두 가지 기호 실행 기술들의 융합하는 방법들

(Methods for Combining Two Symbolic Execution Techniques)

김문수*, 차수영
Moonsu Kim*, Sooyoung Cha

성균관대학교 소프트웨어학과
Department of Computer Science and Engineering, Sungkyunkwan University
kimmail99@g.skku.edu, sooyoung.cha@skku.edu

요 약

기호 실행(Symbolic Execution)은 잘 알려진 하나의 소프트웨어 테스팅 방법론이다. 기호 실행은 프로그램의 실행 경로들을 체계적으로 탐색하여 높은 코드 커버리지 및 다양한 오류 탐지를 목표로 한다. 하지만 실제 프로그램을 테스트할 때, 기호 실행은 여전히 경로-폭발 문제, 높은 제약식 풀이 비용 등 도전적인 과제 때문에 더 높은 테스팅 성능을 성취하지 못하고 있다. 이를 완화하기 위해, 기호 실행 연구자들은 탐색 전략, 공간 축소 전략, 파라미터 조정 전략 등 다양한 개별적인 기술들을 고안하였다. 그러나 다양한 기호 실행 기술들을 동시에 사용하는 방법론은 연구가 되지 않았다. 본 논문은 두 가지 대표적인 기호 실행 기술들을 동시에 사용하는 방법론 및 그 가능성에 대해 모색한다. 구체적으로, 본 연구는 기호 실행의 최신 기술 두 가지 "탐색 전략"과 "외부 파라미터 조정 전략"을 융합하는 방법들과 그 융합의 효과성을 확인해보았다. 본 논문의 실험 결과는 개별적으로 탐색 전략 또는 외부 파라미터 조정 전략을 사용하는 것보다 두 가지를 융합하는 방법이 코드 커버리지와 버그 검출 성능 면에서 평균적으로 가장 우수한 성능을 보임을 관찰했다.

1. 서 론

현대 소프트웨어의 복잡성 증가에 따라, 시스템 결함의 위험도 함께 커지고 있다. 이를 완화하기 위한 대표적 자동화 기법인 기호 실행(symbolic execution)은 입력을 기호 변수로 치환하고 SMT 솔버로 분기 조건을 분석해 다양한 실행 경로를 탐색한다. 그러나 분기가 늘어날수록 경로와 상태가 기하급수적으로 불어나는 경로 폭발 문제가 발생해 대규모 시스템에는 적용이 쉽지 않다.

이 한계를 줄이기 위해 두 가지 보완 기술이 제안되어 왔다. 첫째, 우선 탐색할 상태를 선택해 불필요한 분기 확장을 억제하는 탐색 전략 [1]. 둘째, SMT 타임아웃이나 메모리 한도 같은 실행 환경 파라미터를 동적으로 조정해 제약식 풀이 비용을 줄이는 외부 파라미터 조정 [2]이다. 두 기법은 각각 의미 있는 성능 개선을 달성했지만, 상호 보완적 특성을 결합해 시너지를 체계적으로 검증한 연구는 부족하다.

본 연구는 두 기법을 통합하는 세 가지 프레임워크—(i) 병렬 실행하며 상호 피드백을 주는 병렬 융합, (ii) 실행 시간을 절반씩 나누는 균형 융합, (iii) 주기적으로 두 전략을 교대로 적용하는 교대 융합—를 설계하였다. GNU C 벤치마크 다섯 종을 대상으로 10 시간 동안 수행한

표 1. 5개 벤치마크에서 각 기법이 36000 초 동안 달성한 최고 분기 커버리지

Benchmark	탐색 전략	Parameter 조정 전략	KLEE	병렬 융합	균형 융합	교대 융합
Combine-0.4.0	930	934	403	1190	801	1160
Diffutils-3.7	2217	1780	732	1820	1947	2401
Gcal-4.1	3120	3930	2017	3300	3801	4164
Ls-8.32	1802	1702	901	1801	1751	1897
Trueprint-5.4	934	834	445	807	1017	806

표 2. 5개 벤치마크에서 융합 모델의 단일 기법 대비 상대 커버리지 향상률(%)

융합/단일(%)	병렬/KLEE	병렬/탐색	병렬/Param	균형/KLEE	균형/탐색	균형/Param	교대/KLEE	교대/탐색	교대/Param
Combine	195.3	28.0	27.4	98.8	-13.9	-14.2	187.8	24.7	24.2
Diffutils	148.6	-17.9	2.2	166.0	-12.2	9.4	228.0	8.3	34.9
Gcal	63.6	5.8	-16.0	88.4	21.8	-3.3	106.4	33.5	6.0
Ls	99.9	-0.1	5.8	94.3	-2.8	2.9	110.5	5.3	11.5
Trueprint	81.3	-13.6	-3.2	128.5	8.9	21.9	81.1	-13.7	-3.4

실험 결과, 교대 융합은 모든 단일 전략보다 높은 코드 커버리지를 달성했다. 이 성과로 본 연구는 기술 결합이 기호 실행의 경로 폭발 문제와 제약 풀이 비용을 동시에 완화할 수 있는 가능성을 입증하였다.

2. 상호 보완 전략

다음은 각 결합 모델의 구조와 공유 메커니즘에 대한 설명이다.

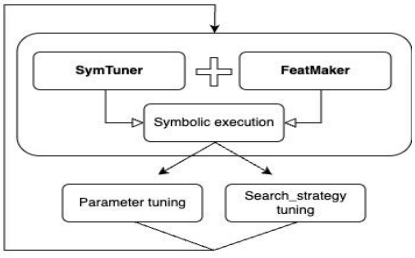

그림 1. 병렬 융합 모델

병렬 융합은 탐색 전략과 외부 파라미터 조정 전략이 병렬로 실행되며, 각각의 업데이트 함수에서 상대 모듈의 출력을 참조하여 학습한다. 탐색 전략은 상태 s의 특성 벡터 F(s)와 가중치 W를 내적하여 탐색 점수 Priority(s)를 계산한다. 이때 사용되는 상태 샘플은 외부 파라미터 조정 전략이 현재 설정한 외부 파라미터 P_t 환경 하에서 생성된 것으로, 간접적으로 상태 탐색 전략의 학습에 영향을 미친다.

외부 파라미터 조정 전략은 다양한 파라미터 조합 $p \in P$에 대해 KLEE를 실행하고, 그 실행 결과로 얻어진 고득점 상태들의 점수 Priority(s)를 참조하여 조합별 성능을 평가한다. 이때 평가 점수는 다음과 같이 정의된다:

$$Score\_param(p) = C(p) - \alpha \cdot E(p)$$

여기서 C(p)는 커버리지, E(p)는 실패율이다.

결과적으로, 탐색 전략은 외부 파라미터 조정 전략의 파라미터 설정을 반영하여 상태를 평가하고, 외부 파라미터 조정 전략은 탐색 전략의 상태 점수를 바탕으로 파라미터를 학습한다.

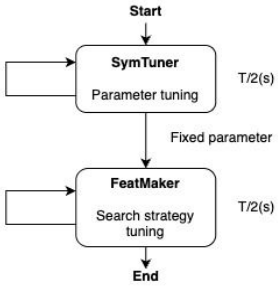

그림 2. 균형 융합 모델

균형 융합은 전체 실행 시간 T을 절반으로 나누어, 전반부 [0, T/2]는 외부 파라미터 조정 전략을, 후반부 [T/2, T]는 기반 탐색을 수행하는 구조이다.
전반부 과정에서 탐색 성능이 가장 높은 파라미터를 학습하고, 후반부에는 선정된 파라미터를 고정하여 KLEE를 실행하며, 상태 특성 벡터 F(s)를 생성하여 탐색 점수를 계산한다.

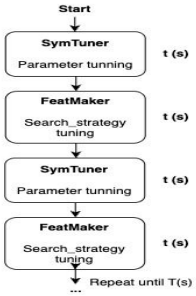

그림 3. 교대 융합 모델

교대 융합은 고정된 시간 간격 τ (20 execution steps)를 기준으로 탐색 전략와 외부 파라미터 조정 전략을 교대로 실행하며, 각 주기에서 얻은 결과를 다음 모듈에 피드백하는 구조이다.
외부 파라미터 조정 전략에서 제시한 P_tuned 설정 하에서 상태 특성 벡터를 계산하고, 이를 기반으로 탐색 점수 를 갱신한다. 학습된 상태 순위는 다음 주기에서 외부 파라미터 조정 전략이 우선 조정해야 할 경로의 힌트로 활용된다.

3. 실험 결과

본 연구에서 제안한 세 가지 결합 모델(병렬, 균형, 융합)과 기존 자동화 기법(탐색, 외부 파라미터 조정), 그리고 baseline 도구인 KLEE를 비교 평가하였다. 실험은 심볼릭 실행 성능 평가에 널리 사용되는 5개의 GNU 오픈소스 C 프로그램(Combine, diff, gcal, ls, trueprint)을 벤치마크로 활용하였다.
각 벤치마크는 동일한 자원 환경에서 36,000초(10시간) 동안 심볼릭 실행을 수행하였으며, 탐색 성능은 커버된 분기 수(branch coverage)를 기준으로 평가되었다.
전체 벤치마크에서의 평균 커버리지는 교대 융합이 가장 높았으며, 평균적으로 KLEE 대비 131.8%, 탐색 전략 대비 15.8%, 외부 파라미터 조정 전략 대비 13.6% 더 많은 분기를 탐색하였다. 균형 융합은 탐색 전략보다 3.5%, 외부 파라미터 조정 전략보다 1.5% 높은 성능을 보였다. 병렬 융합은 KLEE 대비 98.3% 높은 성능을 기록했지만, 탐색 전략 및 외부 파라미터 조정 전략에 비해서는 각각 0.9%, 2.9% 낮은 평균 커버리지를 보였다.
교대 융합 모델은 Gcal, Diffutils, Ls 등 3개 벤치마크에서 최상위 성능을 기록했다. 특히 Gcal(캘린더 유틸리티) 벤치마크에서는 KLEE 대비 두 배 이상의 커버리지를 달성하였다.

4. 결론 및 향후 연구

통합 모델들은 기존의 단일 전략 기반 도구에 비해 전반적으로 높은 분기 커버리지를 달성했다. 이는 상태 기반 탐색과 실행 환경 조정을 함께 고려하는 통합 접근이 성능 향상에 실질적인 기여를 할 수 있음을 보여준다.
아울러, 결합 방식에 따라 탐색 특성과 성능 차이가 관찰되었으며, 이는 사용자 목적이나 실행 환경에 따라 전략을 동적으로 선택하거나 조정할 수 있는 적응형 프레임워크 설계로의 확장 가능성을 시사한다.

참고 문헌

[1] Jaehan Yoon and Sooyoung Cha. 2024. FeatMaker: Automated Feature Engineering for Search Strategy of Symbolic Execution. Proc. ACM Softw. Eng., Vol. 1, No. FSE, Article 108 (July 2024), 22 pages.
[2] Sooyoung Cha, Myungho Lee, Seokhyun Lee, and Hakjoo Oh. 2022. SymTuner: Maximizing the Power of Symbolic Execution by Adaptively Tuning External Parameters. In Proceedings of the 44th International Conference on Software Engineering (ICSE '22). ACM, New York, NY, USA, 12 pages.
[3] Cadar, Cristian, Daniel Dunbar, and Dawson R. Engler. "Klee: Unassisted and automatic generation of high-coverage tests for complex systems programs." OSDI. Vol. 8. p. 209-224.