바이트코드 분석을 이용한 자바 프로그램 표절검사 기법

지정훈° 우균 조환규
부산대학교 컴퓨터공학과
jhji@pusan.ac.kr woogyun@pusan.ac.kr hgcho@pusan.ac.kr

A Plagiarism Detection Technique for Java Program Using Bytecode Analysis

Jeonghoon Ji° Gyun Woo Hwangue Cho
Dept. of Computer Engineering, Pusan National University

1. 서 론

지금까지 연구된 많은 프로그램 표절검사 시스템들은 소스코드를 이용하여 표절을 판별한다. 대표적인 프로그램 표절검사 시스템인 JPlag[1]의 경우, 입력으로 받은 모든 소스코드를 서버로 전송하여 서버에서 표절검사를 수행하고 결과를 웹브라우저를 통하여 사용자에게 보고한다. 웹을 통해 표절검사를 수행하는 대부분의 시스템들이 서버로 소스코드를 전송해 서버에서 표절검사를 수행한다.

이와 같이 소스코드를 이용한 표절검사에서는 소스코드를 직접 표절검사 시스템으로 전송해야 하기 때문에 코드 보안이 문제가 될 수 있다. 컴파일된 코드를 표절검사에 이용한다면, 소스코드를 공개하지 않아도 되기 때문에 코드보안 문제를 해결할 수 있다.

본 논문에서는 바이트코드 분석을 통해 자바 프로그램의 표절을 검사하는 기법에 대해 소개한다. 즉, 자바 프로그램 표절검사를 위해 소스코드를 비교하지 않고 클래스 파일만을 이용하여 표절검사를 수행한다. 본 논문에서 구현한 자동 표절검사 시스템인 PINTB(Plagiarism INvestigating Tool)는 표절검사는 크게 두 단계로 나뉜다. 전단부에서는 클래스 파일을 입력으로 받아 바이트코드를 추출하는 바이트코드 선형화를 수행한다. 바이트코드 선형화에서는 실제 프로그램의 수행에 필요한 메소드의 코드만을 추출한다. 표절검사의 후단부에서는 지역정렬 알고리즘을 이용해 선형화된 두 프로그램 사이의 유사도를 계산하고 유사구간을 밝혀낸다.

2. PINT$_B$: Plagiarism INvestigating Tool using Bytecode

PINT$_B$는 크게 두 부분으로 나누어 표절검사를 수행한다. 표절검사의 전단부에서는 클래스 파일을 입력으로 받아서 중간 표현인 토큰 시퀀스(token sequence)를 생성한다. 그리고 후단부에서 생성된 토큰 시퀀스 쌍들에 대해 지역정렬을 이용해 두 자바 프로그램 사이의 유사도를 산출한다. 그림 2는 PINT$_B$의 시스템 구조도를 나타낸다.

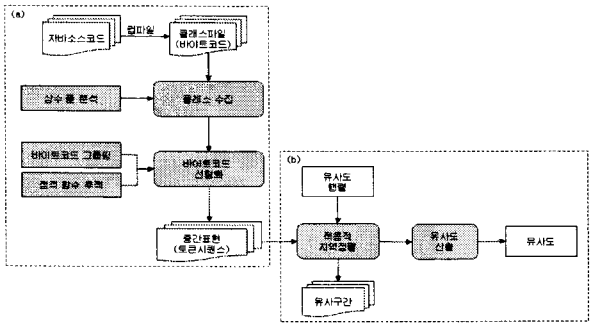


그림 1 PINT$_B$ 표절검사 시스템 구조도

그림 2-(a)에서는 자바 프로그램의 메인 클래스(main class)를 입력으로 받아서, 상수 풀의 정적 링크 정보를 이용하여 필요한 클래스들을 수집한다. 다음으로, PINT$_B$는 정적 함수 추적을 통하여 함수 호출 순서에 따라 토큰 시퀀스를 만든다. 그림 2-(b)에서는 프로그램들 사이의 유사도를 산출한다. PINT$_B$는 유사도를 바탕으로 표절 프로그

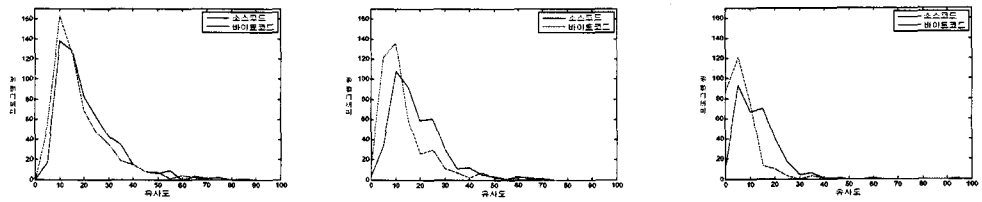
램들을 검출한다. 유사도 산출은 적응적 지역정렬[2]을 토큰 시퀀스에 적용하여 계산한다. 지역정렬은 생물정보학에서 DNA 시퀀스들 사이의 부분적 유사 기능을 밝히는데 사용되는 정렬 기법이다.

3. 실험 및 평가

본 논문에서는 바이트코드를 표절검사에 이용할 수 있는지 알아보기 위해 자바 프로그램을 대상으로 자바 소스코드를 이용한 표절검사 결과와 바이트코드를 이용해 표절검사를 수행한 결과를 비교해 보았다. 표절실험에는 2007학년도 자바프로그래밍 강의에서 학생들이 제출한 소스코드와 JPlag에서 사용한 실험데이터를 이용하였다. 표 1은 실험데이터를 정리한 것이다. 그리고 그림 2는 프로그램 그룹들에 대한 유사도 분포를 나타낸다.

표 1 실험에 사용된 테스트 프로그램 집합: 2006년도 자바프로그래밍 실습문제(G01, G02), 표절검사 시스템 JPlag에서 제공하는 소스코드(G03) : AWT를 이용한 GUI 프로그램

그룹	소스코드	순서쌍 수	클래스 수	소스코드 라인 수			
				최대	최소	평균	표준편차
G01	35	595	84	146	46	79.05	23.84
G02	32	496	69	180	50	87.41	33.23
G03	30	435	33	942	199	355.00	131.30



(a) G01 그룹 (b) G02 그룹 (c) G03 그룹

그림 2 프로그램 유사도 분포

그림 6-(a)는 G01 프로그램 그룹의 모든 프로그램 쌍들에 대한 유사도 분포이다. G01 그룹의 경우는 소스코드를 이용한 표절검사 결과와 바이트코드를 이용한 표절검사 결과가 비슷한 분포를 나타내었다. 그리고 그림 6-(b)와 6-(c)의 G02, G03 그룹의 경우에는 유사도가 30% 미만일 경우에 바이트코드의 유사도가 소스코드에 비해 낮게 분포하였다. G01 그룹의 전체 유사도 평균을 살펴보면, 소스코드의 유사도 평균은 22.96%이었고, 바이트코드의 유사도 평균은 19.11%로 약 3.85% 가량 차이를 보였다. G02 그룹의 경우에는 21.10%와 15.53%로 바이트코드가 소스코드에 약 5.57% 낮은 결과를 보였다. 이는 30% 이하의 유사도를 보인 소스코드들의 바이트코드에서 많은 차이를 보였기 때문이다. 그러나 G02 그룹에서 가장 높은 유사도를 보인 프로그램 쌍은 소스코드와 바이트코드의 경우 모두 동일한 프로그램쌍이 1등을 했으며, 유사도는 71.74%와 69.46%로 약 2.28%의 근소한 차이를 보였다. 또한 G03 그룹에는 소스코드로 검사했을 때, 유사도가 100%인 동일한 프로그램 쌍이 포함되어 있었다. 이는 소스코드를 의도적으로 표절해 유사할 경우, 바이트코드 검사를 통해서 검출할 수 있음을 보여준다.

4. 결론

본 논문에서는 자바 프로그램의 표절검사에서 소스코드 없이 바이트코드만으로 표절검사를 수행하는 제안하였다. 바이트코드를 이용해 표절검사를 할 경우, 소스코드를 직접 비교하지 않기 때문에, 표절검사를 위해 소스코드를 서버에 업로드 하거나 외부에 공개하지 않아도 된다. 이는 특히, 소스코드 보안이 중요한 곳에서 효율적으로 이용될 수 있다. 그리고 본 논문에서 제안한 바이트코드 표절검사 시스템은 소스코드를 이용해 직접 표절을 검사하기 전에 1차적인 검증도구로 이용할 수 있다.

참고문헌

[1] L. Prechelt, G. Malpohl, and M. Philippsen. Finding plagiarisms among a set of programs with JPlag. *Journal of Universal Computer Science*, 8(11):1016-1038, 2002.

[2] 지정훈, 우균, 조환규. 제한된 프로그램 소스 집합에서 표절 탐색을 위한 적응적 알고리즘. *정보과학회논문지: 소프트웨어 및 응용*, 33(12),1090-1102, Dec, 2006.