프로그램 복제 검사 시스템

o
정재은 김영철 유재우
숭실대학교 컴퓨터학과
{archie,ychkim}@amin.soongsil.ac.kr, cwyoo@comp.soongsil.ac.kr

A Program Reproduction Test System

Jae-Une Jung Young-Chul Kim Chae-Woo Yoo
Dept. of Computing, SoongSil University

요 약

본 논문에서는 프로그램의 복제를 검사하기 위하여 서로다른 두 프로그램의 유사도를 측정하는 시스템을 제시한다. 지금까지 유사도 평가 방법은 일반 텍스트에 국한되어 있고 프로그램에 대한 유사도 검사는 극히 드물다. 본 시스템은 서로 다른 프로그램을 입력받아 분석 과정을 거쳐 구문 트리를 구성하고, 생성된 구문트리와 유사도 평가 시스템을 이용하여 프로그램의 유사도를 측정한다. 구문트리를 이용한 유사도 측정은 경제적이고 효율적으로 유사도를 검출해 낼 수 있다는 것을 평가에서 보여준다.

1. 서론

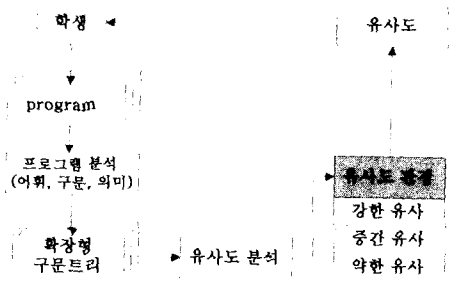
컴퓨터의 급속한 보급으로 이제는 컴퓨터를 사용하지 않는 사람은 극히 드물게 되었다. 얼마전에 모 대학에서는 학생들의 레포트를 컴퓨터로 작성된 것은 받지 않는다고 발표했다. 이것은 컴퓨터를 이용한 복제가 얼마나 심각한 수준에 이르렀는지를 단적으로 보여준다. 본 논문은 학생들이 제출한 과제물에 대한 유사도를 구문트리를 이용하여 측정함으로써 보다 빠른 평가와 복제 검사를 할 수 있도록 설계하였다. 현재 학생들이 제출한 프로그램 과제물에 대한 평가에서 교원은 객관적인 프로그램 평가 방법이 어렵다. 특히 실기 측정은 교원이 직접 프로그램을 실행시켜봐야 하며, 더욱 어려운 것은 표절에 관한 비교 측정 방법이다. 프로그램의 양과 학생 수가 많아지면 비교평가는 더욱 많은 시간과 노력이 필요하게 된다. 또한 실기 측정이 여러 명의 조교나 학생들에 의해서 이루어 지면 실기 측정에 대한 신빙성을 상당히 의심 받게 된다. 본 논문에서는 구문트리를 이용하여 유사도를 측정함으로써 교사 입장에서 학생들의 많은 과제물을 비교 검토할 수 있고, 학생들이 제출한 레포트에 대한 평가 시간을 최소화하고 평가 결과를 알려주어 학생들의 학습능력을 향상시킨다. 또한 표절이나 특정 패턴의 과제물을 적절히 검사할 수가 있다. 학생들은 자신의 과제물에 대한 평가와 더불어 잘못된 점에 대한 오류를 지적 받을 수 있으므로 학습에 도움을 받을 수 있으며, 또한 복제를 근절할 수 있는 장치를 마련함으로써 학생들의 학습능력을 향상시킬 수 있다. 본 논문은 복제자의 유형에 대한 유사 검사 방법과 이것이 얼마나 효율적인지에 대해서 간략히 설명한다.

2. 유사도 평가 시스템 구성도

본 시스템은 lex와 yacc을 이용하여 프로그램을 분석하는 부분과 구문 트리를 이용하여 유사도를 판정하는 부분으로 나뉘어져 있다. 실행되지 않는 프로그램을 제출했을 경우는 lex와 yacc을 이용한 프로그램 분석 부분에서 모두 검출해 낼 수 있으며 다른 사람의 프로그램을 복제한 경우는 구문 트리에 의한 유사도 시스템에서 검출해 낼 수 있다.

그림 1은 제출한 프로그램에 대해서 어떻게 유사도를 산출해 내는지 간략히 보여주고 있다. 학생이 제출한 프로그램은 lex와 yacc을 이용하여 구문과 의미를 분석하게 되며 이때 확장형 구문 트리가 생성된다. 확장형 구문 트리는 유사도 분석을 용이하게 하기 위해서 일반 트리를 확장한 형태로써 좌 노드, 중간노드, 우 노드

를 기본으로 하고 노드의 이름과 몇가지 정보를 추가로 가지고 있다. 유사도 분석에서 확장형 구문 트리를 이용하며 3장에서 자세히 다룬다.



[그림 1]

복제자의 유형과 정도에 따라서 원본과 90% 유사(강한 유사), 60% - 90% 유사 (중간유사), 60% 이하 유사(약한 유사)로 나눌 수 있으며 본 논문에서는 강한 유사를 중점으로 다룬다.

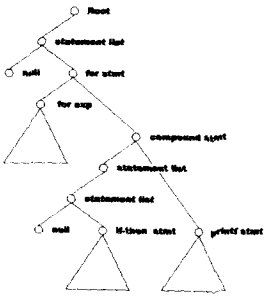
3. 유사도 분석 방법

여기서는 학생들이 제출한 과제물이 어떻게 유사 평가 되는지 실제 예제로써 제시한다. 아래의 예제는 0부터 9까지의 숫자 중 5를 제외한 나머지를 출력하는 간단한 프로그램이다.

```
main()
{
        int   i;

        for(i = 0;i < 10;i++)
        {
            if(i == 5) continue;
            printf("%d", i);
        }
}
```

그림 2는 예제 프로그램을 구문트리로 구성한 것이다. 구문트리는 statement list의 집합체로 되어 있으며 for, while, switch문 안에도 다중의 statement list가 존재할 수 있다. 구문트리는 statement list 별로 node가 구성되므로 분할 검사가 가능하고 흐름에 관련된 부분만을 따로 분류 할 수도 있다.



[그림 2]

여기서는 복제자의 90% 이상을 차지하는 완전복사, 변수 이름, 변수 할당 값 변환, 새로운 문장 삽입의 유사도 검사 방법에 대해서 설명한다.

1) 완전 복사, 변수 이름, 변수 할당 값 변환

복제자의 90% 이상을 이루는 이러한 변환은 구문트리에 전혀 변화를 주지 않기 때문에 구문트리를 하나의 링크로 구성한 후 일대일로 비교하는 것만으로 100% 유사도를 이끌어 낼 수 있다.

2) 새로운 문장 삽입, 원본에 없던 초기화 삽입

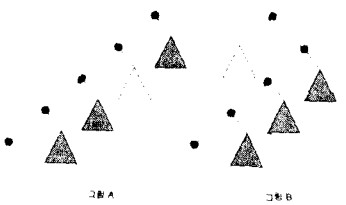


그림 A는 새로운 문장을 삽입했을 경우의 구문트리 변화를 나타내며 그림 B는 원본에 없던 초기화를 삽입한 경우의 구문트리이다.

- for, while, switch등의 삽입 : 새로이 생성된 statement list를 검사하지 않는다.
- printf, getc, puts등의 삽입 : 이와 같은 문장의 삽입은 프로그램의 흐름에 전혀 영향을 주지 않는 것이므로 중요도를 극히 낮게 한다.
- 원본에 없던 초기화 삽입 : 새로이 statement list가 생성되지만 흐름에 영향을 주지 않기 때문에 검사를 하지 않거나 중요도를 낮게 하는 방법사용

3) 두 문의 순서 조작

두 문의 순서를 조작한 프로그램의 유사도를 판별하는 것은 본 논문에서 제외한다. 왜냐하면 복제자의 대부분은 시간이 없거나 실력이 없는 초보자이므로 프로그램의 모든 내용을 이해하고 순서를 조작한 것에 대한 프로그램에 대한 유사도를 측정하는 것은 본 논문의 주제를 벗어나기 때문이다.

4) 구조의 유사성 검사

두 프로그램의 흐름이 완전히 일치한 상태에서 for나 while의 내용을 조금 수정하면 두 프로그램에 대한 구문트리는 구조적으로 일치하게 된다. 그러므로 이러한 구조적 유사성이 있는 구문트리에 대한 유사측정은 for나 while내의 statement list만을 유사검사하는 것으로 유사도를 검사한다.

4. 평가와 시스템의 특징

4.1 평가

과제물에 대한 복제자를 조사한 결과 원본 프로그램에 대한 완전복사 55% 변수 이름, 변수 할당 값 변환 37%, 새로운 문장 삽입 6% 두 문의 순서 조작 1% 였으며, 답과는 전혀 상관 없는 프로그램을 제출한 학생과 답 자체를 printf로 제출한 학생도 있었다. 그러므로 본 논문에서 제시한 유사도 검사에 의해서 약 90% 이상의 복제자를 가려낼 수 있다.

4.2 본 시스템의 특징

① 구문트리를 이용하여 방대한 프로그램에서 작은 프로그램까지 유사도 검사에 적용될 수 있다.
② 복제를 한 당사자가 약간의 프로그램을 수정했을 경우에도 유사도를 추정하는 것이 가능하다.
③ 구문트리는 프로그램에 에러가 있을 경우 트리를 생성할 수 없으므로 에러를 쉽게 판별해 낼 수 있다.
④ 함수 각각에 대해서도 유사도를 검사할 수 있다.
⑤ 복제를 한 당사자가 프로그램의 변수나, 함수 이름, 변수에 할당된 값 등을 바꾸었을 경우 100% 유사 검사가 가능하다.
⑥ While이나 if, swich문 등의 안에 복제자가 간단한 변수나, 수치를 추가 했을 경우 구문 트리에 대한 전체적인 트리는 영향을 받지 않으므로 이 때는 프로그램이 강한 유사도를 나타내게 되고 검사자는 이 프로그램에 대해서 정밀 검사를 진행하게 된다.
⑦ 함수나 순환문, if then, switch등의 순서를 바꾸었을 경우는 구문트리가 크게 변화 하게 된다. 하지만 함수, 순환문, if then, switch등의 개개의 트리를 따로 유사도 검사 한다면 이를 해결할 수 있다. 즉, 함수 순환문, if then, switch등의 순서를 바꾸어도 개개의 트리를 따로 검사함으로써 강한 유사도를 유추해 낼 수 있다.

5. 결론 및 향후 연구 과제

본 연구는 복제자의 다양한 유형에 맞추어 유사도 분석 방법을 제시하였다. 이 시스템은 프로그램에 대해서 분할 비교가 가능하고 과제물의 수나 양이 많을 때 큰 힘을 발휘한다. 또한, 일관성 있는 과제물의 평가가 가능하므로 학생들의 학습능력을 높일 수 있다.

향후 과제로는 구문트리를 보완하는 유틸리티를 제작해야 할 것이다. 구문트리는 악의를 가진 복제자에게는 약한 유사도를 보이는 취약점이 있으므로 이를 극복할 수 있는 유틸리티를 추가함으로써 보다 강력한 유사도 측정 시스템을 구축할 수 있도록 해야 할 것이다. 또한 초고속 정보 통신망의 활용 분야로 확장 가능하도록 기능을 확대해야 할 것이다.

참고 문헌

[1] J. R. Levine, Tony Mason & Doug Brown, *UNIX Programming Tools lex & yacc*, OReilly Pub. 1995.
[2] A. V. Aho and S. C. Johnson, LR Parsing, "ACM Computing Surveys", Vol. 6, 1974.
[3] D. E. Knuth, "Top down Syntax Analysis", No 55, Acta Informatica, 1971.
[4] M. E. Lesk, *Lex A Lexical Analyzer Generate*, Comp. Sci. Tech. Rep. No 39, Bell Laboratories, Murray Hill, New Jersy, 1975.
[5] *Amsterdam Compiler Kit Manual*, UniPress Software Inc. 1987.
[6] R. W. Sebesta, *concepts of : Programming Languages*, Third Edition, Addison Wesley, 1996.
[7] A. V. Aho, R. Sethi & J. D. Ullman, *Compilers Principles, Techniques, and Tools*, Addison Wisley, 1986.
[8] J. M. Spivey, *The Z Notation : A Reference Manual*, Prentice Hall, 1989.
[9] B. W. Kernighan & P. J. Plauger, *The Elements of Programming Style*, McGraw Hill, 1974.