효과적 지식 관리를 위한 동적 폴더 구조

김 영 호*·이 상 진*·강 석 호*

《目 次》

Ⅰ. 서 론

최근 지식 경영에 대한 관심의 증가와 함께 지식 관리의 중요성이 강조되고 있다. 지식관리시스템은 조직의 인적 자원들이 개별적으로 축적하고 있는 지식을 체계화하여 이를 원활히 공유할 수 있게 하는 시스템적 접근 방법이다 [1]. 많은 기업에서 이미 전자우편, 그룹웨어, 문서관리시스템 등과 같은 정보 시스템을 활용하여 업무를 전자적으로 처리하고 있고, 그 처리 결과 또한 전자적으로 관리하고 있다. 이런 정보 시스템에서 생성되고 관리되는 문서들은 단순히 업무 내용뿐만 아니라 인적 자원이 보유한 비정형의 지식을 포함하고 있다. 지식관리시스템은 기존의 정보 시스템에 존재하는 다양한 문서들을 통합적으로 관리하고 효과적인 검색 기능을 제공하여 사용자들이 손쉽게 지식을 공유할 수 있는 환경을 제공한다.

지식관리시스템은 일반적으로 폴더(folder)라고 부르는 계층적 저장 체계를 이용하는데, 기존의 시스템은 대부분 하나의 관점으로 생성한 고정된 구조만을 지원한다. 이는 문서 접근에 일관성을 주는 편리한 점도 있지만, 한 번 생성된 계층 구조는 그 변경이 곤란하기 때문에 상황 변화에 따라 적절히 그 구조를 수정하는 것이 어렵다. 더욱이 이런 고정된 구조는 사용자의 필요에 따라 다를 수 있는 다양한 지식 체계를 수용하기가 근본적으로 불가능하다. 그리고 지식관리시스템이 제공하는 대부분의 검색 기능은 사용자의 질의를 만족하는 결과를 단순히 나열하는 일차원적 접근에 그치고 있다. 검색 결과를 사용자의 관점에 따라 다차원적으로 분류하여 제시할 수 있다면 사용자는 보다 효과적으로 필요한 지식을 획득할 수 있을 것이다.

* 서울대학교 산업공학과

본 논문에서는 기존 지식관리시스템의 정적 구조가 가지는 문제점을 해결하기 위해 동적 폴더 개념을 제시한다. 동적 폴더는 사용자의 필요에 따라 문서 체계를 동적으로 변화시킬 수 있는 구조를 말한다. 즉, 사용자의 요구를 반영하여 계층 구조를 만든다는 것이다. 개별 사용자의 요구 사항 또는 상황 변화에 따라 적절한 문서 저장 구조를 이용하므로 보다 효과적인 지식 관리와 공유가 가능하게 된다. 본 논문에서는 동적 폴더의 개념과 특성에 대해 설명하고, 이를 구현하기 위해 필요한 기술적 요소와 구현 방안을 제시하였다.

Ⅱ. 동적 폴더

기존의 폴더 구조에 대해 먼저 살펴보고, 동적 폴더의 개념을 기존의 폴더와 비교해서 설명하였다. 그리고 동적 폴더의 특성을 실제 사용 예와 함께 알아보았다.

2.1. 폴더 구조

기존의 지식관리시스템은 문서관리시스템 혹은 파일시스템과 마찬가지로 폴더 또는 디렉토리를 이용하여 문서를 저장하고 관리한다. 폴더는 트리 구조를 이용하여 문서를 계층적으로 저장하는 체계적 문서 관리 방법이다. 그러나 한번 정의된 폴더 구조는 변경이 어렵다는 정적인 특성을 가지므로 기존의 폴더를 정적 폴더라 할 수 있다. 이런 폴더를 실제로 사용할 때는 문서들을 특정 프로젝트 혹은 부서 단위로 사전에 지정된 폴더 구조에 따라 분류하여 저장한다.

문서수가 증가하거나 많은 수의 폴더가 존재할 경우 기존의 정적 폴더는 몇 가지 문제점이 있다. 즉, 사용자가 필요로 하는 문서를 찾기 위해서 폴더 구조와 각 폴더의 문서 포함 조건 - 해당 프로젝트 또는 해당 부서 - 을 정확히 파악하고 있어야 원하는 문서를 바로 찾을 수 있다. 그렇지 않으면 필요한 문서를 찾는 것이 매우 어려워진다. 이는 문서에 접근하기 위해 폴더의 이름을 사용한 접근만이 가능하기 때문에 발생한다. 정보의 양이 기하급수적으로 많아지는 오늘날에 이는 더욱 심각한 문제가 될 수 있다. 비록 검색 기능을 이용하여 필요한 문서를 찾을 수는 있지만 방대한 양의 파일 시스템 전부를 검색하는 것은 여전히 어려운 일이다 [7]. 기존의 정적 폴더를 이용하여 관점에 따라 달라지는 다양한 지식 체계를 지원하려면 관점의 수만큼 폴더 구조를 정의하고 이를 토대로 문서를 중복해서 저장하거나 별도의 인덱스 파일을 유지해야 하므로 저장 공간의 낭비 또는 복잡한 인덱스 파일 관리의 문제가 생긴다.

2.2. 문서 포함 조건

본 논문에서 제안하는 동적 폴더에서는 문서 집합을 정의할 때 문서가 가지는 여러 속성을 이용한다. 문서 포함 조건은 문서의 속성에 대한 조건이며, 이 속성은 크게 구조적 속성과 비구조적 속성으로 나눌 수 있다. 구조적 속성은 파일 생성일, 작성자, 종류, 작성 부서 등과 같이 컴퓨터에 의한 직접적 연산이 비교적 쉬운 속성으로, 단순 연산에 의해 분류가 가능하다. 예를 들면, 작성일을 기준으로 폴더를 생성할 경우, 특정 기간별로 폴더가 생성되어 문서 집합이 나누어진다.

비구조적 속성은 문서의 주제나 내용과 같은 속성을 말한다. 이런 속성들은 컴퓨터의 단순 연산으로 분류하는 것이 어렵다. 특히, 문서 주제나 내용의 관련성 유무는 '예' 또는 '아니오'로 답할 수 있는 것이라기보다 관련성의 정도를 따져야 하는 퍼지(fuzzy)한 성질을 지닌다는 점과, 여러 주제간의 관련성이 서로 영향을 미쳐 간접적으로 파생되어 생기는 관련성도 고려해야 한다는 점에서 어려움이 생긴다.

본 논문에서는 비구조적 속성을 문서 분류에 이용하기 위해 컴퓨터가 연산할 수 있는 정량적 관련성 계산 방법을 제시하였다. 이는 기존의 정보 검색 분야의 연구 결과를 개선한 것이다. 본 연구에서 비구조적 속성은 문서의 주제에 한정하기로 하고, 이 주제는 사용자가 시스템에 새로운 문서를 등록할 때에 입력하여야 한다. 문서 주제별 관련성을 퍼지 관계를 통해 정의하도록 하였고, 주제어 사이의 관련성을 정의한 개념 네트워크를 이용하여 직접적인 관련성 뿐 아니라 간접적인 관련성까지 고려한 문서 분류를 가능하도록 하였다. 문서 주제의 관련성에 대한 연산은 3장에서 자세히 설명한다.

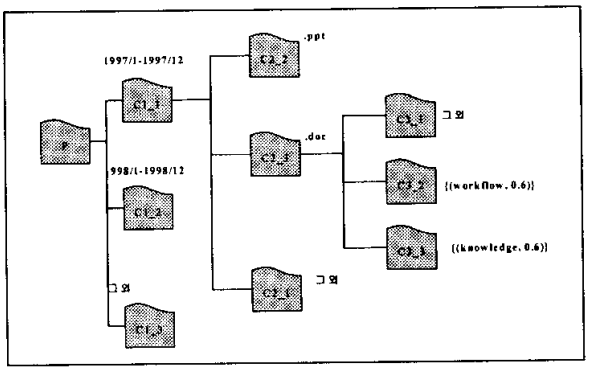
2.3. 동적 폴더

동적 폴더는 전술한 문서 포함 조건에 따라 포함되는 문서 집합을 동적으로 변경할 수 있는 폴더를 말한다. 즉, 문서 포함 조건을 만족하는 문서가 해당 폴더에 능동적으로 포함되기 때문에, 사용자가 자신의 관점을 시스템에 입력하여 폴더의 구조를 동적으로 생성할 수 있다. [그림 1]은 사용자의 관점에 의해 문서를 분류한 예이다.

[그림 1]은 연도, 문서 종류, 문서 주제를 이용해 문서를 분류한 예이다. 이 예는 최상위 수준의 문서 집합을 일차적으로 연도별('1997년,' '1998년,' '그 외')로 분류하고 있다. 그리고 같은 방식으로 다른 기준을 이용하여 문서 집합을 세분해가고 있음을 보여주고 있다. 이 예에서 재미있는 것은 '.doc' 문서 집합을 문서 주제와의 관련성을 고려하여 다시 분류하고 있다는 것이다. 이때, 최하위 폴더 가운데 하나인 'C3-2'는 1997년 1월부터 1997년 12월

사이에 작성된 '.doc' 문서 중 문서 주제 workflow와의 관련성이 0.6 이상인 문서만을 포함하게 된다.

〈그림 1〉 동적 폴더를 이용한 문서 분류 예



동적 폴더는 사용자가 폴더 생성 혹은 삭제를 통해 문서 집합을 세분화 혹은 일반화할 수 있다는 점에서 기존의 폴더와 다르다. 즉, 'C3-2'와 'C3-3' 폴더를 삭제하면 이 폴더에 포함되어 있던 문서들이 상위의 'C2-3' 폴더로 이동된다. 또, 'C2-2' 폴더의 하위에 새로 폴더를 추가하고 이 폴더의 문서 포함 조건을 workflow와의 관련성이 0.6 이상인 문서로 설정하면 'C2-2' 폴더의 문서 집합은 workflow와의 관련성에 의해 세분된다. 이러한 기능을 이용하여 사용자는 자신의 관점에 따라 문서 집합을 체계적으로 재구성할 수 있다.

기존 폴더와의 또 다른 차이점으로 문서 포함 조건을 변경하여 문서 분류를 동적으로 관리할 수 있다. 즉, 폴더 'C3-2'의 문서 포함 조건이 workflow와의 관련성이 0.6 이상인 것을 0.8로 변경할 경우 좀 더 작은 문서 집합으로 변경된다. 반대로 관련성을 0.2로 할 경우 문서 집합이 확대된다.

2.4. 동적 폴더의 특징

기존의 문서관리시스템에서는 이미 존재하는 폴더에 새로운 문서를 등록하는 방식으로 사용된다. 따라서, 폴더 구조의 변경 - 새로운 폴더의 추가나 기존 폴더의 삭제 - 이 발생하면 변화된 구조에 맞도록 사용자가 해당되는 모든 문서들을 재배치해야만 한다. 반면에 동적 폴더에서는 문서 집합의 자동 재분류가 가능하다. 즉, 폴더 구조가 변하면 문서 포함 조건에 맞는 문서가 해당 폴더에 자동으로 포함되게 된다. 이같이 문서 집합을 체계적으로 재구성할 수 있는 동적 폴더의 특성을 그 구조적인 면과 내용적인 면으로 나누어 살펴보면 다음과 같다.

구조적인 면에서의 동적 특성은 다음과 같다. 동적 폴더는 폴더 구조 변화를 통한 세밀도(granularity) 조정이 가능하다. 즉, 폴더의 삭제나 새로운 하위 폴더 생성과 같은 폴더 계층 구조의 변화에 따라서 문서 분류의 기준이 변화하기 때문에, 더 포괄적인 혹은 더 세밀한 기준에 의한 문서의 재분류가 가능하다. 전 절의 예에서와 같이 사용자는 폴더 구조의 변경을 통해 관심 있는 문서 집합을 세분화하거나 혹은 일반화할 수 있다는 점에서 동적 폴더는 구조적인 측면에서 동적 특성을 지닌다.

내용적인 면에서 동적 특성은 단일 폴더의 문서 포함 조건을 변경할 때 나타난다. 각각의 폴더는 자신의 문서 포함 조건을 가지고 있으며, 이 조건은 고정된 것이 아니라 변경이 가능하기 때문에 폴더에 포함되는 문서 집합에 변화를 줄 수 있다. 문서 생성일을 문서 포함 조건으로 사용하는 예를 생각해보자. 조건이 1997년 1월부터 12월까지에서 1997년 1월부터 1998년 12월까지로 변경되면, 폴더에 포함되는 문서 집합이 확대된다. 이처럼 문서 포함 조건의 변경을 통해 폴더에 포함되는 문서 집합을 확장 또는 축소할 수 있다.

이러한 동적 폴더의 특징을 이용하여 모든 사용자는 자신이 관심 있는 문서에 대한 조건을 정량화 하여 이를 문서 포함 조건으로 나타냄으로써 조직 내의 문서를 자신의 관점을 반영하여 체계적으로 관리할 수 있다. 조직 전체의 입장에서는 하나의 문서 집합을 가고 있지만 개별 사용자들은 각각 자신의 관점에 따라 다른 방식으로 정리된 폴더를 사용할 수 있게 된다.

Ⅲ. 문서 주제 관련성

전술한 동적 폴더에서 흥미로운 것은 문서와 주제 사이의 관련성에 따라 문서를 분류하는 것이다. 여기서는 문서 분류에 있어 주제와의 관련성을 계산하고 이를 조직화 하는 개념 네트워크(concept network) 방법론에 대해 설명한다.

3.1. 개념 네트워크

개념 네트워크[8]는 조직에서 관리하는 중요 주제어들을 네트워크 형식으로 표현한 것이다. 이 개념 네트워크에 퍼지 관계(fuzzy relation)를 도입하여 주제간의 관련성과 문서와 주제 사이의 관련성을 나타내는 시도가 [4]과 [5] 등에 의해 있었다. 여기서 개념(concept)은 사용자 혹은 시스템이 정의한 문서의 주제어(keyword)를 말한다.

사용자는 문서를 등록할 때 문서의 주제를 정의할 수 있다. 그러면 시스템은 정해진 방법으로 문서와 각 주제(개념)와의 관련성을 정의한다. 개념 i와 개념 j의 관련성 그리고 개념 j

와 개념 k의 관련성이 설정되면, 개념 k와 개념 i의 관련성을 이행 속성(transitivity)으로 구할 수 있다. 이는 개념 사이의 직접적인 관련성을 이용하여 개념 사이의 간접적인 관련성을 구할 수 있음을 의미한다. 모든 개념 사이의 직접적 관련성을 나타내는 개념 행렬을 M이라 하면, M의 이행폐쇄행렬(transitive closure matrix) T를 구함으로써 개념 사이의 간접적 관련성까지 고려한 개념 행렬을 구할 수 있다.

앞서 설명한 퍼지 정보 검색에서는 개념 네트워크가 이미 존재한다는 가정에서 출발하고 있다. 그러나, 실제 기업에서 사용되는 주제어(개념)는 그 수가 상당히 많으므로, 사용자 혹은 관리자가 개념 네트워크를 직접 생성하고 관리하는 것은 현실적으로 매우 어렵다. 따라서, 시스템에서 자동으로 개념 네트워크를 생성할 필요가 있으며, 본 연구에서는 단어의 발생 빈도를 이용하는 기존의 방법을 개선하여 개념 네트워크를 자동으로 생성한다.

3.2. 주제 관련성

개념 네트워크의 자동 생성을 위해서는 유사어 사전(thesauri)에 관한 연구[8]를 이용할 수 있다. 유사어 사전은 사용자 질의(query)를 확장하여 보다 효과적인 검색을 가능하게 하는 방법으로 제시되었으며, 이를 개념 사이의 관련성을 계산하는 데에 적용할 수 있다. 유사어 사전을 생성하는 방법은 단어의 발생 빈도(term co-occurrence)를 이용하는 것이 일반적이다.

이 빈도를 이용하는 방법은 문서 내 단어들 사이의 관련성 계산을 위해 각 단어가 문서에서 동시에 나타나는 빈도(co-occurrence)를 사용한다. 두 단어가 같은 문서에 발생하는 횟수가 많으면 두 단어 사이의 관련성이 높다고 추정하는 것이 이 방법의 기본 가정이다. 이 방법은 관련성의 대칭성 여부에 따라 대칭적(symmetric) 방법과 비대칭적(asymmetric) 방법으로 구분된다 [2, 3]. 전자는 단어 j에 대한 단어 k의 관련성과 단어 k에 대한 단어 j의 관련성이 동일하게 계산되는데 대표적으로 cosine 알고리즘이 있다. 후자는 이 두 관련성이 서로 다른 값으로 계산되는 것으로cluster 알고리즘이 있다. 일반적으로 후자가 전자보다 보다 정확한 관련성을 계산하는 것으로 알려져 있으므로, 본 연구에서는 cluster 알고리즘을 바탕으로 이를 더 개선하는 방안을 제안한다.

Cluster 알고리즘은 다음과 같다. 단어 j (Tj)와 단어 k (Tk)의 관련성은 다음의 식 (1)로 계산한다. 문서 i에서 단어 j의 발생 여부를 d_{ij}라 할 때 (1: 발생, 0: 발생하지 않음), 단어 j에 대한 단어 k의 관련성은 단어 j, k가 동시에 발생하는 문서의 개수를 단어 j가 발생하는 문서의 개수로 나누어 계산하며, 단어 k에 대한 단어 j의 관련성은 단어 j, k가 동시에 발

생하는 문서의 개수를 단어 k가 발생하는 문서의 개수로 나누어 구한다.

$$\text{Weight}(Tj, Tk) = \frac{\sum_{i=1}^{n} d_{ij} \times d_{ik}}{\sum_{i=1}^{n} d_{ij}}$$
$$\text{Weight}(Tk, Tj) = \frac{\sum_{i=1}^{n} d_{ij} \times d_{ik}}{\sum_{i=1}^{n} d_{ik}} \quad (1)$$

이 식은 두 단어가 동시에 발생하는 문서의 개수가 많으면 두 단어 사이의 관련성을 높게 평가한다.

3.3. 빈도-거리 관련성

전절에서 소개한 단어 발생 빈도를 이용하는 방법은 단순히 단어의 발생 빈도만을 이용한다. 이 방법은 단어 사이의 의미적 관련성을 측정하는데 한계가 있으므로 본 연구에서 필요로 하는 개념 사이의 관련성을 계산하는 방법으로 사용하기에는 부족하다.

따라서, 본 연구에서는 단어 사이의 거리를 이용하여 단어의 의미적 관련성을 고려한 개념 네트워크 자동 생성 방법론을 제안한다. 단어 사이의 발생 거리를 고려한다는 것은 같은 문서 내에서 두 단어가 동시에 발생하더라도 두 단어가 같은 단락에 위치하고 있는지, 같은 문장에 위치하고 있는지, 혹은 같은 문장의 인접 위치에 있는지에 따라 다른 가중치를 부여한다는 것을 의미한다. 각 단어들의 빈도를 구할 때, 기존의 알고리즘에서는 문서 내에 발생하는 횟수를 그대로 이용하였다. 이 기존의 방법에 각 단어 사이의 발생 거리에 대한 〈표 1〉과 같이 가중치를 적용하여 의미적 관련성을 반영하였다.

〈표 1〉 단어간 발생 거리에 따른 가중치

	문서	단락	문장	인접	가중치
관련성이 낮다	×	×	×	×	0
↑	○	×	×	×	$\alpha 1$
	○	○	×	×	$\alpha 2$
↓	○	○	○	×	$\alpha 3$
관련성이 높다	○	○	○	○	$\alpha 4$

$(\alpha 4 \geq \alpha 3 \geq \alpha 2 \geq \alpha 1 \geq 0; \sum_{i=1}^{4} \alpha_i = 1)$

즉, 발생 빈도가 동일하게 높지만 두 단어 사이의 거리가 멀다면 이는 의미적 관련성이 낮다고 말할 수 있다. 또한 발생 빈도는 낮지만 두 단어 사이의 거리가 가깝다면 이 두 단어는 의미적 관련성이 높다고 추정된다. 기존의 cluster 알고리즘이 같은 문서에 발생하는 단어의 개수를 이용한 단순한 방법인 것에 반해, 제안하는 방법은 동일하게 같은 문서에서 발생하지만, 같은 단락에서 발생하는지, 같은 문장에서 발생하는지, 혹은 인접한 위치에서 발생하는지가 의미적 관련성을 발견하는 데 영향을 미친다는 것을 반영한 방법이다. 〈표 1〉의 가중치를 이용하여 기존의 cluster 알고리즘을 식 (2)와 같이 수정하여 사용한다.

$$\text{Weight}(Tj, Tk) = \frac{\sum_{i=1}^{n} tf'_{ijk}}{\sum_{i=1}^{n} tf_{ij}} \times T_{jk} \qquad (2)$$

$$tf'_{ijk} = \alpha 1 * tfd_{ijk} + \alpha 2 * tfp_{ijk} + \alpha 3 * tfs_{ijk} + \alpha 4 * tfa_{ijk}$$

tfa_{ijk} : 단어 j와 k가 문서 i에서 인접 위치에 발생한 횟수

tfs_{ijk} : 단어 j와 k가 문서 i에서 인접하지 않지만 같은 문장에서 발생한 횟수

tfp_{ijk} : 단어 j와 k가 문서 i에서 다른 문장이지만 같은 문단에서 발생한 횟수

tfd_{ijk} : 단어 j와 k가 문서 i에서 서로 다른 문단이지만 동시에 발생한 횟수

tf_{ij} : 문서 i에서 단어 j의 발생 횟수

3.4. 알고리즘 적용 예

전절에서 제안한 단어간 관련성 계산 방법을 〈그림 2〉의 예제에 적용하였다. 이 예는 논문 초록으로, 1개의 단락, 10개의 문장, 132개의 단어들로 구성되어 있다.

〔그림 2〕 실험 예제 1

InfoFlow: A Web-based Workflow Management System

In this paper, we introduce a web-based workflow management system implemented using pure java technology. The goal of the developed system is to manage various business processes occurring in the CITIS (Contractor Integrated Technical Information Services) environment. The developed system is composed of Process Designer, Workflow Engine, and Client programs. The Process Designer is a module that provides the environment for the build-time function, which generates the specification of a process. The Process Designer Module for the modeling of business processes presents the capability of defining nested process models. Since the Workflow Engine is developed using the java language, it can be implemented on any platform. The Client programs can be accessed via the WWW interface, which indicates that there is no need to install any client programs at the client-sides. Users who are connected to the internet with web browsers, such as Internet Explorer and Netscape Navigator, can utilize the normal client program, monitoring client program, and system administration client program based on their access privileges. Communications between the workflow engine and the clients are implemented using the java servlet mechanism. The workflow system can be used as a process management tool in CALS and CITIS environments.

이 예제에서 많이 발생하는 단어는 〈표 2〉의 14개이다. 기존의 cluster 알고리즘과 수정된 cluster 알고리즘으로 단어간의 관련성을 계산하면 〈표 3〉의 결과를 얻을 수 있다.

〈표 2〉 실험 예제 1의 단어 발생 횟수

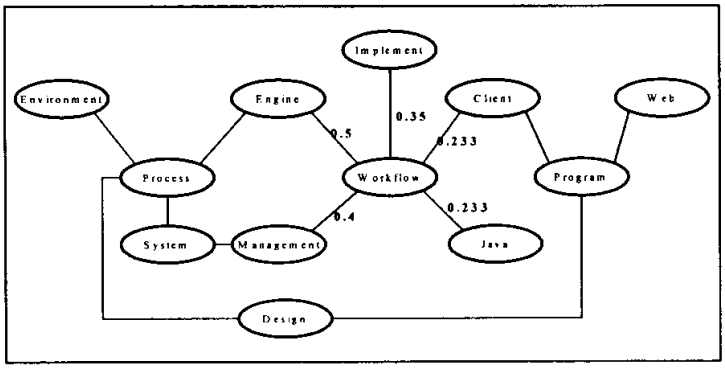
단어	발생 횟수	단어	발생 횟수
Client	8	Develop	3
Process	8	Engine	3
Program	6	Environment	3
System	6	Implement	3
Workflow	6	Java	3
Base	3	Management	3
Designer	3	Web	3

〈표 3〉 실험 예제 1의 관련성 계산- 'client'와 다른 단어들과의 관련성

관련단어	기존의 cluster 알고리즘	수정된 cluster 알고리즘
Process	1	0.0875
Program	0.75	0.375
System	0.75	0.0875
Workflow	0.375	0.175
Designer	0.375	0.0875
Engine	0.375	0.0875

($\alpha 1=0.1$, $\alpha 2=0.2$, $\alpha=0.3$, $\alpha 4=0.4$)

〈그림 3〉 개념 네트워크의 생성 예



기존의 방법을 사용했을 경우 client와 process의 관련성이 가장 높다. 그러나 예제의 내

용을 살펴보면 client는 program과 가장 관련성이 높다. 제안한 방법 식 (4)는 이 것과 일치하는 결과를 제공한다. 즉, 수정된 cluster 알고리즘은 두 단어 사이의 거리의 가까운 정도에 따라 가중치를 부여하였기 때문에 두 단어 사이의 의미적 관련성을 기존의 방법에 비하여 정확히 추정할 수 있음을 알 수 있다. 〈그림 3〉은 수정된 cluster 알고리즘에 의해 생성된 개념 네트워크의 예이다.

Ⅳ. 실험 결과

단어 사이의 발생 거리를 이용한 개념 네트워크 자동 생성 알고리즘의 유의성 여부를 다음과 같은 실험을 통하여 검증하였다.

4.1. 실험 설계

실험 대상으로 IEEE Knowledge and Data Engineering에 게재된 논문 초록을 이용하였으며, 1995년부터 1999년까지 총 350여 편의 논문 초록 중 10 가지 주제의 문서 149개를 선택하여 실험하였다. 아래 〔표 4〕와 같이 모두 네 가지 예제 집합을 가지고 실험을 실시하였다. 1번 예제 집합의 초록은 모두 하나의 단락만으로 구성되어 있다. 2번과 3번 예제 집합은 각각 단락이 2개와 4개인 문서들이다. 4번 예제 집합은 단락의 수가 그 이상인 문서들이다.

〈표 4〉 실험 예제의 설정

번호	예제 크기(개수)	한 문서에 포함된 초록 수
1	149	1
2	74	2
3	32	4
4	10	

실험 예제를 이와 같이 설정한 것은 제안한 알고리즘이 단어 발생의 상대적 위치에 따라 어떤 효과가 있는지 알아보기 위해서 이다.

4.2. 실험 결과 및 분석

실험 결과 생성된 인덱스(index)는 총 1,518개의 단어로 구성되어 있다. 발생 빈도가 상대적으로 높은 20개를 선택하여 R-norm 값을 비교하였다. R-norm은 전문가가 제시한 관련성의 우선 순위와 시스템이 제시하는 관련성의 우선 순위를 비교하여 시스템의 정확성을 평가하는 방법으로 다음 식 (3)과 같이 계산한다 [9]. 이는 0과 1사이의 값을 가지며, 1에 가까울수록 더 정확한 값을 가지는 것으로 평가된다.

$$R_{norm}(\Delta^{sys}) = \frac{1}{2}\left(1 + \frac{S^{+} - S^{-}}{S^{+}_{\max}}\right) \tag{3}$$

Δ^{usr} : 전문가(평가자)가 제시하는 관련성의 우선 순위

Δ^{sys} : 시스템이 제시하는 관련성의 우선 순위

S^{+} : 관련성이 높은 단어가 더 높게 평가된 개수

S^{-} : 관련성이 높은 단어가 더 낮게 평가된 개수

S^{+}_{max} : S+가 발생 가능한 최대 개수

관련성을 평가하는 전문가는 실험 대상 문서 전체의 내용을 충분히 이해하고 난 뒤, 선택된 20개의 단어들 각각 마다 관련성이 높은 단어 5개씩 제시한다. 제시된 단어 집합은 개선된 알고리즘과 기존 알고리즘에 의해 생성되는 단어 집합과 각각 비교되어 R-norm 값을 계산한다. 〈표 5〉는 계산된 R-norm 값을 비교하고 있다.

〈표 5〉 실험 결과 요약

실험 예제	기존의 알고리즘	개선된 알고리즘
1	0.61	0.62
2	0.56	0.57
3	0.53	0.58
4	0.69	0.76

위의 실험 결과는 본 연구에서 새롭게 제시한 단어 사이의 발생 거리를 이용한 개념 네트워크 자동 생성 알고리즘이 기존의 알고리즘보다 더욱 정확하게 개념 네트워크를 생성한다는

것을 보여준다. 이는 단어 사이의 발생 거리를 이용한 것이 의미적 관련성을 반영한다고 볼 수 있다. 두 번째로, 실험 예제의 크기가 작을수록 - 문서의 단락 수가 증가할수록 - 개선된 알고리즘의 R-norm 값과 기존 알고리즘의 R-norm 값의 차이가 커짐을 알 수 있다. 이는 개선된 알고리즘이 단어 발생의 상대적 위치- 단락의 위치 - 를 사용함과 동시에 문서의 개수가 작아지기 때문으로 추정된다. 세 번째로, 실험 예제의 크기가 큰 경우에는 - 문서의 단락 수가 1인 경우 - 개선된 알고리즘의 R-norm 값과 기존의 알고리즘의 R-norm 값의 차이가 작음을 알 수 있다. 이는 개선된 알고리즘이 사용하는 단어 발생의 상대적 위치- 단락의 위치 - 를 사용할 수 없기 때문에 개선된 알고리즘과 기존의 알고리즘의 차이가 없어지기 때문으로 추측된다.

V. 결 론

본 연구에서는 효과적 지식 공유를 위해 동적 폴더라는 새로운 개념을 제시하고 그 구현 방안을 제안했다. 동적 폴더는 사용자의 관점과 관심도에 따라 문서 포함 속성을 동적으로 변경하는 것을 가능하게 한다. 또, 개념 네트워크를 이용함으로써 문서 내용에 의한 분류를 가능하게 하였고, 문서와 주제어 사이의 직접적 관련성 뿐 아니라 간접적 관련성까지 고려가 가능하다. 정확한 개념 네트워크를 생성하기 위해 단어 사이의 거리를 고려한 개념 네트워크 자동 생성론을 제시하였고, 단어 사이의 거리를 고려하여 좀 더 정확한 단어의 관련성을 구할 수 있음을 보였다.

동적 폴더는 지식관리시스템의 궁극적인 목적인 지식 공유를 보다 효과적으로 지원하는 도구가 될 것이다. 뿐만 아니라 여러 사용자가 나름대로의 문서 구조를 가지지만 실제 문서는 한 번만 저장되므로 효과 대비 저장 효율도 매우 높일 수 있다. 한편, 수정된 cluster 알고리즘의 최적화, 적절한 가중치에 대한 튜닝, 반자동 및 자동 분류 지원에 대해서는 추가적인 연구가 필요하다.

참 고 문 헌

[1] M. Alavi and D. Leidner, "Knowledge Management Systems: Emerging Views and Practices from the Field," In IEEE Proceedings of the 32nd Hawaii International Conference on System Sciences, pp. 1-11, 1999.

[2] H. Chen and K. J. Lynch, Automatic Construction of Networks of concepts characterizing document database, IEEE Transactions on Systems, Man and Cybernetics, Vol. 22, pp. 885-902, 1992.

[3] H. Chen, T. Yim, D. Fye, and B. Schatz, Automatic Thesaurus Generation for an Electronic Community System, Journal of the American Society for Information Science, Vol. 46, No. 3, pp. 175-193,1995.

[4] S.-M. Chen and J.-Y. Wang, Document Retrieval Using Knowledge-Based Fuzzy Information Retrieval Techniques, Transactions on Systems, Man, and Cybernetics, Vol. 25, No. 5, May 1995.

[5] S.-M. Chen and J.-Y. Wang, Fuzzy Query Processing for Document Retrieval Based on Extended Fuzzy Concept Network, Transactions on Systems, Man, and Cybernetics, Vol. 29, No. 1, Feb. 1999.

[6] D. K. Gifford, P. Jouvelot, M. A. Sheldon, and J. W. O'Toole, Semantic File Systems, In ACM SIGOPS 91, pp. 16-25, 1991.

[7] B. Gopal and U. Manber, Integrating Content-Based Access Mechanisms with Hierarchical File Systems, In ACM OSDI, pp. 265-278, 1999.

[8] D. A. Grossman and O. Frieder, Information Retrieval: Algorithms and Heuristics, Kluwer Academic Publishers, 1998.

[9] V. N. Gudivada and V. V. Raghavan, ACM Transaction on Information Systems, Vol. 13, No. 2, pp. 115-144, 1995.

[10] 이상진, 효과적 지식 공유를 위한 동적 폴더의 구현, 서울대학교 공학석사학위논문, 2000.