OOPSLA 2026 참가 후기

박성민 (Sungmin Park)

2026 년 10 월 4 일 - 9 일

미국 캘리포니아주 오클랜드



1 학회 및 논문

2026 년 10 월 4 일부터 9 일까지 미국 캘리포니아주 오클랜드에서 열린 SPLASH/ISSTA 2026 에 참가하였다. SPLASH/ISSTA 는 프로그래밍 언어 및 소프트웨어 공학 분야의 다양한 연구를 공유하는 학회로, 이번에는 OOPSLA 에서 발표된 여러 흥미로운 연구들을 접할 수 있었다.

이번 학회에서 특히 흥미로웠던 논문은 **Block Tests** 와 **Spatial and Temporal Decomposition for Faster Translation Validation 이었다**.

Block Tests

Block Tests 는 기존의 unit test 와 inline test 사이의 간극을 메우기 위해 새로운 테스트 단위인 block test 를 제안하는 논문이다.

일반적인 unit test 는 메서드 단위로 프로그램을 검증하기 때문에 메서드 내부의 특정 코드에 도달하기 어렵거나 오류가 최종 결과까지 전파되지 않는 경우 버그를 발견하지 못할 수 있다. 반면 inline test 는 개별 문장을 직접 검증할 수 있지만, 여러 문장에 걸친 데이터 흐름이나 제어 흐름을 테스트하기 어렵다는 한계가 있다.

이 논문에서는 메서드 내부의 임의의 코드 조각을 독립적으로 검증하는 block test 를 제안하고, Java 에서 이를 작성하고 실행할 수 있는 Block-test Development Kit(BDK)를 구현하였다. 연구진은 146 개 오픈소스 프로젝트에서 1,012 개의 block test 를 작성하였으며, 기존 unit test 가 발견하지 못한 많은 mutant 를 탐지할 수 있음을 확인하였다.

Block test 는 검증할 코드의 범위와 입력, 기대 결과를 명시한다. 여러 문장을 하나의 단위로 다룰 수 있으므로, 문장 사이의 값 전달이나 분기까지 함께 검증할 수 있다는 점이 핵심이다.

실험 대상인 346 개 코드 조각 중 58.7%는 개발자가 작성한 unit test 로 실행되지 않았다. 또한 unit test 가 탐지하지 못한 mutant 9,554 개 중 4,418 개를 block test 가 탐지했다. 이는 단순한 실행 범위 확대뿐 아니라 오류를 판별하는 능력도 보완할 수 있음을 보여준다.

특히 테스트의 단위를 메서드나 문장으로 제한하지 않고, 개발자가 관심 있는 코드 영역으로 확장한다는 아이디어가 인상적이었다. 기존 테스트에서 충분히 검증되지 않는 코드 영역을 보다 직접적으로 테스트할 수 있다는 점에서 흥미로운 연구였다.

Spatial and Temporal Decomposition for Faster Translation Validation

Spatial and Temporal Decomposition for Faster Translation Validation 은 컴파일러가 프로그램을 변환하기 전후에 의미가 동일하게 유지되는지를 검증하는 translation validation 의 성능을 개선하는 논문이다.

Translation validation 은 컴파일러 전체의 정확성을 증명하는 대신, 특정 컴파일 결과가 원본 프로그램과 의미적으로 동일한지 검증하는 방법이다. 하지만 복잡한 프로그램 전체를 대상으로 동등성을 검증하는 데 많은 시간이 필요하다는 문제가 있다.

이 논문에서는 검증 문제를 공간적(spatial), 시간적(temporal) 두 관점에서 분해하는 방법을 제안한다. 공간적 분해는 컴파일러가 제공하는 정보를 활용하여 전체 프로그램의 동등성을 보장할 수 있는 작은 부분 프로그램들을 추출하는 방식이다. 시간적 분해는 컴파일러의 여러 최적화 패스를 동적으로 묶어 효율적으로 검증하는 방식이다.

구현체인 DESTIVA 는 Alive2 를 확장하여 LLVM 이 추가·삭제·수정한 명령어를 추적한다. 여기에 가벼운 프로그램 분석을 더해 검증에 필요한 부분을 추출하고, 변환과 무관한 코드를 제외하여 해결해야 할 제약의 규모를 줄인다.

실험에서는 전체 3,025 개 벤치마크 중 기존 두 방식이 모두 시간 초과로 검증하지 못한 298 개를 추가로 검증하였다. 기존 패스별 검증에 10 초 넘게 걸리던 문제 집합에서는 두 분해 방식을 결합하여 약 2.4 배의 성능 향상을 보였다.

다만 분해 자체에도 비용이 들어 작은 문제에서는 오히려 느려질 수 있어, 기존 검증 방식과 병행하는 전략도 평가하였다. 공간적 분해의 정확성은 컴파일러가 제공하는 변경 기록이 정확하다는 전제에 의존한다.

컴파일러가 이미 가지고 있는 정보를 활용하여 복잡한 검증 문제를 더 작은 문제로 분해한다는 접근이 흥미로웠다. 특히 검증의 정확성을 유지하면서 실용적인 성능을 개선했다는 점이 인상적이었다.

2 함께 살펴볼 만한 논문

Detecting Flaky Tests by Controlling Nondeterministic API Behavior

Detecting Flaky Tests by Controlling Nondeterministic API Behavior 는 같은 코드를 테스트해도 실행할 때마다 성공과 실패가 달라지는 flaky test 를 찾아내는 연구이다.

논문에서 제안하는 ChaosAPI 는 시스템 시간, 동시성 등과 관련된 Java 표준 라이브러리 API 의 입력과 반환값을 명세가 허용하는 범위에서 바꾸어 비결정적 동작을 유도하고, 테스트가 실패하는지를 관찰한다. 평가에서는 테스트를 반복 실행하는 방법보다 더 많은 flaky test 를 효율적으로 탐지하였다. 실패가 우연히 발생하기를 기다리지 않고 원인에 직접 개입한다는 점이 흥미로웠다.

Automated Debugging of Datalog Programs

Automated Debugging of Datalog Programs 는 Datalog 프로그램에서 오류가 있는 규칙을 자동으로 찾아내는 연구이다. 거대한 증명 트리를 사람이 일일이 살피는 대신, 도출된 사실을 테스트 케이스로 보고 그 증명 트리에 사용된 규칙을 실행 정보로 활용한다.

정상 결과와 잘못된 결과의 증명 트리를 함께 비교하고, 오류 결과에 자주 관여하는 규칙의 의심도를 계산하는 SBFL 기법을 적용한다. Doop 기반의 96 개 디버깅 사례에서 가장 좋은 지표는 오류 규칙을 첫 번째 후보로 찾는 비율 87.5%를 보였다. 개별 증명 트리를 따라가는 문제를 여러 결과의 통계적 비교로 바꾼다는 점이 흥미로웠다.



3 음식

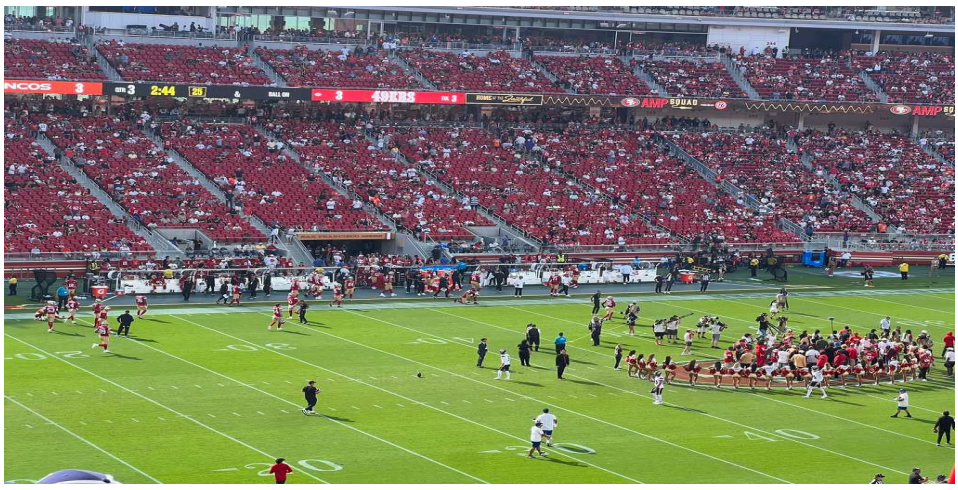
미국에서 먹었던 음식들은 전반적으로 맛있었다. 특히 고기나 튀김을 활용한 음식들이 많았고, 양도 상당히 푸짐했다. 다만 대부분의 음식이 한국 음식에 비해 매우 기름져서 며칠 동안 먹다 보니 다소 부담스럽기도 했다. 맛은 있었지만 담백한 음식이 생각나는 경우도 많았다.



4 관광

학회 일정 외에는 미국의 미식축구 경기를 관람하였다. 평소 한국에서는 미식축구를 직접 볼 기회가 거의 없었기 때문에 특별한 경험이었다.

경기장의 규모와 관중들의 열기가 인상적이었으며, 경기뿐 아니라 관중들의 응원과 현장 분위기까지 즐길 수 있었다. 미국에서 미식축구가 얼마나 인기 있는 스포츠인지 직접 체감할 수 있었고, 학회 참가와 더불어 기억에 남는 경험이었다.



5 마무리

이번 OOPSLA 2026 참가를 통해 프로그래밍 언어와 소프트웨어 테스팅, 프로그램 검증 분야의 다양한 연구를 접할 수 있었다. 흥미로운 논문들을 살펴보고 미국의 음식과 문화를 경험할 수 있었던 뜻깊은 시간이었다. 이번 학회에 참가할 수 있도록 지원해 주신 교수님께 진심으로 감사드린다.