α-cut 선적용에 의한 시소러스 구축의 가속화에 관한 연구

Study on Acceleration of Building a Thesaurus by Means of Pre-applying of α-cut

김창민, 김용기
경상대학교 컴퓨터과학과 및 정보통신연구센터

Kim, Chang-Min and Kim, Yong-Gi
Dept. of Computer Science, Information and Communication Research Center
Kyungsang National University

요 약

퍼지 관계 개념을 응용한 퍼지 정보 검색은 형태론에 입각한 기존의 정보 검색과는 달리 문서와 용어의 의미론에 근거하는 정보 검색을 할 수 있다. 퍼지 정보 검색은 문헌의 집합, 용어의 집합으로 나누고 문헌과 용어의 관계성을 문서×용어의 퍼지 관계 행렬로 나타내며 퍼지 관계곱 연산을 이용하여 시소러스(thesaurus)를 형성하고 사용자로부터 주어진 질의에 적합한 문서를 제공한다. 그러나 이러한 퍼지 관계곱 연산은 매우 큰 시간 복잡도를 요구하는 연산이고 퍼지값은 부동소수점으로 표현해야하므로 대용량의 문서 시스템에 적용할 수 없어 비현실적이다. 부동소수점 연산은 연산속도가 느리고 저장공간도 많이 요구하므로 부동소수점 연산을 비트 연산으로 대체할 수 있다면 처리속도와 처리공간에 있어 성능 향상을 기대할 수 있다. 본 연구는 퍼지 정보 검색의 시소러스 형성에 있어 *α-cut* 적용의 시기를 조정하여 성능을 향상하는 방법을 제안한다.

I. 서론

퍼지 관계 개념은 퍼지 정보 검색(fuzzy information retrieval), 의료 진단(medical diagnosis), 근사 추론(approximate reasoning) 등 다양한 분야에 응용되고 있다[3]. 특히 퍼지 관계 개념을 응용한 퍼지 정보 검색은 형태론에 입각한 기존의 정보 검색과는 달리 문서와 용어의 의미론에 근거하는 정보 검색을 할 수 있다.

퍼지 정보 검색은 문헌의 집합, 용어의 집합으로 나누고 문헌과 용어의 관계성을 문서×용어의 퍼지 관계 행렬로 나타내며 퍼지 관계곱(fuzzy relational product) 연산을 이용하여 시소러스(thesaurus)를 형성하고, 이를 이용하여 사용자로부터 주어진 질의에 적절한 문헌을 검색하는 방법이다. 그러나 시소러스를 형성하기 위해 사용되어지는 퍼지 관계곱 연산은 매우 큰 시간 복잡도를 요구하는 연산이므로 대용량의 문서를 처리하기 위한 시스템에는 현실적이지 못하다는 단점을 가지고 있다.

본 논문에서는 퍼지 정보 검색 시스템에서의 시소러스 형성에 관하여 알아보고 이에 관한 연산을 수행하는 데 있어 수행성(performance)를 높일 수 있는 방법에 관하여 알아본다.

II. 퍼지 정보 검색

그림 1은 Kohout와 Bandler가 확장 불리언 모델로 제안한 퍼지 정보 모델[5,7]이다. 퍼지 검색 모델은 문서(documents), 시소러스(thesaurus), 퍼지검색요구(FS-request) 그리고 관계요구(R-request)로 구성된다. 퍼지검색요구는 질의어를 구성하여 이에 적합한 문서를 요구하는 것이고, 관계요구는 특정 용어에 관련있는 용어들에 관한 요구하는 것이다. 퍼지검색요구출력(FS-output)은 퍼지검색요구에 대한 결과로서 퍼지검색요구에 관계된 문서들의 참조들로 구성된다. 관계출력은 관계요구에 대한 결과로서 인덱스 용어들의 목록으로 구성된다[7].

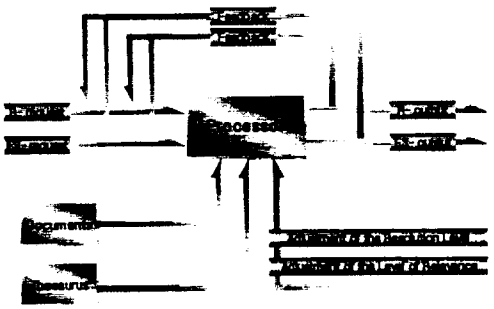

그림 1. 퍼지 정보 검색 모델

1. 퍼지 관계곱 연산

퍼지 정보 검색 기법의 이론적 바탕이 되는 퍼지 관계곱 연산은 Kohout와 Bandler에 의해 이진 관계곱이 확장된 것으로 용어의 계층 구조인 시소러스를 구축하는 데 편리하게 이용되어질 수 있다.

퍼지 집합론에서 '집합 A가 집합 B의 부분집합이다'는 것은 수식(1)와 같은 의미를 가진다.

$$\mu_A \le \mu_B \text{ 즉, } {}^{\forall}x \in U,\ \mu_A(x) \le \mu_B(x) \qquad (1)$$

'퍼지집합 A가 퍼지집합 B의 부분집합이다'에 대한 정도는 수식(2)과 같이 표현된다.

$$\frac{1}{|U|}\sum(\mu_A(x) \rightarrow \mu_B(x)) \qquad (2)$$

이는 '퍼지집합 A가 퍼지집합 B에 포함된다'의 평균등급을 나타내며 이진 집합(crisp set)에서의 최소치를 선택하는 것 보다 정보 검색의 개념에 적합하므로, 퍼지 정보 검색 기법에서는 이를 사용한다. 이때 퍼지 조건 연산자는 이진 논리의 조건 연산자를 포함하는 다른 형태의 처리방식을 가진다[1,4,6].

퍼지집합 A에서 퍼지집합 B로의 퍼지관계 R과 퍼지집합 B에서 퍼지집합 C로의 퍼지관계 S가 있을 때, R과 S의 퍼지 관계곱은 퍼지집합 A에서 퍼지집합 C의 관계를 표현하는 것으로서 다음 세 가지 퍼지 관계곱 연산이 있다.

$$(R \lhd S)_{ik} = \frac{1}{|B|}\sum(R_{ij} \rightarrow S_{jk}) \qquad (3)$$

$$(R \rhd S)_{ik} = \frac{1}{|B|}\sum(R_{ij} \leftarrow S_{jk}) \qquad (4)$$

$$(R \square S)_{ik} = \frac{1}{|B|}\sum(R_{ij} \leftrightarrow S_{jk}) \qquad (5)$$

$a_i \in A$, $c_k \in C$라 가정할 때, 수식(3) 퍼지 삼각 서브 논리곱(triangle sub-product relation) $(R \rhd S)_{ik}$는 a_i가 c_i를 포함하는 정도를 의미하고 수식(4) 퍼지 삼각 슈퍼 논리곱(triangle super-product relation) $(R \lhd S)_{ik}$는 a_i가 c_i에 포함되는 정도를 의미한다. 그리고 수식(5) 퍼지 사각 논리곱(square product relation) $(R \square S)_{ik}$는 a_i와 c_i가 유사한 정도를 의미한다[5,7]. 이러한 퍼지 관계곱 연산을 이용하여 퍼지 정보 검색 시스템의 시소러스를 구축할 수 있다.

2. 시소러스와 관계요구

관계요구[1,2,4]는 시소러스를 이용하여 주어진 용어에 대한 다른 용어들 간의 관계를 보여주는 것이다. 따라서 관계요구의 처리를 위해서는 시소러스의 구축이 선 처리되어야 한다.

어떤 퍼지 관계 R(R : 문서×용어)이 주어질 때, 퍼지 관계 R에 대한 용어들의 시소러스 형성은 R의 전치행렬 R^T과 R에 퍼지삼각서브논리곱 연산을 이용하여 결과 행렬을 구한 후 이에 α-cut을 적용하여 이진관계행렬을 만들고 이에 해세도식(Hasse diagram)을 적용하여 구한다.

3. α-cut의 적용

α-cut은 퍼지집합을 이진집합(crisp)으로 변환하는 방법 중의 하나이다. 어떤 퍼지 집합에 α-cut = c (단, $0.0 \le c \le 1.0$)를 적용하면 c 이상의 퍼지값을 가지는 구성원(member)는 1이 되

고 c 미만의 퍼지값을 가지는 구성원은 0이 된다.

4. 해세도식을 이용한 시소러스 생성

해세도식은 이진 관계 행렬 내에 존재하는 두 집합의 원소들 간의 관계성을 도식으로 보여주는 도구이다. 어떤 이진 관계 행렬이 주어질 때, 해세도식은 다음과 같은 4단계의 절차를 통하여 생성할 수 있다.

① 이진 관계 행렬로 표현.
② 재귀 순환을 가지는 모든 항을 제거.
③ 전치 가능한 라인을 제거
④ 상위 노드를 향하도록 모든 행을 재배치, 화살표를 제거.

5. 기존의 방법이 가지는 문제점

용어의 개수를 T_N이라 두고 문서의 개수를 D_N이라 두면, 시소러스 형성과 관련된 연산은 $D_N^2 \times T_N$에 비례하는 시간 복잡도를 갖게 된다. 퍼지값은 0과 1사이의 실수값이고 실제 디지털 컴퓨터에는 부동소수점으로 표현된다. 부동소수점은 구조가 복잡하여 처리속도가 느리고 큰 저장공간도 필요로 한다. 따라서 높은 시간 복잡도와 부동소수점 연산으로 이루어져 있는 퍼지 관계곱 연산의 성능은 매우 취약할 수밖에 없다.

III. α-cut 선적용

본 논문에서는 시소러스를 구축에 있어 기존의 방법이 가지고 있던 취약점을 개선하기 위해 퍼지 관계 연산에 앞서 α-cut을 적용하는 새로운 방법을 제안한다. 이 방법은 기존의 실수 연산을 논리 연산으로 바꿈으로써 기억공간을 절약하고 처리속도 향상을 가져온다.

제안하는 방법은 우선 퍼지입력행렬에 α-cut을 적용하여 이진행렬을 만든 후 관계곱 연산을 통하여 결과행렬을 산출하고 이에 해세도식을 적용하여 시소러스를 구축하는 것이다.

초기 퍼지입력행렬이 주어지면 α-cut을 적용하여 이진입력행렬로 바꾸고 이진관계행렬에 적용할 수 있는 관계곱 연산을 사용할 수 있게 된다. 본 논문의 관계곱 연산은 수식(6)(7)(8)과 같다.

$$(R \triangleleft S)_{ik} = \begin{cases} 0, & if \frac{1}{|B| \times A} \sum (R_{ij} \rightarrow S_{jk}) < 1 \\ 1, & otherwise \end{cases} \quad (6)$$

$$(R \triangleright S)_{ik} = \begin{cases} 0, & if \frac{1}{|B| \times A} \sum (R_{ij} \leftarrow S_{jk}) < 1 \\ 1, & otherwise \end{cases} \quad (7)$$

$$(R \square S)_{ik} = \begin{cases} 0, & if \frac{1}{|B| \times A} \sum (R_{ij} \leftrightarrow S_{jk}) < 1 \\ 1, & otherwise \end{cases} \quad (8)$$

A는 관계곱 연산을 적용하는 관계의 특성과 α-cut에 따라 연산을 적절하게 변화시키기 위한 보정치이다. 본 논문에서는 수식(9)을 이용하여 A의 값을 산출하였다.

$$A = \alpha_{cut} + (1 - \alpha_{cut})^2 \times 0.8 \times (1 - \frac{4.5}{\sqrt{|B|}}) \quad (9)$$

퍼지 조건연산자가 수 십여 가지의 방법[1,6]으로 구현됨과는 달리 이진 논리 조건연산자는 수식(10)(11)(12)와 같이 정의된다.

$$a \rightarrow b = \sim a \vee b \quad (10)$$
$$a \leftarrow b = a \vee \sim b \quad (11)$$
$$a \leftrightarrow b = (\sim a \vee b) \wedge (a \vee \sim b) \quad (12)$$

IV. 효율성 비교

퍼지 관계곱 혹은 관계곱 연산은 $D_N^2 \times T_N$에 비례하는 시간 복잡도를 갖는다. 따라서 퍼지값을 처리하기 위한 부동소수점 연산을 논리연산으로 바꾸면 매우 큰 성능향상을 기대할 수 있다. 표 3은 두 대의 워크스테이션에서 실험한 것으로써 퍼지조건연산자와 이진조건연산자의 초당 처리회수와 두 연산의 처리속도차이를 보여준다.

machine	퍼지조건 연산자^f	이진조건 연산자^b	처리속도비 (b/f)
A (64 bit)	0.476×10^7	0.457×10^{12}	0.960×10^5
B (32 bit)	0.149×10^7	0.604×10^{11}	0.405×10^4

* 처리단위 : Operations/Second
* 퍼지조건연산자^f : min(1,b/a)[1,4,6]
* 이진조건연산자^b : $\sim a \vee b$

표 3. 조건 연산자의 성능 비교

이진조건연산자는 퍼지조건연산자 보다 대략

305 배의 연산 처리량를 가지는 것으로 나타났다. 이는 논리 연산은 비교적 간단한 로직으로 구성되므로 복잡한 실수 연산보다 처리속도가 빠르고, 64 bit 컴퓨터의 경우는 동시에 64개 항에 대하여 연산을 적용하는 벡터처리가 이루어지기 때문이다.

용어의 개수가 10^4개인 시소러스를 형성한다고 가정할 때, 표 3의 결과를 토대로, 한계시간(10^4초, 2시간 46분 4초) 내에 처리할 수 있는 문서의 양을 비교해 보면 다음과 같다. 퍼지조건연산자^{f}는 0.476×10^{11}회의 연산을 처리하고, 이진조건연산자^{b}는 0.457×10^{16}회의 조건 연산자를 처리한다. 이에 시소러스 구축의 시간 복잡도는 $D_N^2\times T_N$이고 용어의 개수는 10^4이므로 퍼지조건연산자^{f}는 2.18×10^9 개의 문서를 처리할 수 있고 이진조건연산자^{b}는 6.76×10^5 개의 문서를 처리할 수 있다. 따라서 제안된 방법은 기존의 방법에 비해 약 310배 정도의 문서 처리량을 가진다.

V. 결론

관계요구는 용어와 용어의 관계를 구하는 연산인데 퍼지 관계곱 연산은 높은 시간 및 공간 복잡도를 가지고 퍼지값 처리를 위해서 부동소수점연산을 처리해야하므로 수십만 혹은 수백만 개의 문서와 수천 혹은 수만 개의 용어를 가지는 시스템에 적용하기는 거의 불가능하다. 본 논문은 이를 해결하기 위해 실수 연산을 이진 논리 연산으로 대체하기 위한 방법을 제안하였다. 통상적인 퍼지 관계곱 연산 후 α-cut을 적용하는 것이 아니라 α-cut을 먼저 적용하여 퍼지 관계를 이진 관계로 바꾼 후 이에 이진 관계곱을 적용하는 방법을 시도하였다.

제안된 방법은 기존의 방법에 비해 상당한 처리속도의 향상을 가져왔다. 그러나 처리된 결과의 신뢰성 측면에서 제안된 방법은 좋은 특성을 가지지 못할 것이 자명하다. 왜냐하면 α-cut을 적용한다는 것은 퍼지값이 의미하는 것을 왜곡하는 것으로써 시소러스 형성 초기에 α-cut을 적용하는 제안된 방법에 의한 결과행렬은 기존의 방법에 비해 신뢰도가 낮아지기 때문이다. 차후 이에 대한 깊이있는 연구가 요구된다.

참고문헌

[1] Bandler, W. and Kohout, L. J., "Fuzzy power sets and fuzzy implication operators", in Fuzzy sets and systems edited by Wang, P. P. and Chang, S. K., plenum press, 1980

[2] Bandler, W. and Kohout, L. J., "Fuzzy products as a tool for analysis and synthesis of the behaviour of complex natural and artificial systems", in Theory and Applications to Policy Analysis and Information Systems edited by Wang, P. P., Plenum Press, 1980

[3] Kerre, E. E., "A walk through fuzzy relations and their application to information retrieval, medical diagnosis and expert systems", Elservier Science Pub., 1992

[4] Kim, Yong-Gi and Kohout, L. J., "Comparison of Fuzzy Implication Operators by means of Weighting Strategy in Resolution Based Automated Reasoning", SAC '92 Proceedings, pages, ACM Symposium on Applied Computing Kansas City, MO, 1992

[5] Kohout, L. J. and Bandler, W., "Relational-Product Architectures for Information Processing", Information Science, Elsevier Science Publishing, 1985

[6] Kohout, L. J. and Bandler, W., "Semantics of implication operators and fuzzy relational products", International Journal of Man-Machine Studies 12, 1980

[7] Kohout, L. J., Keravnou, E. and Bandler, W., "Automatic Documentary Information Retrieval by means of Fuzzy Relational Products", in Fuzzy Sets in Decision Analysis by Gaines, B. R., Zadeh L. A. and Zimmermann, H. J., pages 308-404, North-Holland, Amsterdam, 1984