효과적인 변이 분석을 위한 C 프로그램 변이 도구 비교: Proteum과 Milu를 사용한 사례 연구

김윤호○ , 김현우, 양웅규, 김문주
한국과학기술원 전산학부
kimyunho@kaist.ac.kr, hyunooudy@gmail.com, wg.yang92@gmail.com, moonzoo@cs.kaist.ac.kr

A Comparative Study of C Program Mutation Tools for Effective Mutation Analysis: A Case Study of Proteum and Milu

Yunho Kim○, Hyunwoo Kim, Woonggyu Yang, Moonzoo Kim
School of Computing, KAIST

요 약

프로그램 변이 분석은 분석 대상 프로그램의 코드를 변형한 프로그램 변이를 활용해 프로그램 변형에 따른 실행 결과 변화를 분석하는 기법이다. 프로그램 변이 분석이 효과적이기 위해선 프로그램 변이 도구가 효과적인 프로그램 변이를 생성할 수 있어야 한다. 예를 들어 생성된 프로그램 변이가 분석 대상 프로그램과 의미가 동일한 무효 변이거나 다른 변이와 의미가 동일한 중복 변이인 경우 변이 분석을 통한 다양한 실행 결과 변화를 볼 수 없기 때문에 변이 분석에 효과적이지 않다. 본 논문에서는 C 프로그램의 효과적인 변이 분석을 위해 변이 도구 Proteum과 Milu를 대상으로 얼마나 효과적인 프로그램 변이를 생성하는지 비교하였다. SIR 벤치마크의 grep, sed 프로그램을 대상으로 수행한 비교 연구 결과 Proteum과 Milu가 생성한 변이 중 각각 평균 54.7%, 82.7%가 유용한 변이였으며 Milu가 더 효과적으로 변이를 생성함을 확인하였다.

1. 서 론

변이 분석(mutation analysis)[1]은 테스트 대상 프로그램의 코드를 다양하게 변형한 다수의 “프로그램 변이” (mutant, 이하 변이)를 생성한 후, 코드 변화에 따른 테스트 실행 결과 변화를 분석하는 소프트웨어 분석 기법이다. 변이 분석에서는 분석대상 프로그램 전반에서 다양한 변이를 생성하기 위해 여러 개의 “변형 연산자”(mutation operator)를 활용하는데, 이 때 각 변형 연산자는 분석대상 프로그램 코드의 특정 패턴마다 특정한 형태의 변형을 발생시킨 변이를 자동으로 생성한다.

변이 분석을 효과적으로 하기 위해선 분석 대상 프로그램과 다른 동작을 수행하는 다양한 변이를 생성하는 것이 중요하다. 예를 들어 생성된 프로그램 변이가 분석 대상 프로그램과 의미가 동일한 무효 변이거나 다른 변이와 의미가 동일한 중복 변이인 경우 변이 분석을 통한 다양한 실행 결과 변화를 볼 수 없기 때문에 변이 분석에 효과적이지 않다. 변이 분석을 수행하려면 생성된 변이 각각에 대해 주어진 테스트 케이스를 실행해야 하기 때문에 효과적이지 않은 변이가 다수 생성되면 변이 분석의 효과가 떨어질 뿐 아니라 변이 분석 시간이 길어지게 된다. 따라서 변이 분석을 수행할 때 효과적인 변이를 다수 생성할 수 있는 변이 도구의 선택이 중요하다.

본 논문에서는 현재 공개되어 사용할 수 있는 오픈 소스 C 프로그램 변이 도구 Proteum[2]과 Milu[3]를 비교하여 각 도구가 얼마나 효과적인 프로그램 변이를 생성하는지 비교하였다. Proteum과 Milu 모두 C 프로그램으로 작성된 소스 코드를 입력으로 받아 변이 프로그램을 생성하는 도구이다. 하지만 Proteum과 Milu가 생성할 수 있는 변이 프로그램의 종류가 서로 다르기 때문에 변이 분석의 효과성과 효율성이 달라지게 된다. 특히 변이 분석의 경우 다수의 프로그램 변이를 대상으로 테스트 케이스를 실행해야 하기 때문에 단순히 많은 수의 변이 프로그램을 생성하는 도구보다 적은 수의 변이 프로그램을 생성하더라도 효과적인 변이 분석을 수행할 수 있는 도구를 선택해야 한다.

Proteum과 Milu가 생성하는 변이 프로그램의 효과를 비교하기 위해 SIR 벤치마크[4]의 grep과 sed 프로그램을 대상으로 비교 연구를 수행하였다. Proteum과 Milu가 지원하는 변형 연산자를 모두 적용하여 생성한 변이 프로그램을 대상으로 테스트를 수행하여 어느 도구에서 생성한 변이 프로그램이 더 효과적인지 비교하였다. 비교 결과 Proteum의 경우 전체 변이 프로그램 중 54.7%가 유용하였고 Milu는 82.7%가 유용한 변이 프로그램이었다.

논문 구성은 다음과 같다. 2장에서는 변이 분석 기법과 관련 연구에 대해서 소개한다. 3장에서는 변이 분석 비교를 위한 실험 환경을 설명하고 4장에서는 비교 실험 결과를 설명한다. 끝으로 5장에서는 본 논문의 결론 및 향후 연구 방향을 제시한다.

본 연구는 한국연구재단 한-아프리카 협력기반조성사업(NRF-2014K1A3A1A09063167) 및 중견연구자지원사업(2016R1A2B4008113), 미래창조과학부 및 정보통신기술진흥센터의 대학ICT연구센터육성 지원사업(IITP-2016-H8501161012000l002) 및 정보통신·방송 연구개발 사업[1711035350, 초소형·고신뢰(99.999%) OS와 고성능 멀티코어 OS를 동시 실행하는 듀얼 운영체제 원천 기술 개발]의 지원으로 수행되었음

2. 변이 분석 기법 및 도구

2.1 변이 분석 기법

변이 분석 기법은 원본 프로그램의 특정 요소 (구문, 표현식, 연산자, 변수/상수 값 등)를 다른 형태로 변경한 후 프로그램 변이를 생성/실행하고 원본 프로그램과 생성된 프로그램 변이의 실행 결과 차이를 분석하는 기법이다. 프로그램 변이는 원본 프로그램의 특정 요소를 미리 정의된 패턴의 다른 요소로 치환하는 방식을 통해 생성되며 프로그램 변이 도구는 분석 대상 프로그램의 모든 구성 요소를 탐색하여 변이를 생성한다. 이 때 변경할 프로그램 구성 요소의 패턴과 해당 패턴을 어떻게 변경하여 변이를 생성할 지 정의한 것을 변형 연산자(mutation operator)라고 하며 변형 연산자를 적용하여 변형된 프로그램 위치를 변경점(mutation point)이라고 한다. 프로그램 변이가 생성되면 원본 프로그램과 프로그램 변이의 실행 결과 차이를 확인하기 위해 주어진 테스트 케이스를 실행한다. 주어진 테스트 케이스를 원본 프로그램과 변이 프로그램에서 실행했을 때 하나 이상의 테스트 케이스에서 원본 프로그램과 프로그램 변이의 실행 결과가 다른 경우 해당 프로그램 변이를 죽은 변이(killed mutant)라고 하며 주어진 테스트 케이스를 실행했을 때 원본 프로그램과 프로그램 변이의 실행 결과가 똑같다면 해당 변이는 살아있는 변이(live mutant)라고 한다.

2.2 효과적인 프로그램 변이

변이 분석의 효과성은 원본 프로그램으로부터 생성되는 프로그램 변이가 얼마나 효과적인지에 따라 결정된다. 생성된 프로그램 변이가 원본 프로그램과 다른 동작을 수행하고 프로그램 변이 사이의 동작이 다를수록 변이 분석에 효과적이다. 반면 생성된 프로그램 변이가 원본 프로그램과 동작이 동일한 무효 변이(equivalent mutant)거나 다른 변이 프로그램과 동작이 같은 중복 변이(redundant mutant)인 경우 변이 분석의 효과가 떨어진다. 뿐만 아니라 효과적이지 않은 변이를 대상으로 테스트를 수행하느라 테스트 비용이 증가하기 때문에 효과적이지 않은 변이를 적게 생성하는 것이 중요하다.

변이 분석에 효과적이지 못한 변이의 종류는 다음과 같다.

- 죽이기 쉬운 변이 프로그램(Easy-to-kill mutant): 해당 변경점을 실행한 테스트 케이스 중 다수(예: 전체 테스트 중 2/3 이상)의 테스트 케이스가 해당 변이 프로그램을 죽일 수 있는 경우 변경점에 대한 실행 유무만 판단하는 커버리지 조건과 효과가 같으므로 변이 프로그램 실행을 통한 동작 변화를 분석하는 변이 분석에 효과적이지 않다.

표 1. 변이 분석 대상 프로그램 정보

대상 프로그램	LoC	테스트 수	분기 커버리지(%)	라인 커버리지(%)
grep-2.0	5956	132	46.0	65.4
Sed-1.17	4085	73	34.8	62.4
평균	5021	103	40.4	63.9

- 무효 변이(equivalent mutant): 프로그램 구문 변경을 통해 생성된 변이 프로그램이 원본 프로그램과 의미적으로 동일한 경우이다. 이 경우 모든 입력 값에 대해서 변이 프로그램과 원본 프로그램의 실행 결과가 동일하기 때문에 변이 분석에 효과적이지 않다.
- 중복 변이(redundant mutant): 서로 다른 두 프로그램 변이가 모든 입력 값에 대해 동일한 실행 결과를 갖는 변이이다. 중복 변이의 경우 변이에 따른 동작 변화가 서로 갖기 때문에 변이 분석에 효과적이지 않다.

2.3 변형 연산자

변형 연산자는 변경을 수행할 변경점과 해당 변경점에서 치환을 수행하여 변이 프로그램을 생성하는 방법을 정의하는 연산자이다. Agrawal 외 8인이 수행한 기존 연구[5]에서는 변형 연산자가 적용되는 위치에 따라 구문 변이(statement mutation), 연산자 변이(operator mutation), 변수 변이(variable mutation), 상수 변이(constant mutation)의 4 종류로 변형 연산자를 분류하고 각 분류마다 적용되는 코드 패턴 및 변경될 코드에 따라 변형 연산자를 정의하였다.

2.4 C 프로그램 변이 분석 도구

본 논문에서는 현재 오픈 소스로 공개되어 활용할 수 있는 C 프로그램 변이 분석 도구인 Proteum과 Milu를 선택하여 비교하였다.

2.4.1 Proteum

Proteum[2]은 브라질 University of São Paulo 대학의 Marcio Eduardo Delamaro와 JoseCarlos Maldonado 가 1996년 처음 개발하였으며 2001년 최신 버전인 Proteum 2.0 버전이 발표되었다. Proteum은 Agrawal 외 8인이 정의한 C 프로그램 변형 연산자 중 75개의 변형 연산자를 지원하여 가장 많은 종류의 변형 연산자를 지원한다.

2.4.2 Milu

Milu[3]는 University of College London의 Yue Jia가 개발한 C 프로그램 변이 분석 도구이다. 2008년 처음 개발되었으며 현재 3.2 버전이 가장 최신 버전이다. Milu는 23개의 변형 연산자를 지원한다.

표 2. 변이 분석 결과

대상 프로그램	Proteum 생성 변이						Milu 생성 변이					
	중복없는 변이				중복 변이	전체 변이	중복없는 변이				중복 변이	전체 변이
	죽은 변이		살아있는 변이				죽은 변이		살아있는 변이			
	죽이기 쉬운 변이	죽이기 어려운 변이	변경점 실행	변경점 실행 못함			죽이기 쉬운 변이	죽이기 어려운 변이	변경점 실행	변경점 실행 못함		
grep	33228 (9.9%)	43046 (12.8%)	66454 (19.7%)	83382 (24.8%)	110672 (32.9%)	336782 (100.0%)	6481 (16.5%)	5449 (13.9%)	17674 (45.0%)	9536 (24.3%)	120 (0.3%)	39260 (100.0%)
sed	21141 (8.3%)	8768 (3.4%)	44052 (17.3%)	77765 (30.5%)	102944 (40.4%)	254670 (100.0%)	4381 (17.7%)	1180 (4.8%)	7516 (30.3%)	11655 (47.0%)	47 (0.2%)	24779 (100.0%)
평균	27184.5 (9.2%)	25907.0 (8.8%)	55252.0 (18.7%)	80573.5 (27.2%)	106808.0 (36.1%)	295726.0 (100.0%)	5431.0 (17.0%)	3314.5 (10.3%)	12595.0 (39.3%)	10595.5 (33.1%)	83.5 (0.3%)	32019.5 (100.0%)

3. 변이 테스트 도구 비교 실험 환경

Proteum과 Milu가 효과적인 프로그램 변이를 생성하는지 비교하기 위해 SIR 벤치마크[4]의 grep 과 sed 프로그램을 대상으로 변이 분석을 수행하였다. 표 1은 프로그램의 크기와 사용한 테스트 케이스 수 및 테스트 커버리지를 나타낸다. 사용한 Proteum과 Milu 버전은 각각 최신 버전인 Proteum 2.0과 Milu 3.2 버전을 사용하였으며 Proteum의 75개 변형 연산자와 Milu의 23개 변형 연산자를 사용하여 변이 프로그램을 생성하였다.

생성된 변이 프로그램의 효과성을 확인하기 위해 생성된 변이 프로그램 중 2.2절에서 설명한 효과적이지 못한 변이 프로그램의 비율을 비교하였다. 효과적이지 못한 변이 프로그램 중 죽이기 쉬운 변이 프로그램은 해당 변이의 변경점을 실행한 테스트 케이스 중 2/3 이상이 죽인 변이 프로그램으로 정의하였으며 중복 변이는 서로 다른 변이 중 컴파일 후 동일한 바이너리를 생성한 변이의 수로 측정하였다. 무효 변이의 경우 생성된 각각의 변이와 원본 프로그램의 의미 구조를 직접 분석해야 하기 때문에 비교하지 못했다.

4. 변이 테스트 도구 비교 실험 결과

표 2는 변이 분석 결과를 나타낸다. Proteum과 Milu 는 평균적으로 295726.0개와 32019.5개의 변이를 생성하였다. Proteum이 지원 연산자의 수가 더 많기 때문에 Milu 생성 변이 수의 9.2 배의 변이를 생성할 수 있었다. Proteum이 생성된 변이의 평균 36.1%의 변이가 중복 변이고 9.2%의 변이가 죽이기 쉬운 변이이다. 반면 Milu의 경우 0.3%의 변이가 중복 변이고 17.0%의 변이가 죽이기 쉬운 변이였다.

Proteum에서 생성된 중복 변이를 분석한 결과 상수 값과 변수를 다른 상수로 변형하는 변형 연산자에서 전체 중복 변이의 74.5%를 생성하였다. Proteum에서 상수 값이나 변수를 다른 상수로 치환할 때 치환할 다른 상수 값에 enum 타입으로 정의된 값도 사용되는데 서로 다른 enum 타입이 동일한 정수 값을 가질 수 있기 때문에 다수의 중복 변이를 생성하였다. 반면 Milu의 상수 변이 변형 연산자는 상수 값을 0, -1, 1, 원본 값+1, 원본 값-1 로 치환하고 프로그램에서 사용한 다른 값으로 치환하지 않기 때문에 Proteum 보다 적은 중복 변이가 생성되었다.

5. 결론 및 향후 연구

본 논문에서는 C 프로그램의 변이 도구 Proteum과 Milu를 대상으로 효과적인 변이가 얼마나 생성되는지 비교 분석 연구를 수행하였다. SIR 벤치마크의 grep과 sed 프로그램을 대상으로 변이 분석을 수행한 결과 Proteum에서 생성된 변이 중 54.7%의 변이가 효과적인 변이였고 Milu에서 생성된 변이 중 82.7%의 변이가 효과적인 변이로 Milu가 더 효과적으로 변이를 생성하는 것을 확인하였다. 향후 연구로는 살아있는 변이 중 무효 변이가 얼마나 존재하는지 비교하는 연구를 수행할 예정이다. 무효 변이를 정확하게 알아내는 것은 정지 결정 문제(halting problem)이기 때문에 휴리스틱 기법을 사용하여 무효 변이를 탐지하고 비교하고자 한다.

참고문헌

[1] Jia and Harman, An Analysis and Survey of the Development of Mutation Testing, TSE 37(5), 2011

[2] Maldonado et al., Proteum: A family of tools to support specification and program testing based on mutation, Mutation testing for the new century, 2001

[3] Jia and Harman, A Customizable, Runtime-Optimized Higher Order Mutation Testing Tool for the Full C Language, TAIC PART, 2008

[4] Do et al., Supporting controlled experimentation with testing techniques: An infrastructure and its potential impact, ESE, 10(4), 2005

[5] Agrawal et al., Design of mutant operators for the C programming language, SERC-TR-120-P, Purdue University, 1989