 문서의 수가 50개까지 정확률 모두 올라가는 것을 볼 수 있다. 이에 반해 L는 문서의 수가 증가할수록 정확률이 내려가는 것을 볼 수 있다. 이는 참조 관계를 형성하는 링크가 문서의 수가 증가할수록 링크의 수가 증가하는데 검색 순위보다 제안한 방법이 참조된 링크를 정확하게 파악했다는 것을 의미한다.



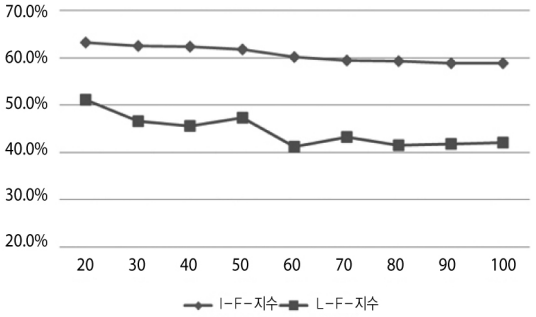
<그림 10> 정확률 비교



<그림 11> 재현율 비교

<그림 11>는 실험 평가에 대한 재현율을 나타낸다. <그림 11>에서 나타나는 것처럼 I의 평균 재현율은 61.5% 이고, L의 재현율은 54.1%이다.

이 것은 *L*이 실험 데이터에서 사용된 참조 관계를 복원하는 것 보다 *I*가 참조 관계를 복원하는 것이 더 올바르게 복원 하였다. 즉, <그림 10>과 <그림 11>의 실험 결과는 *I*가 *L*보다 복원하고자 하는 참조 관계를 더 많이 찾아냈다. 또한 *I*는 *L*보다 상대적으로 잘못 복원한 참조 관계의 수가 적다. 전반적으로 *L*이 가지는 성능보다 *I*가 가지는 성능이 월등히 뛰어남을 확인할 수 있다.



<그림 12> F-지수

<그림 12>는 정확률과 재현율을 모두 반영하기 위해 *F*-지수를 사용한 결과이다. *I*-*F*-지수는 전체 문서의 참조관계의 복원이 60.7% 복원이 되었음을 알 수 있고 *L*-*F*-지수는 44.5% 복원이 되었다. 즉 *L*은 참조 관계의 링크는 많은데 정확하게 일치하는 링크는 적다는 것을 알 수 있다. 이에 비해 *I*는 참조 관계의 링크 수가 일정 증감 이상의 증가는 이루어 지지 않고 성능도 *L*에 비해 15% 정도 더 높은 성능을 나타내는 것을 알 수 있다.

6. 결론 및 향후 과제

6.1 결론

이 논문에서는 계층적 개념 그래프를 이용하여 문서를 통합하고 통합된 문서에 대해서 각각의 문서를 계층적으로 표현할 수 있는 방법을 제시하였다. 이 방법의 문제점은 다음과 같다.

1. 문서 A에서 사용된 문서와 문서 B에서 사용된 동사는 같은데 의미가 다른 경우 통합 그래프에 병합되어 잘못된 참조 관계를 형성할 수 있다.
2. 문장의 주어, 동사, 목적어를 추출하여 문서의 그래프를 형성하고 통합하는데 현재, 주어, 동사, 목적어를 정확하게 추출해 줄 만한 형태소 분석기가 존재하지 않는다.

이 방법의 이점은 다음과 같다.

1. 연구 논문에 관련된 자료 검색 시 연구자가 참고 논문에 기록하지 않았던 논문까지 문서의 계층 구조로 검색 가능하다.
2. 계층적 개념 그래프로 표현하였기 때문에 연구 논문을 바탕으로 요약된 타 문서의 검색도 용이해 졌다

6.2 향후 과제

통합된 문서의 그래프를 이용해 참조 관계를 복원해 주는 것이 어느 정도 타당함을 보여 줌으로써 통합된 문서의 그래프를 다음과 같이 이용할 수 있다.

1. 명사와 동사를 이용하여 질의어를 작성하고, 파싱된 명사와 동사를 통해 통합된 문서의 그래프가 가지는 영역을 구해 해당 영역을 포함하고 있는 문서를 검색해 주는 새로운 우선순위 검색 엔진을 만들 수 있다.
2. 통합된 문서의 그래프가 가지고 있는 특성을

살려 그래프 색인 방법을 만들 수 있다. 이 색인 방법으로 문서와 문서간의 참조 관계를 검출해 주는 검색엔진을 만들어 낼 수 있다.

참고문헌

Boyer, R. S. and J. S. Moore, "A fast string searching algorithm", *CACM*, Vol.20, No.10 (1977), 762~772.

CiteSeer., http://citeseer.ist.psu.edu.

Eijck van Jan, and Z. Joost, "Formal Concept Analysis and Prototypes", 2004.

Giles, C. L., K. D., Bollacker and S. Lawrence, "CiteSeer : An Automatic Citation Indexing System", *ACM Conference*, 1998.

Giles, C. L., "CiteSeer : Past, Present, and Future", *Computer Science Adances in Web Intelligence*, 2004.

Manning, C. and D. Klein, "Fast Exact Inference with a Factored Model for Natural Language Parsing", *Advances in Neural Information Processing Systems*, Vol.15(NIPS, 2002).

Meadow, C. T., "Text Information Retrieval Systems", Academic Press, Inc, 1992, 201~211.

Mingjun, L., Y. Shui, B. Ruth, and Z. Walei, "A Co-Recommendation Algorithm for Web Searching", Fifth International Conference on Algorithms and Architectures for Parallel Processing ICA3PP '02). *IEEE International Conference*, 2002.

Philipp, C., H. Andreas and S. Steffen, "Learning Concept Hierarchies from Text Corpora using Formal Concept Analysis", *AI Access Foundation*, 2005.

Salton, G., "Automatic Text Processing", *Addison-Wesley Publishing Company*, 229~236, 1989.

Abstract

Methods for Integration of Documents using Hierarchical Structure based on the Formal Concept Analysis

Taehwan Kim* · Hocheol Jeon* · Joongmin Choi*

The World Wide Web is a very large distributed digital information space. From its origins in 1991, the web has grown to encompass diverse information resources as personal home pasges, online digital libraries and virtual museums. Some estimates suggest that the web currently includes over 500 billion pages in the deep web. The ability to search and retrieve information from the web efficiently and effectively is an enabling technology for realizing its full potential. With powerful workstations and parallel processing technology, efficiency is not a bottleneck. In fact, some existing search tools sift through gigabyte-syze precompiled web indexes in a fraction of a second. But retrieval effectiveness is a different matter. Current search tools retrieve too many documents, of which only a small fraction are relevant to the user query. Furthermore, the most relevant documents do not nessarily appear at the top of the query output order. Also, current search tools can not retrieve the documents related with retrieved document from gigantic amount of documents. The most important problem for lots of current searching systems is to increase the quality of search. It means to provide related documents or decrease the number of unrelated documents as low as possible in the results of search.

For this problem, CiteSeer proposed the ACI (Autonomous Citation Indexing) of the articles on the World Wide Web. A “citation index” indexes the links between articles that researchers make when they cite other articles. Citation indexes are very useful for a number of purposes, including literature search and analysis of the academic literature. For details of this work, references contained in academic articles are used to give credit to previous work in the literature and provide a link between the “citing” and “cited” articles. A citation index indexes the citations that an article makes, linking the articleswith the cited works. Citation indexes were originally designed mainly for information retrieval. The citation links allow navigating the literature in unique ways. Papers can be located independent of language, and words in thetitle, keywords or document. A citation index allows navigation backward in time (the list of cited articles) and forwardin time (which subsequent articles cite the current article?) But CiteSeer can not indexes the links between articles that researchers

* Department of Computer Science and Engineering, Hanyang University

doesn't make. Because it indexes the links between articles that only researchers make when they cite other articles. Also, CiteSeer is not easy to scalability. Because CiteSeer can not indexes the links between articles that researchers doesn't make.

All these problems make us orient for designing more effective search system. This paper shows a method that extracts subject and predicate per each sentence in documents. A document will be changed into the tabular form that extracted predicate checked value of possible subject and object. We make a hierarchical graph of a document using the table and then integrate graphs of documents. The graph of entire documents calculates the area of document as compared with integrated documents. We mark relation among the documents as compared with the area of documents. Also it proposes a method for structural integration of documents that retrieves documents from the graph. It makes that the user can find information easier. We compared the performance of the proposed approaches with lucene search engine using the formulas for ranking. As a result, the F-measure is about 60% and it is better as about 15%.

Key Words : Related Documents Retrieval, Integration of Documents, Semantic Expansion

저 자 소 개

김태환

인천대학교 컴퓨터공학과를 졸업하였고, 2007년에 한양대학교 대학원 컴퓨터공학과에서 석사학위를 취득하였다. 2007년부터 2011년 현재까지 한양대학교 박사과정 중이다.

한국 정보과학회, 정보처리학회, 지능정보시스템학회, 인터넷정보학회 등의 정회원(혹은 준회원)이며, 관심분야는 웹지능, 텍스트마이닝, 정보검색/정보추출, 인공지능, 지능형 모바일정보시스템 등이다.

전호철

서원대학교 전자계산학과 공학사(1998), 한양대학교 컴퓨터공학과 공학석사(2000), 한양대학교 컴퓨터공학과 공학박사(2011), 현재 한양대학교 컴퓨터공학과에서 Post-Doc로 재직 중이며, 주요 관심분야는 지능형 에이전트, 정보검색/정보추출, 인공지능, 상황인지 등이다.

최중민

서울대학교 컴퓨터공학과를 졸업하였고, 1986년에 서울대학교 대학원 컴퓨터공학과에서 석사학위를, 1993년에 미국 State University of New York at Buffalo에서 컴퓨터학 박사학위를 각각 취득하였다. 1993년부터 1995년까지 한국전자통신연구원(ETRI)에서 선임연구원으로 재직하였으며, 1995년부터 현재까지 한양대학교 컴퓨터공학과 교수로 재직 중이다.

한국 정보과학회, 정보처리학회, 지능정보시스템학회, 인터넷정보학회, 미국 IEEE, ACM 등의 정회원이며, 관심분야는 웹지능, 텍스트마이닝, 정보검색/정보추출, 인공지능, 지능형 모바일정보시스템 등이다.