유사도 측정 기법을 이용한 효율적인 요구 분석 지원 시스템의 구현

(Implementation of an Efficient Requirements Analysis supporting System using Similarity Measure Techniques)

김 학 수 † 고 영 중 †† 박 수 용 ††† 서 정 연 ††††
(Harksoo Kim) (Youngjoong Ko) (Sooyong Park) (Jungyun Seo)

요 약 소프트웨어가 점점 복잡해지고 대형화됨에 따라서 사용자의 요구가 매우 다양해지고 있으며, 제품에 대한 기대 수준도 높아지고 있다. 그러므로, 사용자의 요구 사항을 정확히 분석하여 효과적으로 개발 단계에 적용하는 것은 매우 중요하다. 본 논문에서는 자연어로 표현되는 요구 사항 문서의 분석 시에 나타나는 오류를 효과적으로 줄이고, 수정하는데 사용될 수 있는 요구 분석 시스템을 제안한다. 제안된 시스템은 문서간 유사도 측정에 의해서 문서간의 의존성(*dependency*) 분석을 지원하고 문장간 유사도 측정에 의해서 요구 사항간의 연계성(traceability), 중복성(redundancy), 불일치성(inconsistency), 그리고 불완전성(imcompleteness)을 발견하는 것을 지원한다. 또한 모호한 문장을 추출하여 요구사항의 불명확성(ambiguity)을 발견하는 기능도 제공한다. 문서간 유사도 측정을 위해서 사용된 색인 방법은 슬라이딩 윈도우 모델과 의존 구조 모델을 결합한 것으로 각 모델이 가지는 단점을 효과적으로 보완할 수 있다. 본 논문에서는 문서간, 문장간 유사도 측정 기법의 효율성을 실험을 통해 검증하였으며 구현된 시스템을 통해 분석 처리되는 과정을 보여주고 있다.

Abstract As software becomes more complicated and large-scaled, user's demands become more varied and his expectation levels about software products are raised. Therefore it is very important that a software engineer analyzes user's requirements precisely and applies it effectively in the development step. This paper presents a requirements analysis system that reduces and revises errors of requirements specifications analysis effectively. As this system measures the similarity among *requirements documents and sentences*, it assists users in analyzing the dependency among requirements specifications and finding the traceability, redundancy, inconsistency and incompleteness among requirements sentences. It also extracts sentences that contain ambiguous words. Indexing method for the similarity measurement combines sliding window model and dependency structure model. This method can complement each model's weeknesses. This paper verifies the efficiency of similarity measure techniques through experiments and presents a proccess of the requirements specifications analysis using the embodied system.

· 본 연구는 교육부의 BK21 사업 핵심연구과제를 통한 지원으로 이루어 진 것입니다.
† 학생회원 : 서강대학교 컴퓨터학과
hskim@nlpzodiac.sogang.ac.kr
†† 비 회 원 : 서강대학교 컴퓨터학과
kyj@nlpzodiac.sogang.ac.kr
††† 정 회 원 : 서강대학교 컴퓨터학과 교수
sypark@ccs.sogang.ac.kr
†††† 종신회원 : 서강대학교 컴퓨터학과 교수
seojy@ccs.sogang.ac.kr
논문접수 : 1998년 4월 6일
심사완료 : 1999년 11월 6일

1. 서 론

소프트웨어(software)가 점점 복잡해지고 대형화됨에 따라서 사용자의 요구가 매우 다양해지고 있으며, 제품에 대한 기대 수준도 높지고 있다. 그러므로, 사용자의 요구 사항을 정확히 분석하여 효과적으로 개발 단계에 적용하는 것은 매우 중요하다. 사용자의 요구 사항을 정리하고 분석하는 것을 요구 분석(requirement analysis)이라고 하며, 전체 소프트웨어 공정에서 매우 중요한 위치를 차지한다. 요구 분석이 이처럼 중요하게 다루어지는 이유는 다음과 같다. 첫째, 소프트웨어 초기 단계에

서 발견되지 않은 오류(error)는 개발의 마지막 단계까지 영향을 미쳐서 많은 오류를 발생시키는 원인이 되고 있다. 둘째, 소프트웨어 개발은 요구사항으로부터 시작되므로 요구사항들의 잘못된 분석과 사용자나 개발자들 사이의 잘못된 이해가 시스템 전체에 대한 개발 실패의 원인이 될 수 있다. 셋째, 요구분석단계에서나 혹은 개발의 초기 단계의 오류 수정 비용이 개발 말기의 비용보다 훨씬 적게 든다. 넷째, 소프트웨어 자체가 매우 복잡해지고 대형화됨에 따라 요구사항에 대한 개발자들의 이해와 관리가 매우 어려워서 많은 오류가 발생할 가능성이 매우 높아지고 있다[2].

요구 분석 단계에서 가장 자주 나타나는 문제들은 부정확성(imprecision), 다른 요구 사항과의 충돌(conflict), 불일치성(inconsistency), 요구 사항 자체의 불명확성(ambiguity)과 불완전성(incompleteness) 등을 들 수 있다. 요구 분석 단계에 이러한 문제점들을 많이 내포되는 이유는 이것들이 대부분 자연어(natural language) 문서로 이루어지며, 요구 분석가의 특징에 따라 다양하게 서술될 수 있다는 것이다. 이러한 요구 분석 시 나타나 문제점들을 보완, 해결하기 위한 연구들이 있어 왔으며, 그 중 하나가 자연어 정보 검색 기술을 응용한 것이다[1]. 소프트웨어 개발 시 문서화 작업은 요구 분석 단계에서 뿐만이 아니라 개발 작업 전체에서 매우 중요하다. 즉, 문서화 작업은 개발 단계 전체에 걸쳐서 수행되며, 각 단계에서의 산출물이나 개발 절차를 명시한 문서, 사용자 지침서 등이 모두 문서화 작업을 통해서 이루어진다. 그러므로, 개발 시에 발생하는 문서들의 일관성을 유지하고 관리하는 것은 매우 중요하다. 효율적인 요구 사항 분석 작업과 요구분석서의 유지 보수를 위해 요구 분석을 지원하는 자동화된 도구의 개발이 필요하며, 이러한 도구는 요구사항 분석과 전체 프로젝트(project) 수행을 위해 좋은 기초를 제공해 줄 수 있을 것이다. 본 논문에서는 문서간의 의존성(dependency)과 문장간의 연계성(traceability) 분석을 통해 요구 분석 시 나타나는 요구 사항의 중복성(redundancy), 불일치성, 불명확성 그리고 불완전성 등의 문제점들을 효율적으로 관리할 수 있는 요구 분석 시스템을 제안한다. 제안된 요구 분석 시스템은 정보 검색 기술의 유사도(similarity) 검사 기법을 기반으로 한다.

본 논문의 구성은 다음과 같다. 먼저, 2장에서 관련된 연구들을 설명한다. 그리고, 3장에서 시스템의 구조를 간략하게 보이고, 유사도 측정 기법에 대해 자세히 설명한다. 4장에서는 실험을 통하여 유사도 측정의 정확도와 효율성을 평가한다. 5장에서는 앞장에서 제시한 유사도 측정 기법을 이용한 요구분석 시스템의 구현 및 결과 분석을 한다. 마지막으로 6장에서 결론을 내리고, 향후 연구 과제를 제시한다.

2. 관련 연구

소프트웨어 요구 분석서를 자동으로 색인하고 분류하는 기존의 연구들은 주로 문서에서 추출된 특정한 키워드와 이것들 사이의 간단한 유사도 측정 방법을 이용하였다. Palmer는 [2]에서 2층 구조의 TTC(two-tired clustering) 알고리즘을 이용하여 요구 분석서를 색인하고 클러스터링(clustering)하는 방법을 제안하였다. TTC 알고리즘은 먼저 각 요구 분석 문서에 속해 있는 동사들을 키워드로 하여 문서를 기능별로 분류하고, 이렇게 기능별로 분류된 문서들을 사이에 코사인(cosine) 유사도를 측정하여 재분류하는 방법이다. 그리고, Maarek은 [1]에서 자동화된 소프트웨어 라이브러리를 만들기 위하여 슬라이딩 윈도우 기법을 이용한 색인 추출과 유사도 그래프의 기울기를 이용한 클러스터링 방법을 제안하였다. 이외에도 유의어 사전(thesaurus)을 이용하거나 단어의 반복(reiteration)과 공기(collocation) 같은 언어 현상을 이용한 많은 연구들이 정보 검색 분야에서 이루어졌다[3]-[9]. 그러나, 이러한 방법들은 분산 환경에서의 요구분석 시에 필요한 문서간 유사도 검사뿐만 아니라 한 문서 내에서 나타나는 요구 사항의 충돌, 일관성의 결여, 그리고 요구 사항 자체의 불명확성들에 대한 효율적인 정보를 제공해 주지 못한다. 그리고, 유의어 사전을 이용한 방법들은 문서간 유사도 측정의 정확률 면에서 개선된 성능을 보이고는 있으나 특정 영역마다 이러한 사전을 구축하고 관리하는 일은 결코 쉬운 작업이 아니다. 또한, 정보 검색 분야에서 많은 연구들이 있어 왔지만, 이러한 방법론들이 소프트웨어 공학 분야에 실제 적용되어진 예를 아직은 쉽게 찾아 볼 수 없다. 그러므로, 본 논문에서는 공기 정보(co-occurrence information)를 이용하여 문서간 유사도를 측정하고, 문서 내에서의 일관성이 결여된 문장과 불명확성을 가진 문장을 찾아주는 통합 시스템을 제안한다.

3. 문서간 유사도 측정 기법

3.1 전체 시스템 구조도

본 논문에서 제안하는 시스템은 크게 문서간 유사도 측정부와 문장간 유사도 측정부, 그리고 모호한 문장 추출부로 나뉜다.

문서간 유사도 측정부는 요구 분석시에 상위 문서와

하위 문서 사이의 연계성을 유지하기 위한 도구로 사용될 수 있으며, [그림1]과 같이 색인 추출부와 유사도 측정부로 나뉠 수 있다.

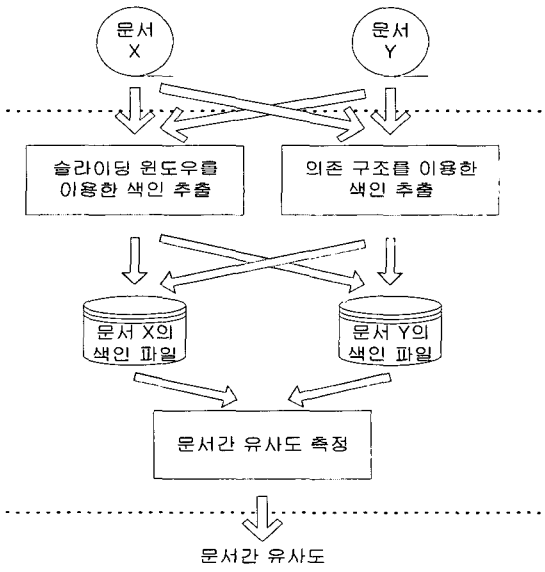

그림 1 문서간 유사도 측정부

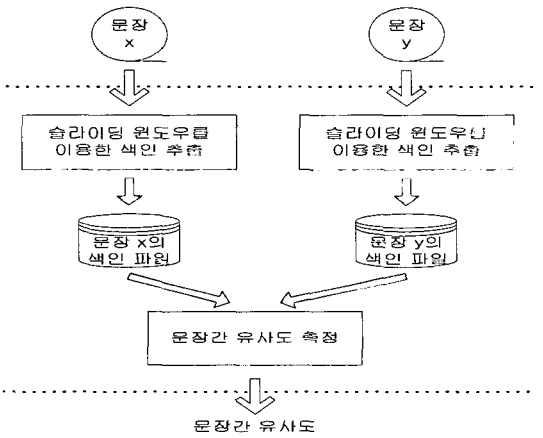

그림 2 문장간 유사도 측정부

색인 추출부는 슬라이딩 윈도우와 의존 구조(dependency tree)[10][11]를 이용하여 정보량이 많은 단어쌍을 추출하는 부분이고, 유사도 측정부는 실제로 입력된 두 문서 사이의 유사도를 측정하는 부분이다. 슬라이딩 윈도우 기법은 인접한 단어 사이의 공기 정보를 추출하기 위한 것이며, 의존 구조 기법은 멀리 떨어져 있지만 서로 연관된 단어 사이의 공기 정보를 추출하기 위한 것이다.

문장간 유사도 측정부는 한 문서 내의 문장들 사이의 유사도를 측정하여 요구 사항의 일관성 유지에 도움을 주는 도구로 사용될 수 있다. [그림2]는 문장간 유사도 측정부의 구조도이다. [그림2]에서 볼 수 있듯이 의존 구조를 이용한 색인 추출부가 없다는 것을 제외하면 문서간 유사도 측정부와 유사하다. 그러나, 추출되는 색인의 수가 상대적으로 적기 때문에 동일한 유사도 측정 방법을 적용하는 것은 적절하지 못하다. 본 논문에서는 문서간 유사도 측정과 문장간 유사도 측정을 위해 독립된 두 종류의 방법을 제안한다.

모호한 문장 추출부는 요구 분석 문서에 적당하지 않는 문장들을 찾아주는 역할을 하며, 이를 위하여 [그림3]과 같이 모호한 뜻을 지닐 수 있는 단어들의 사전과 품사 태거(POS tagger)를 이용한다.

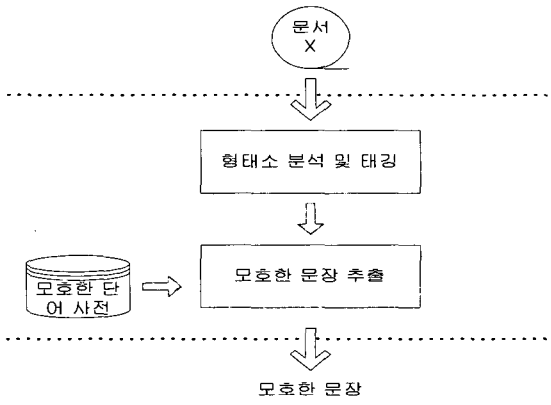

그림 3 모호한 문장 추출부

3.2 색인 추출

문서나 문장간의 유사도를 계산하기 위해서는 각각으로부터 색인들을 추출하고, 그 색인들로 구성되어진 색인 파일(index file)을 만들어야 한다. 색인 파일은 대상 데이터(data)가 가지는 특징을 잘 표현할 수 있는 중심어 파일(keyword file)이라고 할 수 있다. 색인 추출 방법은 형태에 따라 크게 두 가지로 구분된다. 한 단어가 하나의 색인을 구성하는 single-term 방법과 여러 단어로 이루어진 하나의 구가 색인이 되는 term-phrase 방법이 그것이다. Single-term 색인은 단일 단어로 구성되어 적은 양의 데이터에서도 많은 색인을 추출할 수 있다는 장점이 있는 반면에 문맥 정보를 포함할 수 없다는 단점이 있다. 이에 반해 term-phrase 색인은 공기

정보 등을 이용해서 단어의 쌍 등을 색인으로 보는 것으로 문맥 정보를 어느 정도 반영할 수 있다는 장점이 있다[1][7]. 본 논문에서는 슬라이딩 윈도우(sliding window) 기법[1]을 이용한 공기 정보와 의존 관계를 이용한 공기 정보를 색인 추출에 이용한다.

3.2.1 슬라이딩 윈도우에 의한 단어쌍 추출

색인 추출에 문맥 정보를 반영하기 위한 방법으로 인접한 단어 사이의 공기 정보가 그 동안 많이 사용되어 왔다. 본 논문에서는 인접한 단어 사이의 공기 정보를 추출하기 위해서 슬라이딩 윈도우 기법을 사용하는데, 다음과 같은 단계를 거쳐 수행된다. 먼저 문장을 각 형태소별로 나누어 품사를 결정한다. 그리고, 이를 통해 각 형태소 별로 그 문장의 내용이나 특징을 잘 내포 할 수 있는 단어와 그렇지 못한 단어를 구분한다. 문장의 내용이나 특성을 잘 반영하는 단어를 내용어(content word; open-class word)라고 하며, 명사, 동사, 형용사 등에 해당되는 단어들을 말한다. 다음으로 추출된 내용어의 순서열에 일정 크기의 윈도우(window)를 설정하고, 윈도우의 맨 앞의 내용어와 다음 내용어들간의 쌍을 추출한다. 본 논문에서는 윈도우의 크기를 5로 한다[12]. 윈도우는 문장의 처음에서부터 마지막 내용어까지 움직이며, 크기는 문장의 끝에서 문장의 경계를 넘지 않도록 줄어든다. 문장의 끝에서 윈도우의 크기를 줄이는 이유는 다른 문장에 속해 있는 내용어가 같은 문장내의 내용어 보다 약한 문맥 정보를 가지기 때문이다. 그리고, 윈도우의 슬라이딩(sliding)을 한 문장으로 제한하여 추출되는 색인의 수를 적절한 수준으로 유지하기 위해서이다. [그림4]는 본 논문에서 사용한 슬라이딩 윈도우 기법을 이용하여 색인어를 추출한 예이다.

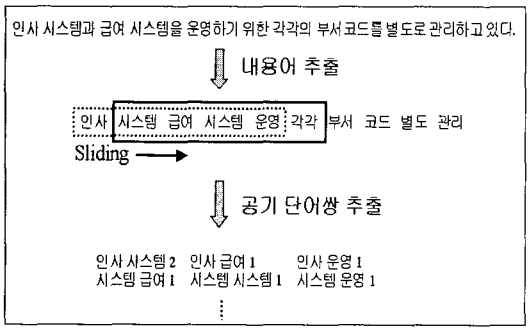

그림 4 슬라이딩 윈도우 기법의 적용 예

3.2.2 의존 구조에 의한 단어쌍 추출

문맥 정보는 인접한 단어의 쌍 뿐만 아니라 의존 관계를 맺고 있는 지배소와 의존소 사이에서도 발견된다. 그리고, 이러한 지배소와 의존소 사이 문맥 정보는 슬라이딩 윈도우 기법에서 다루지 못하는 윈도우 밖에 존재하는 단어와의 상관 관계를 설명해줄 수 있다. 그러므로, 단어 사이의 의존 관계를 색인어 추출에 이용한다면 보다 문서의 특징을 잘 반영하는 단어쌍을 얻을 수 있다. 본 논문에서는 이러한 특징을 이용하여 의존 문법(dependancy grammar)[11]을 기반으로 의존 구조를 추출하고, 그 결과를 색인어 추출에 반영한다. 그러나, 일반적인 의존 구조를 색인어 추출에 그대로 이용하는 것은 다음과 같은 문제점을 가지고 있다. 첫째, 일반적인 의존 구조는 어절 단위로 지배소와 의존소를 결정하기 때문에 한 어절이 여러 개의 내용어로 구성된다면, 지배소에 존재하는 내용어와 의존소에 존재하는 내용어 그리고 둘 사이의 의존 관계를 표현하지 못한다. 예를 들어, "각각의 부서코드를 별도로 관리하고 있다."라는 문장의 의존 구조를 분석하면 '부서코드를'이 '관리하고'에 의존한다는 사실을 알 수 있다. 그러나, 의존소와 지배소 내에 존재하는 내용어와 그것들 사이의 의존 정보는 얻지 못한다. 즉, 의존소 내의 의존 관계인 '(부서,코드)'와 지배소와 의존소 사이의 의존 관계인 '(코드,관리)'가 추출되지 않는다. 둘째, 지배소가 '체언+하다'나 '체언+이다' 형태인 경우에 올바른 단어쌍이 추출되지 않는다. 일반적으로 '체언+하다'는 '체언'과 '하다'로 나누어 의존 구조를 구성하기 때문에 "신규코드 발생시 DB로 입력한다."라는 문장이 있을 경우에 'DB'로는 '한다.'와 의존 관계를 맺는다. 그러므로, 올바른 색인어라고 생각되는 '(DB,입력)'이라는 단어쌍이 추출되지 않는다. 이러한 문제점을 극복하기 위해서 본 논문에서는 [그림5]와 같은 휴리스틱을 제안한다.

① 한 어절 내의 의존 관계
IF *한 어절이 합성어* THEN
어절 내의 내용어들은 자신의 바로 다음에 위치한 내용어와 의존 관계를 맺음
END

② 어절 간의 의존 관계
IF *의존소 D가 지배소 H에 의존* THEN
의존소 D의 마지막 내용어는 지배소 H의 마지막 내용어에 의존
END

그림 5 의존 구조로부터의 색인어 추출 휴리스틱

[그림5]에 제안된 휴리스틱은 문제점으로 지적한 두 가지 경우로 인해 추출되지 않는 모든 단어쌍을 추출하

며, 의존 구조 분석 후처리 모듈로 구현되어 자동으로 처리된다. [그림6]은 제안된 휴리스틱에 의해서 단어쌍이 추출되는 예를 도식화한 것이다.

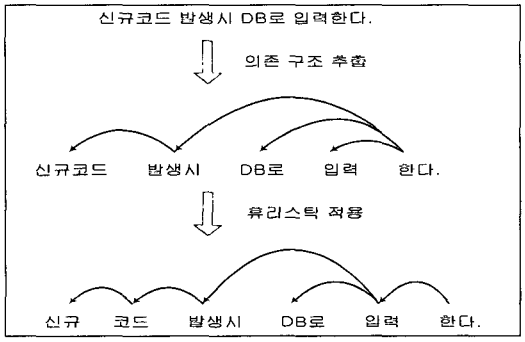

그림 6 휴리스틱 적용 예

3.2.3 문서간 유사도 측정을 위한 색인의 생성

특정 문서로부터 추출된 단어쌍을 분석해 보면 어떤 것은 문서의 특징을 매우 잘 반영한 반면에 어떤 것들은 그렇지 못하다. 문서의 특징을 잘 반영하지 못하는 단어쌍은 주로 특정 문서에서 특별한 의미를 내포하지 않고 어떤 문서에서나 자주 등장하는 단어가 포함된 것으로 영어의 'be' 동사를 포함하는 단어쌍 같은 것이 이에 해당한다. 이러한 단어들의 영향을 줄이기 위해서는 정보를 많이 포함하는 단어쌍을 선출해 내는 작업이 필요하다. 그러므로, 단어쌍의 중요도를 계산하기 위해서는 단어쌍의 문서내 출현 빈도뿐만 아니라 각 단어의 정보량까지 고려해야 한다. 특정 문서에 포함된 단어의 정보량은 [식1]과 같이 나타낼 수 있다[13][14].

$$INFO(w) = -\log_2(P(w)) \quad (1)$$

[식1]에서 P(w)는 문서에서 단어 w가 나타나는 출현 빈도를 확률로 나타낸 것이다. 즉, 그 문서에서 가장 출현 빈도가 많은 단어는 가장 적은 정보량을 가지게 된다. 그러므로, 단어쌍의 정보량은 다음 식과 같이 근사화될 수 있다[1].

$$\begin{aligned} INFO((w1,w2)) &= -\log_2(P(w1,w2)) \\ &\approx -\log_2(P(w1)\times P(w2)) \end{aligned} \quad (2)$$

[식2]를 통해 계산되어진 단어쌍의 정보량과 출현 빈도를 이용해서 [식3]과 같이 단어쌍의 중요도를 계산한다[14]. 중요도가 높을수록 좀 더 문서의 특징을 잘 반영한다고 할 수 있다. [식3]에서 f는 문서 내에 출현한 단어쌍 (w1,w2)의 출현 빈도이다.

$$\rho((w1,w2,f) = f\times INFO((w1,w2)) \quad (3)$$

문서의 크기와 관계없이 상대 비교가 가능하도록 하기 위하여 [식3]을 [식4]와 같이 표준화한다. $\bar{\rho}$는 단어쌍 중요도 ρ의 평균이며, σ는 표준 편차이다.

$$\rho_z = \frac{(\rho - \bar{\rho})}{\sigma} \quad (4)$$

[식4]를 z-score라 하며, 이 값이 문서간의 유사도를 측정하는 기준으로 사용될 수 있다[1]. 본 논문에서는 슬라이딩 윈도우에 의한 단어쌍과 의존 구조에 의한 단어쌍을 모두 합하여 z-score 계산하고, 이것을 이용하여 색인 파일을 구성한다.

3.3 문서간 유사도 측정

3.2.3절에서 구성된 색인 파일을 이용하여 문서간의 유사도를 계산한다. 문서간의 유사도를 계산하는 방법은 [식5]와 같다[1].

$$\delta(X,Y) = \sum_{i\in p(X)\cap p(Y)} (\rho X(i)\times \rho Y(i)) \quad (5)$$

[식5]에서 $p(X)$와 $p(Y)$는 문서 X, Y의 색인 파일을 나타내며, $\rho X(i)$와 $\rho Y(i)$는 문서 X와 Y에 존재하는 i번째 색인의 ρ_z 값을 의미한다. [식5]의 의미는 두 문서의 색인 파일에 공통으로 존재하는 단어 쌍의 z-score 값의 곱을 더하는 것으로 이 값이 클수록 유사도가 높게 된다. 본 논문에서는 각 문서가 가지는 색인의 ρ_z 값 중에서 0보다 큰 모든 값이 $\delta(X,Y)$를 증가시키는 방향으로 작용하도록 하기 위해 X축으로 1만큼 평행 이동시킨다. 즉, 모든 ρ_z값에 1을 더하고, 0보다 작은 것들은 계산 대상에서 제외한다. 이것은 같은 어휘를 갖는 단어쌍의 빈도가 매우 적기 때문에 발생하는 희소 데이터 문제를 보정하기 위한 것이다.

3.4 문장간 유사도 측정

한 문서 내에 존재하는 문장 사이의 유사도를 측정하기 위해서 3.3절의 방법을 그대로 이용하는 것은 다음의 두 가지 이유로 인하여 적합하지 않다. 첫째, 3.2.3절 [식3]의 인자 f는 문서 내에 출현한 단어쌍의 빈도수로 문장간에는 대부분 1에 가까운 상수가 된다. 그러므로, [식3]이 큰 의미를 갖지 못한다. 둘째, 문장간의 유사도 측정을 위한 색인의 수가 문서간의 유사도 측정을 위한 색인의 수보다 상대적으로 적다. 그러므로, 단어쌍 중요도 ρ의 평균과 표준 편차의 변화가 거의 없어 모든 단어쌍이 항상 비슷한 중요도를 가지게 된다. 본 논문에서는 문장간의 유사도를 측정하기 위해서 [식6]과 같은 Salton의 코사인 계수(cosine coefficient)[13]을 이용한다.

$$\cos(x, y) = \frac{|x \cap y|}{|x| \times |y|} \quad (6)$$

[식6]에서 $|x|$와 $|y|$는 문장 x의 색인 수와 문장 y의 색인 수이고, $|x \cap y|$는 두 문장에서 공통된 색인의 수이다. [식6]에서 알 수 있듯이 Salton의 코사인 계수는 공통된 색인의 수가 같을 때, 각 문장의 색인의 수가 적을수록 높은 유사도를 갖는 특징을 지니고 있다. 이것은 긴 문장에서 많은 색인이 뽑히면 그만큼 공통된 색인의 수가 나올 확률이 높아지므로 이러한 경우에 불이익을 주기 위한 것으로 해석될 수 있다.

3.5 모호한 문장의 추출

본 논문에서 모호한 단어는 모호한 문장 즉 분석가에 따라 다르게 해석될 수 있는 문장의 원인이 되는 단어를 의미한다. 예를 들어, "많은 조회 기능을 제공한다." 라고 하는 문장이 있을 때, 이 문장은 분석가에 따라 의미가 달라질 수 있다. 즉, 어떤 분석가는 조회 기능을 10회 제공하는 것이 많다고 볼 수 있고, 다른 전문가는 100회 제공하는 것이 많다고 볼 수 있다. 이렇게 분석가의 주관에 따라 다르게 해석될 수 있는 '많은'과 같은 단어를 모호한 단어라고 한다.

모호한 문장을 추출하기 위한 방법으로 본 논문에서는 요구 분석 문서에 적합하지 않은 모호한 단어들의 사전을 구축하고 형태소 단위로 비교하는 방법을 이용한다. 먼저, 입력된 문서를 형태소 태깅(tagging)한 후, 모호성을 가질 수 있는 부사나 동사, 형용사에 해당하는 단어들을 추출한다. 그리고, 구축되어 있는 사전과 비교하여 해당되는 단어를 포함하는 문장을 출력한다.

모호한 단어 사전은 다음의 과정을 따라 수정, 보완된다. 먼저, 시스템 설계 시에는 '많다', '적다'와 같이 명확히 부적절한 단어들을 이용하여 기본 단어 사전을 구성하고 각각의 단어에 기본 가중치 1을 부여한다. 그리고, 분석가는 기본 사전을 이용하여 분석을 하면서 현재 추출된 문장이 모호한 의미를 가지고 있는지 그렇지 않은지에 대한 결과를 시스템에게 돌려준다. 모호한 문장에 대한 분석가의 의견은 그 문장을 추출한 모호한 단어의 가중치로 계산된다. 요구 분석을 하는 과정에서 새롭게 출현한 모호한 단어들은 가중치 1을 부여받고 사전에 추가된다. 본 논문에서는 [식7]과 같이 가중치를 부여한다.

$$W(w_i) = W(w_i) + 1, \text{ if } w_i \subset S_j \quad (7)$$

[식7]에서 w_i는 가중치를 부여할 사전의 i번째 모호한 단어를 의미하고, S_j는 분석가에 의해 모호하다고 판단된 j번째 문장을 의미한다. 시스템은 [식7]에 의해서 가중치가 높은 모호한 단어를 포함하는 문장을 먼저 분석가에게 제공한다. 이것은 요구 분석 문서에 습관적으로 자주 모호하게 사용되는 말들에 분석가가 보다 주의를 기울일 수 있도록 해준다.

4. 실험 및 평가

4.1 실험 방법 및 데이터 구성

문서간 유사도 측정 기법은 주로 분산 환경에서 입력되는 요구 사항 문서들을 효율적으로 분류하고 관리하는 작업에 사용될 수 있다. 본 논문에서는 이러한 상황을 가정하고, 제안된 문서간 유사도 측정 기법의 효율성을 평가하기 위하여 다음과 같은 방법으로 실험 데이터를 구성하였다. 먼저, 33개의 요구 분석 문서를 각각 A와 B 두 부분으로 나누어 66개의 데이터 파일을 만들었다. 그리고, A에 있는 33개 각 문서와 B의 33개 문서에 대해 유사도를 측정하였다. A에 있는 한 문서에 대해 가장 유사도가 높게 나온 B의 문서가 원래 같은 문서-하나였는데 둘로 나누어진 경우-였다면, 정답으로 간주하였다. 수집된 요구 사항 문서의 길이는 1,090문장(7,883어절, 7.23어절/문장)이다. 유사도 측정은 슬라이딩 윈도우만을 이용하여 색인 파일을 만들었을 경우, 의존 구조만을 이용하여 색인 파일을 만들었을 경우 그리고 이 둘은 결합하여 만들었을 경우로 나누어서 실험하였다.

문장간 유사도 측정 기법은 상, 하위 관계에 있는 문장들의 연계성 유지에 이용될 수 있다. 본 논문에서는 제안된 문장간 유사도 측정 기법의 효율성을 평가하기 위하여 다음과 같이 실험 데이터를 구성하였다. 먼저, 하나의 요구 분석 문서를 상, 하위 문서로 나누고, 상위 문서의 문장과 하위 문서의 문장들 사이의 유사도를 측정하였다. 그리고, 상위 문서에 속한 문장을 보다 세분화하여 기술한 하위 문서의 문장을 정답으로 간주하였다. 문장간 유사도 실험을 한 상위 문서의 길이는 22문장(198어절)이고, 하위 문서의 길이는 20문장(138어절)이다.

모호한 문장을 추출하는 실험을 위한 데이터는 문장간 유사도 측정을 위한 실험 데이터와 동일하게 구성하였다. 그리고, 모호한 단어 사전은 '많다', '적다', '쉽다', '어렵다', '크다', '다양하다', '편리하다' 등의 단어를 포함하여 16단어로 구성되었다. 이것은 저자들의 경험에 의해 수작업으로 이루어졌다.

4.2 실험 결과

[표1]은 문서간 유사도 측정을 위해 3.1에서 언급한 3가지 색인추출 방법에 대한 정확률(precision)이며, [그

림7]은 이를 도식화한 것이다. 본 논문에서의 정확률은 지정된 상위 퍼센트 안에 정답이 포함될 확률을 의미한다. [표1]에서 보듯이 제안한 방법이 기존의 슬라이딩 윈도우 방법보다 상위 3%만을 고려한 경우에 12.1%정도 정확률이 향상되었다. 그리고, 상위 10%까지 고려한 경우에는 9.1%정도 향상되었다.

표 1 문서간 유사도 측정 기법들의 정확률 비교

	슬라이딩 윈도우 모델	의존 구조 모델	통합 모델
상위3%	60.6	9.1	72.7
상위10%	81.8	42.4	90.9

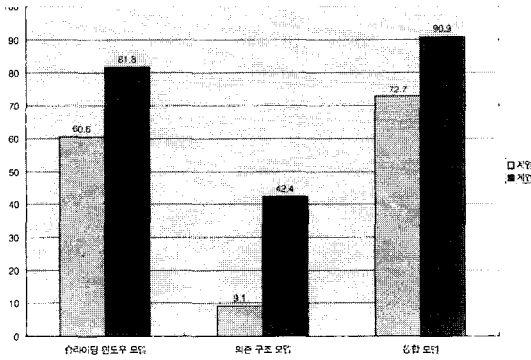

그림 7 문서간 유사도 측정 기법들의 정확률 비교

의존 구조만을 이용하여 색인을 추출한 경우에 정확률이 굉장히 낮았는데, 이것은 의존 구조만을 고려하기 때문에 다른 방법에 비해 상대적으로 적은 양의 색인이 추출되고, 인접한 단어들에 대한 공기 정보가 많이 반영되지 않기 때문인 것으로 보인다. 의존소와 지배소 내에 존재하는 인접한 내용어들의 공기 정보를 반영하기 위해서, 구문 관계를 맺고 있는 의존소와 지배소의 내용어들에 슬라이딩 윈도우 기법을 적용해 색인을 추출하는 방법도 실험하였다. 이 경우에 상위 1위의 정확도는 57.6%, 상위 3위는 81.8%로 기존의 슬라이딩 윈도우 방법과 비슷한 성능을 보였다. [그림8]은 A에 속해 있는 33개의 문서에 대해 B의 정답 문서가 몇 번째로 순위화(rank)되었는지를 보여준다. [그림8]에서 주목할 만한 것은 16번째 문서와 같이 슬라이딩 윈도우 모델과 의존 구조 모델이 모두 정답을 3위로 순위화한 것을 본 논문에서 제안한 통합 모델은 1위로 찾았다는 것이다. 이것은 본 논문에서 제안한 방법이 슬라이딩 윈도우를 사용한 인접 어절의 공기 정보와 의존 구조를 이용한 구문적 공기 정보를 효과적으로 이용한다는 것을 보여준다.

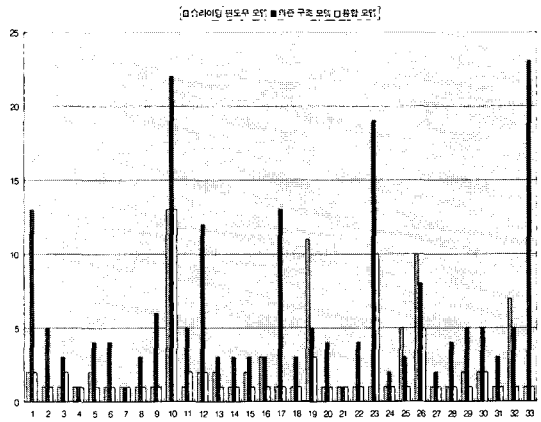

그림 8 문서별 정답 순위

제안된 통합 모델을 이용하면 각 문서 당 대략 3개의 문서를 살펴서 90% 정도의 확률로 유사한 문서를 찾을 수 있다. 이는 32개의 대상 문서를 모두 살펴야 하는 것과 비교해 보면 9.4%의 노력으로 90.9% 정도의 만족을 얻을 수 있다는 것을 의미한다. 그러므로, 분산 환경에서 대량으로 존재하는 문서들을 관리하고 분류하는데 제안된 통합 모델이 효과적으로 이용될 수 있을 것이다.

표 2 문장간 유사도 측정 기법들의 정확률 비교

	single-term	term-phrase
상위10%	68.2	72.7

[표2]는 3.4절의 방법을 통해 문장간 유사도를 측정한 결과이다. 비교 대상은 색인 추출 방법에 single-term을 이용한 것이다. [표2]에서 보듯이 슬라이딩 윈도우를 이용하여 색인을 추출하는 것이 single-term을 이용하는 것보다 4.5% 정도 효율적이었다. 그리고, 상위 10%이내에 들어 정답으로 계산된 문장들 사이에서도 슬라이딩 윈도우를 이용한 것이 single-term을 이용한 것보다 전반적으로 상위에 위치했다. 이것은 term-phrase를 이용하여 추출된 색인이 문장의 내용을 single-term보다 잘 표현해 준다는 것을 보여준다. 그리고, 72.7%의 신뢰도를 가지고 요구 분석 내용의 연계성을 검사할 수 있다는 것을 의미한다. 그러므로, 상, 하위

의 요구 분석 내용들 간의 충돌이나 불일치 문제 등을 해결하는데 효과적으로 이용될 수 있을 것이다.

모호한 문장을 추출하는 실험은 본 논문에서 제안한 단순한 방법이 얼마나 효과적으로 사용될 수 있는지를 보이는 것에 중점을 두었다. [표3]은 모호한 문장 추출에 대한 실험 결과이다. 제안된 방법은 1,090개의 전체 문장 중에서 34개의 문장을 추출했으며, 실제로 모호한 의미를 갖고 있는 16개의 문장을 모두 포함했다. 문맥을 고려해야만 모호성을 판단할 수 있는 문장은 실험에서 제외했다.

표 3 모호한 문장 추출 실험 결과

추출된 후보 문장 수	실제 모호한 문장수
34	16
재현율	정확률
100	47.1

[표3]에서 알 수 있듯이 제안된 방법은 매우 단순하지만 효과적이다. 모든 문장의 모호성 제거를 위해 분석가는 1,090문장 중에 단지 34문장만을 살펴보면 된다. 이것은 분석가가 모호성이 있는 문장을 찾기 위해서 전체 문장을 살펴보는 것에 비해서 3.1%의 노력만 기울이면 된다는 것을 의미한다. 모호한 문장 추출 실험에서의 재현율은 구성된 사전에 매우 의존적이다. 그러나, 요구 분석을 하는 과정에서 새롭게 발견된 모호한 단어들이 계속적으로 추가된다면 재현율의 급격한 하락은 없을 것으로 기대된다. 그러므로, 제안된 모듈이 모호한 문장을 찾는 도구로 사용된다면 문서의 질(quality)을 유지하는데 효과적으로 이용될 수 있을 것이다.

5. 구현 및 평가

본 장은 본 논문에서 제시한 방법들을 실제 시스템에 적용한 기능별 구현의 모습과 평가들을 기술한다. 시스템의 서버는 Solaris상에서 C를 사용하여 구현하였으며 클라이언트는 MS Window상에서 Visual Basic을 이용해서 구현하였다.

[그림 9]는 본 시스템의 초기 사용자 인터페이스를 보이고 있다. 초기 사용자 인터페이스는 유사도 측정을 하기 위해서 색인을 생성하는 부분과 기능을 수행하는 부분으로 나누어진다. 본 시스템의 기능은 문서간의 유사도 측정을 통해 문서간의 의존성을 분석하는 기능, 두 문서에서 문장간의 유사도 측정을 통해 문장간의 연계성을 설정하고 불완전성을 발견하는 기능, 한 문서에서 문장간의 유사도 측정을 통해 문장간의 중복성과 불일치성을 발견하는 것을 지원하는 기능 그리고 요구 분석 문서에 적합하지 않은 모호한 단어들을 가진 문장을 추출하여 문서의 불명확성을 발견하는 것을 지원하는 기능으로 이루어진다.

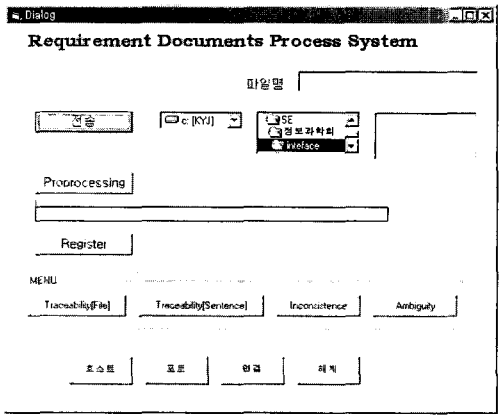

그림 9 요구 분석 시스템의 클라이언트 사용자 인터페이스

[그림 10]은 문서간의 유사도를 측정하는 모습이다. 사용자는 이 기능을 통해 새로 발생한 요구사항문서들이 기존의 어떤 요구사항문서와 관련이 있는지를 쉽게 찾을 수 있다. 그러므로, 요구사항문서의 관리와 분석을 효과적으로 수행할 수 있으며 또한 요구 사항 문서간의 의존성을 분석할 수 있는 기능을 지원한다.

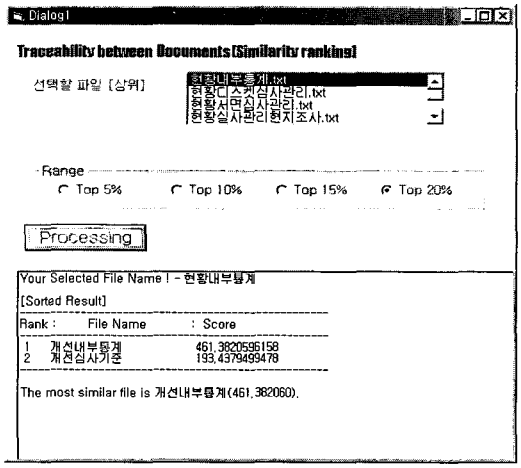

그림 10 문서간의 유사도 측정 기능의 실행 화면

[그림 11]은 상위 레벨과 하위레벨 문서의 문장간의

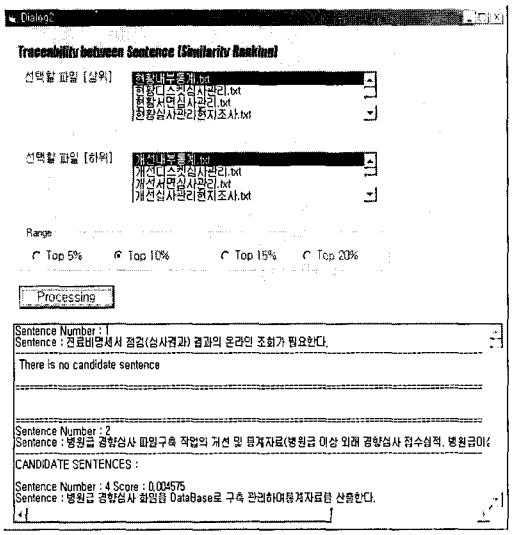

그림 11 두 문서의 문장간의 유사도 측정 기능의 실행 화면

유사도를 측정하는 모습이다. 이 기능은 사용자에게 요구사항 문장간의 유사도 측정을 통해 하위레벨 문서의 요구사항 문장 중에 어떤 문장이 상위레벨 문서의 요구사항 문장과 가장 비슷한 지를 자동으로 제시한다. 그러

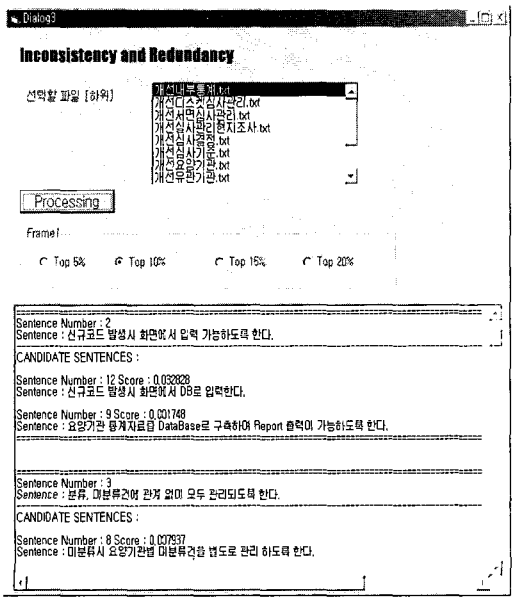

그림 12 한 문서의 문장간의 유사도 측정 기능의 실행 화면

므로, 상, 하위 레벨 문서에서 문장간의 연계성을 효과적으로 유지 할 수 있도록 도와준다. [그림 11]에서 상위레벨 문서의 첫 번째 문장은 하위레벨 문서에 해당되는 내용의 요구사항문장이 없음을 알 수 있다. 이는 하위레벨의 문서에 상위레벨 문서의 첫 번째 문장에 대한 요구사항 문장이 누락되어 있음을 나타내는 결과이며, 이 결과를 바탕으로 누락된 문장을 보충함으로써 요구사항문서의 불완전성(incompleteness)을 줄일 수 있다.

[그림 12]는 한 문서에서 문장간의 유사도 측정을 통해 문장간의 중복성과 불일치성을 줄이는 기능의 실행 모습이다. [그림 12]의 첫 번째 문장의 결과는 같은 내용의 문장이 같은 요구사항 문서에 두 번 언급되어 있음을 보이고 있는데 이러한 문장을 제거함으로써 요구사항문서의 중복성을 줄일 수 있다. 또한 두 번째 문장의 결과를 보면 두 문장이 서로 충돌(conflict)되는 내용의 문장임을 알 수 있다. 이런 문장들을 자동으로 제시함으로써 문장간의 불일치성을 줄일 수 있다.

[그림 13]은 요구 분석 문서에 적합하지 않은 모호한 단어들을 가진 문장을 추출하여 문서의 완전성을 높이는 기능의 실행 모습이다. 요구사항 문서에서 명확하지 않은(non-quantifiable) 단어들은 요구사항자체의 불명확성 문제를 발생하므로 이러한 단어가 나타나는 문장을 자동으로 찾아내어 문서의 질(quality)를 높이는 기능을 지원한다.

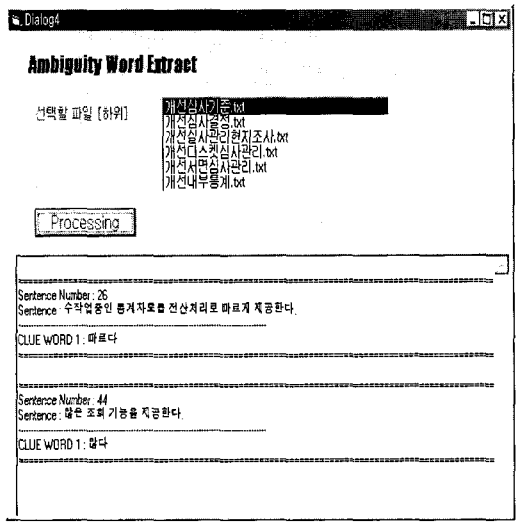

그림 13 모호한 단어를 가진 문장을 추출하는 기능의 실행 화면

6. 결론 및 향후 과제

본 논문은 요구 분석시 나타나는 오류를 효과적으로 줄이고, 수정하는데 사용될 수 있는 요구 분석 시스템을 제안하였다. 문서간의 유사도 측정을 위해서는 기존의 색인 추출 방법인 슬라이딩 윈도우 모델과 의존 구조 모델을 결합하여 각 모델이 가지는 단점을 효과적으로 보완할 수 있었다. 슬라이딩 윈도우 모델이 가지고 있는 제한된 윈도우의 크기에 의해서 먼 거리 공기 정보를 찾을 수 없다는 단점은 의존 구조 모델에 의해 보완될 수 있었다. 그리고, 의존 구조 모델이 가지고 있는 인접한 어절의 공기 정보를 효과적으로 추출하지 못한다는 단점은 슬라이딩 윈도우 모델에 의해 보완될 수 있었다. 의존 구조 모델에서는 지배소와 의존소에 존재하는 내용어들의 공기 정보들을 효과적으로 추출하기 위한 휴리스틱도 제안하였다. 문장간 유사도 측정을 위해서는 슬라이딩 윈도우 기법과 Salton의 코사인 계수를 이용하여 요구 사항의 충돌과 일관성이 결여된 문장을 찾을 수 있는 효율적인 방법을 제공하였다. 모호한 문장을 찾기 위해서는 미리 구축된 사전과 형태소 분석기를 이용하여 요구 분석서에 포함된 모호한 문장을 쉽게 찾을 수 있도록 하였다. 본 논문에서 제안된 시스템은 언어 분석의 비교적 하위 단계인 형태소 분석과 구문 분석만을 이용하여 유사도를 측정하기 때문에 비교적 단순하고 쉽게 구현될 수 있다는 장점이 있다. 제안된 유사도 측정 시스템을 이용한다면 연관된 문서를 쉽고 빠르게 찾을 수 있기 때문에, 사용자 요구 분석 시 발생하는 오류의 분석과 수정에 효과적으로 대처할 수 있다. 그러므로, 요구 분석 오류의 발생 빈도도 상당히 줄어들 것이다.

본 논문에서 제안된 방법에는 여러 문서에 공통적으로 많이 발생하기 때문에 별다른 정보를 주지 못하는 단어들을 제거하는 불용어 처리 부분이 없다. 효과적인 불용어 처리 부분이 있다면 보다 좋은 결과를 얻을 수 있을 것으로 보인다. 그리고, 색인 파일 구성에 단어쌍을 그대로 이용하였기 때문에 발생하는 희소 데이터(sparse data) 문제를 효과적으로 해결할 수 있는 보간법(interpolation)의 개발도 필요할 것으로 보인다. 마지막으로, 구문 분석 정보뿐만 아니라 명사나 동사의 의미 계층 구조를 이용하여 같은 의미를 가지는 것은 동일한 단어쌍으로 본다면 더 좋은 결과를 얻을 수 있을 것이다.

참 고 문 헌

[1] Maarek Y., Berry D. and Kaiser G, An Information Retrieval Approach For Automatically Construction Software Libraries, *IEEE Transaction On Software Engineering*, Vol. 17, No. 8, pp.800-813, August 1991.

[2] Palmer J. and Liang Y., Indexing and clustering of software requirements specifications, *Information and decision Technologies,* Vol 18, pp.283-299, 1992.

[3] Hearst M., "Multi-Paragraph Segmentation of Expository Text," *Proceedings of the ACL'94,* June 1994.

[4] Litman D. and Passonneau R., "Combining Multiple Knowledge Sources for Discourse Segmentation," *Proceedings of the 33rd ACL*, May 1995.

[5] Kozima H., "Text Segmentation Based on Similarity between Words," *Proceedings of ACL'93*, pp.286-288, January 1993.

[6] Yaari Y., "Segmentation of Expository Texts by Hierarchical Agglomerative Clustering," *Proceedings of RANLP'97*, pp.135-142, September, 1997.

[7] Jobbins A. and Evett L., "Text Segmentation Using Reiteration and Collocation," *Proceedings of the COLING-ACL'98*, pp.614-618, August 1998.

[8] Hajime M., Takeo H. and Manabu O., "Text Segmentation with Multiple Surface Linguistic Cues," *Proceedings of the COLING-ACL'98*, pp.881-885, August 1998.

[9] Kim M., Klavans J. and McKeown K., "Linear Segmentation and Segment Significance," *Proceedings of the 6th International Workshop of Very Large Corpora(WVLC-6)*, pp.197-205, August, 1998.

[10] Hellwig P., "Dependency Unification Grammar," *Proceedings of COLLING86*, pp.195-198, 1986.

[11] Mel'cuk I. A., Dependency Syntax: Theory and Practice, *State Univ. of New York Press*, 1988.

[12] Martin W.J.R., Al B. P. F., and van Sterkenburg P. J. G., "On the processing of a text corpus: From textual data to lexicographic information," in *Lexicography: Principles and Practice* (Applied Language Studies Series), Hartmann R. R. K., Ed. London: Academic, 1983.

[13] Salton G. and McGill M.J., *Introduction to Modern Information Retrieval* (Computer Series), New York:McGraw-Hill, 1983.

[14] Ash R.. *Information Theory*, New York:Wiley-Interscience, 1965.

김 학 수

1996년 건국대학교 전자계산학과 학사. 1998년 서강대학교 컴퓨터학과 석사. 1998년 ~ 현재 서강대학교 컴퓨터학과 박사과정. 관심 분야는 자연어 처리, 대화 시스템, 생략 및 대용어 처리, 화행 분석, 정보 검색

고 영 중

1996년 서강대학교 수학과 학사. 1996년 ~ 1997년 LG-EDS system 근무. 1998년 ~ 현재 서강대학교 컴퓨터학과 석사과정. 관심 분야는 자연어 처리, 정보 검색, 한국어 정보 처리. 대화 시스템

박 수 용

1986년 서강대학교 전자계산학과 공학사. 1988년 후로리다 주립대 전산학 석사. 1995년 George Mason University 정보기술학 박사, 연구 조교수. 1996년 ~ 1998년 TRW Senior Software Engineer. 1998년 ~ 현재 서강대학교 컴퓨터학과 조교수. 관심분야는 요구공학, 에이전트지향 소프트웨어 공학, 소프트웨어 지식 관리 시스템

서 정 연

1981년 서강대학교 수학과 학사. 1985년 미국 Univ. of Texas, Austin 전산학과 석사. 1990년 미국 Univ. of Texas, Austin 전산학과 박사. 1990년 ~ 1991년 미국 Texas Austin, UniSQL Inc. Senior Researcher. 1991년 한국과학기술원 인공지능 연구센터 선임연구원. 1991년 ~ 1995년 한국과학기술원 전산학과 조교수. 1995년 ~ 1996년 서강대학교 전산학과 조교수. 1996년 ~ 현재 서강대학교 컴퓨터학과 부교수. 관심분야는 한국어정보처리, 자연언어처리, 기계번역, 대화처리