소스코드 재사용을 위한 효율적인 의미망 구성에 관한 연구

A Study on Efficient Construction of Sementic Net for Source Code Reuse

김귀정
건양대학교

Kim Gui-Jung
KonYang University

요약

본 연구에서는 객체 지향 소스 코드의 검색과 재사용을 효율적으로 수행할 수 있는 의미망을 구축하였다. 이를 위하여 각 노드 간 객체지향 상속의 개념을 표현할 수 있도록 의미망의 초기 관련값을 시소러스로 구축하였다. 또한, 의미망의 노드와 간선을 활성화시키고 활성값을 전파시키기 위해 사용되는 spreading activation 방법의 단점을 보완하여 spreading activation의 성능은 최대한 유지하면서 검색 속도를 향상시킬 수 있는 방법을 제안하였다.

Abstract

In this paper we constructed semantic net that can efficiently conform retrieval and reuse of object-oriented source code. In oder that initial relevance of semantic net was constructed using thesaurus to represent concept of object-oriented inheritance between each node. Also we made up for the weak points in spreading activation method that use to activate node and line of semantic net and to impulse activation value. Therefore we proposed the method to enhance retrieval time and to keep the quality of spreading activation

Ⅰ. 서 론

최근 20년 간 소프트웨어 재사용에 대해 많은 노력들이 이루어져 왔으며, 특히 Case-Based Reasoning (CBR) 시스템에 대한 다양한 연구가 이루어져 왔다. CBR 사이클의 검색(retrieval) - 재사용(reuse) - 수정(revise) - 유지(retain) 중 재사용을 위해 가장 핵심이 되는 부분이 컴포넌트 검색인데, 이는 검색 대상에 따라 컴포넌트 기술 방법과 구축 방법 그리고 검색 방법이 크게 달라진다. 또한 컴포넌트를 검색하는 방법에 따라 사용자의 요구사항을 얼마나 반영할 수 있는가와 라이브러리 확장성 등이 결정되기 때문에 검색 방법은 매우 중요하다. 특히 소스 코드를 재사용하기 위한 Case 기반 검색에 있어서 의미망의 효율적인 구성은 무엇보다 중요하다[1].

이에 본 연구에서는 의미망의 효율적인 구성을 위해서 각 노드 간 객체지향 상속의 개념을 표현할 수 있도록 의미망의 초기 관련값을 시소러스로 구축하고자 한다. 이때, 각 Case를 구성하는 클래스들을 상속관계에 따라 개념적으로 분류하였고, 시소러스 방법에 퍼지 논리를 적용하여 객체지향 시소러스를 구축하였다. 또한, 의미망의 노드와 간선을 활성화시키고 활성값을 전파시키기 위해 사용되는 spreading activation 방법의 단점을 보완하여 spreading activation의 성능은 최대한 유지하면서 검색 속도를 향상시킬 수 있는 방법을 제안하고자 한다.

본 연구에서의 의미망 구축은 크게 2가지 관점에서

재사용에 유용하다. 하나는 프로그래머의 관심을 라이브러리 내에 있는 컴포넌트로 유도하여 재사용성을 높일 수 있다. 이는 검색된 컴포넌트 내에 각 클래스를 하이퍼링크로 라이브러리 API 연결시키는 방법 등을 사용하여 재사용을 유도할 수 있다. 다른 하나는, 여러 다양한 클래스들이 포함된 전형적인 유형의 프로그래밍 패턴을 제공함으로써 프로그래머로 하여금 프로그램의 가이드라인으로 사용할 수 있도록 도움을 준다.

Ⅱ. 소스코드로 부터의 의미망 구축

많은 CBR 시스템에서 라이브러리 내에 있는 각 클래스를 케이스(Case)로 취급하고 있으며, 또한 이러한 Case를 재사용 단위인 컴포넌트화 하고 있다. Case는 각 컴포넌트의 소스 코드를 가지고 있으며, 그 컴포넌트를 표현할 수 있는 속성이나 특징을 포함할 수 있다. 또한 컴포넌트의 행위적 특성을 텍스트로 포함할 수 있다[2].

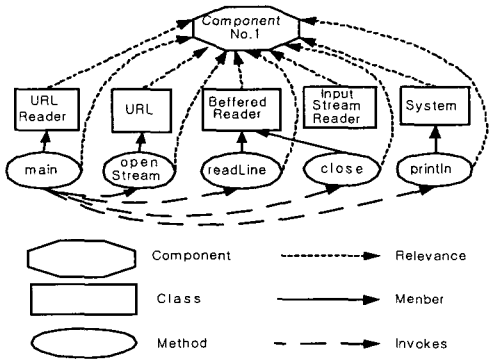
그림 1은 자바로 구현된 컴포넌트의 Case 예이다. 'URL_Reader' 컴포넌트는 BufferedReader를 사용해서 URL을 직접 읽어오는 방법에 대한 소스이다. Case 단위로 라이브러리에 저장된 각 컴포넌트는 파싱 단계를 거쳐 파싱 트리를 형성한다. 'class', 'interface', 'method', 그리고 'variable' 등을 식별자로 추출한다. 추출 과정은 컴포넌트 소스 코드를 읽어 각 식별자를 추출하고 식별자 사이의 관계 정보를 저장한다. 의미망은 파싱 트리로부터 만들어진다. 파싱으로부터 생성된 'class', 'interface', 'method', 'variable'은 의미망에서 노드로 구성되며, 'relevance', 'subclass', 'implements', 'member', 그리고 'invoke'등은 클래스들 간의 관계로 취급하여 노드간 간선으로 구성된다. 그림 1의 Case-URL_Reader에 대한 의미망은 그림 2와 같이 구성된다.

Source Code

```
import java.net.*;
import java.io.*;

public class URL_Reader
{
  public static void main(String[] args)
  {
    URL kbs=new URL("http://www.kbs.co.kr");
    BufferedReader in=new BufferedReader(
      new InputStreamReader(kbs.openStream()));
    String inputLine;
    while ((inputLine=in.readLine())!=null)
      Sysem.out.println(inputLine);
    in.close();
  }
}
```

▶▶ 그림 1. Case-URL_Reader



▶▶ 그림 2. 'URL_Reader' 컴포넌트의 의미망

의미망을 구축하기 위해서는 각 노드간 초기 관련값을 지정해주어야 하는데, 이에 대한 충분한 연구가 이루어지고 있지 않은 것이 현실이다. 지금까지는 대부분 정보 검색을 위해 용어들 간의 관련값 또는 유사값을 시소러스로 구축하는 방식이었기 때문에 객체지향 소스 코드의 재사용을 위한 검색에는 기존의 방법을 그대로 적용할 수 없다. 더욱이 컴포넌트 검색을 위한 시소러스의 구축은 전통적인 정보 검색 시스템에서 사용한 통계적 방법을 그대로 사용할 수 없다. 따라서 Case 기반의 소스 코드의 효율적인 검색

을 위하여 객체지향적인 특성을 만족하는 시소러스 구축방법이 필요하다.

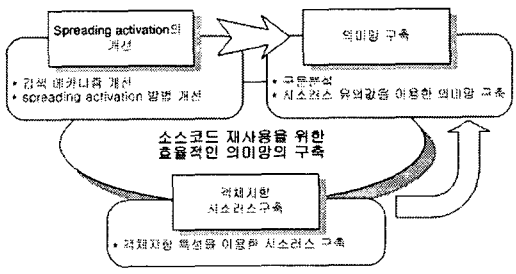
Ⅲ. 효율적인 의미망의 구축

1. 의미망 구축을 위한 요구 사항

Case 기반 컴포넌트 검색을 위해 효율적인 의미망을 구축하기 위해서는 객체지향 파라다임을 검색에 적합한 형태로 해석·적용하고, 기존의 시소러스에서 이용된 관계성을 클래스와 컴포넌트의 관계로 재정의함으로써 의미망의 구축과 검색을 위한 새로운 기본 구조가 만들어져야 한다[3]. 또한 구축된 의미망에 효율적인 검색 메카니즘을 적용함으로서 원하는 컴포넌트를 쉽게 검색할 수 있을 것이다. 기존 시소러스의 문제점들과 spreading activation 방법을 분석한 결과 Case 기반의 소스 코드를 재사용 하기위한 효율적인 의미망 구축을 위한 요구사항은 다음과 같다[4].

- 소스 코드를 분석하여 각 클래스의 정보와 계층구조를 이루는 상속관계 정보의 추출기능을 제공해야 한다.
- 객체지향 컴포넌트의 특징인 상속성을 잘 표현할 수 있는 관계와 구조를 제공해야 한다.
- Case으로부터 관계성을 추출하고 추출된 관계성을 시스템 차원에서 파악해줌으로써 시소러스를 자동할 수 있도록 해야 한다.
- 우리의 직감에 일치하는 잘 정의된 관계성을 포함한 시소러스를 도메인 전문가가 체계적이고 구조적으로 구축할 수 있도록 지원해야 한다.
- 구축과정에서 Case 내의 관계를 구조적으로 파악해서 구축자에게 필요한 정보를 제공함으로써 구축자의 부담을 최소화시켜야 한다.
- 클래스와 컴포넌트에 대한 시소러스 확장과 유지가 용이해야 한다.
- spreading activation의 장점은 유지하면서 빠른 검색을 위한 검색 메카니즘을 선정해야 한다.
- 시스템의 평가를 위한 정확도, 재현율 측정이 필요하다.

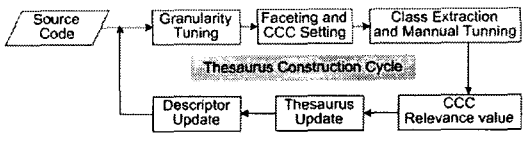
본 연구는 소스코드 기반 재사용을 위해 객체 지향 시소러스를 기반으로 한 의미망의 구축과 이를 통한 효율적인 검색을 위하여 spreading activation 방법의 개선 방법을 제안한다. 그림 3은 의미망 구축 방법을 나타낸 것이다.



▶▶ 그림 3. 의미망의 구축 방법

2. 시소러스 기반의 의미망 구축

본 연구에서 제안하는 의미망의 구축은 객체지향 시소러스를 기반으로 한다. 이는 객체지향 코드에서 서로 관련 있는 클래스들을 일정한 기준에 따라 그룹화해서 여러 개의 클래스 그룹으로 구성하는 방식이다. 여기에서 여러 개의 클래스 그룹은 개념들 간에 나타나는 일종의 시소러스를 의미하며, 클래스의 상속관계를 이용하여 개념들 사이의 관계를 자연스럽게 표현해야 한다. 그림 4는 소스 코드로부터 시소러스가 구축되는 전반적인 과정을 나타낸 것이다.



▶▶ 그림 4. 객체지향 시소러스 구축 과정

다음은 본 연구에서 제안하는 객체지향 시소러스를 기반으로 한 의미망 구축 단계이다.

·1 단계 : 클래스의 개념적 분류

클래스를 기능과 개념에 따라 여러 개로 나눈다. 이 과정은 도메인 전문가에 의해 행해지며 시스템의 응용에 따라 모든 클래스를 최적으로 표현할 수 있는 개념을 선정한다. 각 개념은 패싯 항목으로 표현되고 이에 따라 클래스가 분류되어진다.

·2 단계 : 클래스 범주 생성

개념적으로 분류된 클래스를 이용하여 클래스 개념 범주를 생성한다. 클래스가 사용될 수 있는 여러 경험적 상황을 패싯 항목으로 설정하는 패싯 분류방법을 사용한다. 클래스의 사용범위가 한가지로 국한되지 않고 여러 경우가 가능하며 이에 따른 경험적 솔루션을 제시해준다는 점에서 컴포넌트의 분류와 검색을 위해 클래스를 이용한 패싯 분류는 효율적인 방법이다.

·3 단계 : 클래스와 범주의 관계값 계산

분류된 범주를 사용하여 클래스와 범주간의 관계값을 계산한다. 이 과정은 범주를 이용한 클래스 빈도를 계산하는 과정이다. 이 빈도값이 범주별 클래스 관계값이 되며, 관계값에 의해 초기 시소러스 유의어 테이블이 구성된다.

·4 단계 : 객체지향 시소러스 생성

클래스와 범주의 관계값을 이용하여 퍼지 시소러스를 구축한다. 각 클래스와 범주에 대한 매칭 정도를 비교함으로써 이들 사이의 퍼지 정도를 계산하여 시소러스를 구축한다.

·5 단계 : 시소러스 기반의 의미망 구축

소스 코드로부터 추출된 각 클래스와 메소드는 의미망의 노드와 간선으로 표현되며, 초기 각 노드의 관련값은 시소러스의 유의값으로 설정된다.

3. Spreading Activation의 개선 방법

spreading activation 방법에서 활성값 계산을 위한 순환을 반복하는 것은 정확한 활성값으로 인한 유사 컴포넌트를 검색하기 위한 목적이다. 이 방법으로는 각 질의어와 컴포넌트의 활성값이 다른 질의어와 컴포넌트의 활성값 계산에 영향을 주기 때문에 시간이 많이 걸리는 단점이 있었다.

이에 본 연구에서는 spreading activation 방법의 단점을 해결하기 위하여 인덱싱이 적은 컴포넌트의 연결정보를 제거함으로써 활성값 계산회수를 줄여 검색시간을 단축시키는 방법으로 개선하고자 한다. 즉, 순환과정이 일정 수준 반복된 후 기준에 미치지 못하는 질의어나 컴포넌트의 연결정보를 제거하여 연산에서 제외시킴으로써 질의어의 확장범위를 줄여 보다 관계가 깊은 컴포넌트만을 검색하도록 하는 것이다.

제안한 spreading activation 방법은 다음과 같다.

- ·1 단계 : 의미망의 각 노드에 시소러스의 유의값을 초기 활성값으로 설정한다.
- ·2 단계 : 질의와 매칭되는 의미망의 노드가 활성화된다.
- ·3 단계 : 활성화된 노드의 초기 활성값이 연결된 다른 노드로 이동한다.
- ·4 단계 : 순환이 반복된다.
- ·5 단계 : 노드 간 연결의 참조 회수를 이용하여 제거 기준을 설정한다.
- ·6 단계 : 참조 회수가 기준에 못 미치면 연결을 삭제한다.
- ·7 단계 : 순환이 끝나면 질의와 직접 연결된 컴포넌트의 활성값과 유사한 활성값을 갖는 컴포넌트를 검색한다.

Ⅳ. 결 론

본 연구에서는 의미망의 효율적인 구성을 위해서 각 노드 간 객체지향 상속의 개념을 표현할 수 있도록 의미망의 초기 관련값을 시소러스로 구축하였다. 또한, 의미망의 노드와 간선을 활성화시키고 활성값

을 전파시키기 위해 사용되는 spreading activation 방법의 단점을 보완하여 인덱싱이 적은 컴포넌트의 연결정보를 제거함으로써 활성값 계산회수를 줄여 검색시간을 단축시키는 방법을 제안하였다. 이를 통하여 객체 지향 소스 코드의 검색과 재사용을 효율적으로 수행할 수 있는 의미망을 구축하였다.

▌참 고 문 헌▐

[1] Markus, G, Derek B, "Case-Based Reuse of Software Examplets," GWEM 2003 Proceedings of the Workshop on Experience Management, Apr. 2003.

[2] Gomes, P., Pereira, F.C., Paiva, P., Seco, N., Carreiro, P., Ferriera, J.L. & Bento, C., "Using CBR for Automation of Software Design Patterns," Proceedings of the Sixth European Workshop on Case-Based Reasoning, LNAI 2416, Springer, pp.543-548, 2002.

[3] 김귀정, 한정수, 송영재, "컴포넌트 검색을 지원하는 퍼지 기반 시소러스 구축", 한국정보처리학회논문지 제10-D권, 제5호, pp.753-762, 2003, 8.

[4] 김귀정, 한정수, 송영재, "설계 패턴 기반 컴포넌트 분류와 E-SARM을 이용한 검색,"한국정보처리학회논문지, 제 11-D권, 제 5호, pp.1133-1142, Dec., 2004.