ASE 2025 Conference Report

Jungwoong Kim | PLRG

2025.11.17 – 2025.11.19

Seoul, Republic of Korea

Figure 1: Where's Wally? - ASE'25

1. Introduction

ASE(IEEE/ACM International Conference on Automated Software Engineering)는 소프트웨어 공학 분야의 세계적인 학회로, 소프트웨어 시스템에 대한 분석, 구현, 테스팅 등 산업에 영향을 주는 다양한 분야들에 대해 학계와 기업이 상호작용하는 자리이다. 올해는 서울에서 진행되었으며, 박지혁 교수님께서 지원해주신 덕분에 논문을 제출하지 않았지만 학회에 참여할 수 있었다. 이 자리를 빌려 박지혁 교수님께 감사의 말씀을 드린다.

학회는 11.16(일)부터 11.20(목)까지 진행되었으며 우리 연구실은 월요일부터 수요일까지의 Main Conference 에 참여하였다. 오전 9 시 키노트로 시작하여 오전 11 시부터 오후 3 시 반까지 Research Paper 발표 세션을 듣고, 3 시 반부터는 Poster Core Time 으로 발표자들과 직접 소통할 수 있는 시간으로 구성되었다. 아침 7 시에 일어나서 부지런히 출발하여 9 시 전까지 도착하고, 모든 일정을 마친 뒤 집에 저녁 9 시 넘어서 들어가기를 3 일 내내 하다 보니 체력적으로 조금 부담되긴 했지만 그만큼 많은 것들을 배우고 얻어가는 시간이었다.

첫 국제 학회 참여다 보니 첫날은 특히 긴장되었던 것 같다. 지하철에서 내릴 때까지는 한국이었는데 학회장에 들어가니 해외에 온 듯한 분위기였다. 학회장 분위기에 적응하는 과정이나 타임 테이블을 보면서 발표장 사이를 오가는 것들이 초반에는 어색했지만, 적응되고 나니 발표도 잘 들리고 중간중간 연구실 형들과 교수님과 이야기 나누고 다시 각자 듣고 싶은 발표로 흩어지는 것도 재미있었다. 물론 주제가 너무 생소하거나 이해되지 않는 발표들도 있었다. 그리고 처음에는 포스터에 가서 질문하려면 우선 발표를 듣고 논문 내용을 이해한 뒤 가야 되나 싶었다. 하지만 걱정했던 것보다 학회의 분위기는 친절했다. 포스터에서 궁금한 것을 물어보면 잘 대답해주었고, 심지어 발표때보다 더 자세하게 설명해주는 분들도 있어서 다행이었다.

학회에서 얻은 것들이 있다면 그 중에 하나는 ‘논문이면 이런 걸 해야되구나’ 라는 현실적인 감각인 것 같다. 사실 연구실에 들어와서 연구적인 뭔가는 제대로 한 적이 없기 때문에 ‘so what – 그래서 뭘 했는데’에 대한 질문을 많이 하지 않았던 것 같다. 발표들을 보면서 그 ‘so what’이 ‘버그를 몇 개 찾았어요’ 이든, ‘속도가 얼마나 개선되었어요’ 이든, 아니면 ‘지금 현실에는 이런 문제들이 있어요’ 가 되었든 각자 어떤 노력을 해왔고 그 노력으로 ‘so what’을 증명해내는 것이라는 것을 느낄 수 있었다. 여름방학동안 살펴보던 주제인 polyfill 에 대입해 생각해보았는데, 그 ‘so what’이 제대로 서있지 않다는 느낌을 받았다. 그런 부분에서 더 많이 고민을 하게 되고 연구에 동기부여가 되었다는 점에서 학회 참여가 도움이 되었다 느껴진다.

개인적인 이야기는 이 정도에서 맺음 짓고, 학회에서 기억에 남는 논문과 발표를 소개하고자 한다.

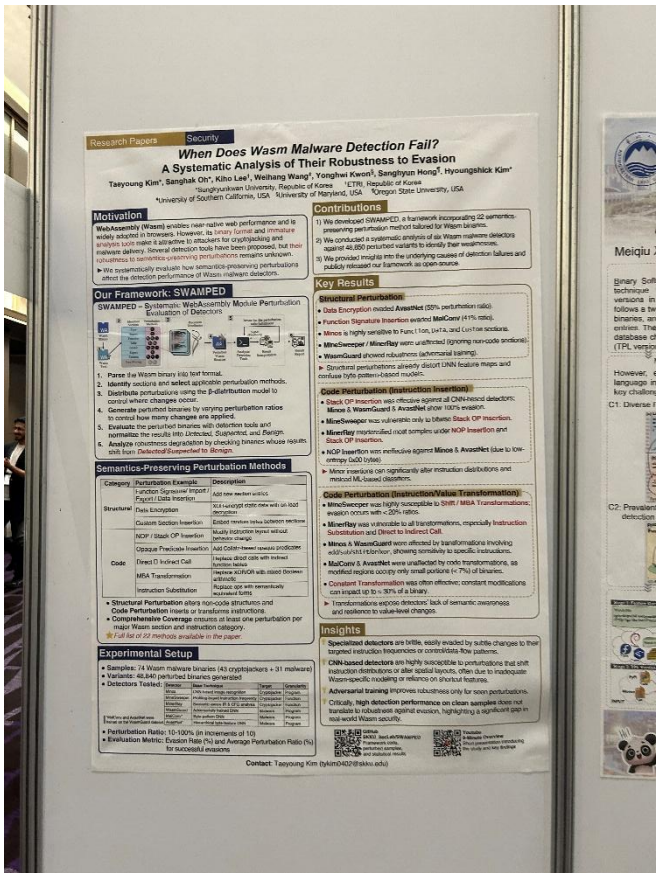
2. Research

2-1. Finding Bugs in MLIR Compiler Infrastructure via Lowering Space Exploration

MLIR 의 Dialect 간 변환(conversion pass)과 최적화 경로(optimization pass)에 존재하는 오류를 찾는 방법을 제시한 논문이다. 공통된 IR 에서 최적화를 하는 일반적인 컴파일러 시스템과 달리 MLIR 은 중간 단계들을 사용자가 선택할 수 있기 때문에 어떻게 조합하는가에 따라 다양한 lowering path 가 존재한다. 이 lowering path 중 특정 path 에서 다른 결과(논문에선 동적으로 같은 입력에 대해 다른 출력을 내는 경우를 내놓는다면 해당 path 를 구성하는 단계에 오류가 있음을 알 수 있다. lowering path 중 유의미한 path 를 효율적으로 탐색하는 방법을 제시하였고, 실제로 25 개의 confirmed/fixed bug 를 찾아냈다.

논문이 하고자 하는 바(중간 단계가 많으니 잘 섞어서 오류 탐색)는 어떻게 보면 매우 단순해서 발표만 들어도 어떤 것을 목표했는지 이해할 수 있었다. 아이디어는 간단해 보여도 실제 오류를 찾기 위해 여러 엔지니어링이 들어갔을 것이고, 그 과정에서 찾은 효율적인 알고리즘이 이 논문의 novelty 이지 않을까 싶다.

2-2. When Does Wasm Malware Detection Fail? A Systematic Analysis of Their Robustness to Evasion



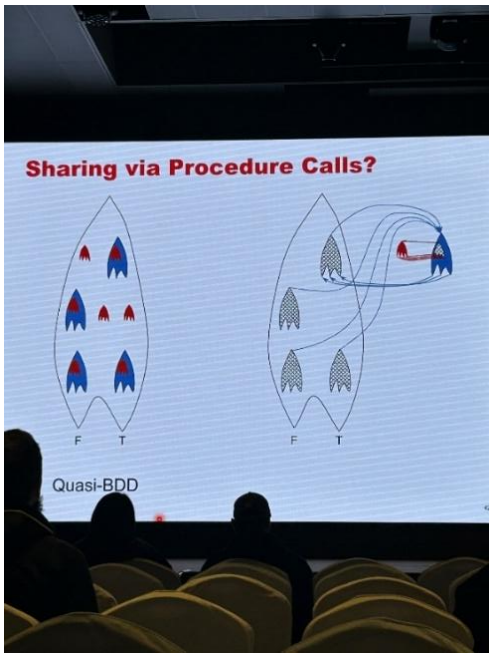
WASM 기반 악성코드 탐지 소프트웨어들이 적용된 난독화 종류들에 따라 실패하는 경우와 원인을 종합적인 진행한 현상 분석 연구였다. Semantic 을 유지하는 선에서 22 가지 perturbation 을 정의하고, 74 개의 악성코드 샘플과 6 가지 탐지 소프트웨어에 대해 perturbation 을 무작위적으로 적용한 샘플을 탐지하도록 하여 어떤 경우 실패하는지 분석하였다. Perturbation 을 적용할 때 위치에 대한 경향성을 제공하는 방법으로 beta distribution 을 사용했다는 점이 신기했다.

조사하고자 하는 대상에 대한 체계적인 틀 제시, 해당 틀에 맞는 난독화 샘플 제작, 각 샘플들에 대해 악성코드 탐지가 실패하는 경우와 그 원인분석까지 전체적인 플로우가 적절한 논문이었다는 느낌이 들었다. 또한, 악성코드 탐지 소프트웨어를 개선하거나 새롭게 개발할 때 참고할 수 있다고 생각되었고, 연구 이후의 단계들이 명확히 보이는 점이 좋다고 느껴졌다.

3. Keynote

키노트에서는 학회에서 발표된 논문들과는 조금 결이 다른 내용들의 발표가 있었다. 월요일과 수요일의 키노트를 들었는데 둘 다 인상적이어서 여기에 소개한다.

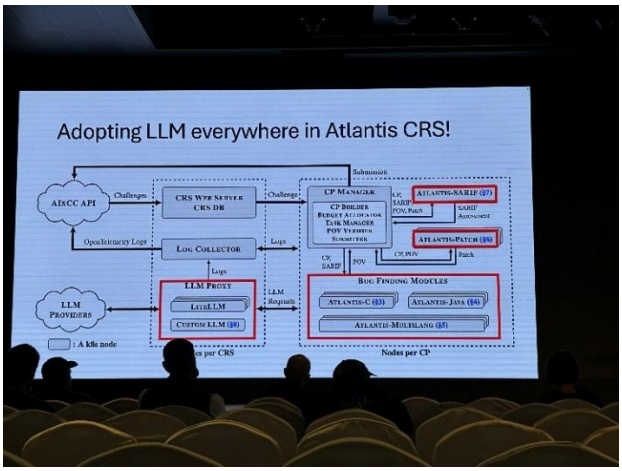
3-1. We Will Publish No Algorithm Before Its Time (CFLOBDD)



ASE'25 Main Conference 의 첫 발표였는데, 말하자면 그래프나 벡터, 행렬의 형태와 연산이 특정 조건을 만족하는 경우 시간/공간적으로 훨씬 높은 효율을 보이는 데이터구조로 압축할 수 있다는 것이었다.

내용도 내용이었지만, CFLOBDD 를 응용할 분야가 없어 10 년 이상을 가만히 두다가 양자 알고리즘이 나온 뒤에 빛을 받아 논문으로 발표했다는 점이 놀라웠다. 첫 아이디어부터 논문까지 25 년이 걸렸다고 한다.

3-2. Hyperscale Bug Finding and Fixing: DAPRA AIxCC



올해 AIxCC 를 우승한 Team Atlanta 의 1 등을 향한 기록이자 이론과 현실의 차이를 느끼게 한 발표였다. 발표에서 인상적인 부분을 아래에 정리해두었다.

- 학계의 분석은 거대 프로젝트 (e.g. linux kernel) 에서는 적용이 불가능하다. 오류 탐지, 원인 분석, 코드 패치까지의 과정을 LLM 이 훨씬 잘한다.

- $120 면 Repository 1 개를 검사할 수 있다. 24 시간 연중무휴에다가 사람보다 훨씬 싸다.

- LLM 은 엄청난 능력을 가졌지만 다루기 까다롭다. LLM 의 무작위성을 극복하고, 능력을 필요한 곳에 집중하도록 하는 prompting 과 장애 발생 시 오류 복구할 수 있는 시스템 설계가 중요하다.

- '안정성을 유지한 채로 성능을 극한으로 끌어올리기' 테크닉들이 신기했다. 자체 벤치마크를 만들어 LLM 의 신규 버전이 나와도 벤치마크를 통과하지 못하면 사용하지 않는다는 과정이 소개되었다.

4. One Liners

집에 돌아가면서 그날의 기억나는 발표/논문을 한두 문장정도로 짧게 정리한 내용들이다.

[11.17] Agents in the Sandbox: End-to-End Crash Bug Reproduction for Minecraft

- 이런 주제로 논문이 된다고? 라는 신기함. 게임과 학회는 거리가 멀어 보였는데 그 선입견을 없앰.

- Minecraft 같은 시뮬레이션 게임에서 가능하면 Physical AI 로 확장 가능하지 않을까 싶었다.

[11.17] It's Not Easy Being Green: On the Energy Efficiency of Programming Languages

- 잘못된 연구를 반박하려면 치밀한 문제정의/실험이 필요하다.

- 멀티코어, 메모리 지역성을 최대로 활용하는 것이 최적의 에너지 효율을 내고, 결론적으로 지금까지 하던 엔지니어링 그대로 하는 것이 최적이다.

[11.17] Non-termination Witnesses and their Validation

- 탐색(오류가 있는가?)도 중요하고, localization(오류가 어디에 있는가?)도 중요함.

- 정적분석의 응용 사례를 볼 수 있었다. 소프트웨어 분석 대회가 있다는 사실도 처음 알았음.

[11.18] Evolving Code Completion

- JetBrains 에서 산학연계 프로젝트 같은 느낌으로 진행된 연구.

- Context 내에서 가능한 Code Completion 경우들을 나열하고, LLM 의 output probability 값을 사용하여 다음 token 예측하기. LLM 을 필요한 만큼만 사용하는 것이 인상적임.

[11.18] Faster Runtime Verification during Testing via Feedback-Guided Selective Monitoring

- 기존의 비효율적인 Runtime Verification point 를 찾기 위해 RL 을 사용

[11.18] ZendDiff: Differential Testing of PHP Interpreter

- Implicit Randomness 를 찾기 위해 JIT 과 Normal Mode 실행을 두 번씩 하는데, 이걸 Dual Verification 으로 부름. 간단한 trick 인데 성능을 높일 수 있다는 점이 인상적임.

- PHP + JIT 을 잘 타겟한 것 같다.

[11.18] SATORI: Static Test Oracle Generation for REST APIs

- 중간에 LLM 이 나와서 좀 당황스러웠다. 하지만 정해진 규격이 없는 '설명'을 해석하는 과정이라 LLM 이 제일 괜찮은 방법인 듯하다. OpenAPI Spec 에 type annotation 을 하도록 확장하면 이런 문제를 해결할 수 있지 않을까 싶었음.

- 대형 API Provider 에서 여러 버그를 찾아낸 것이 novelty 로 보임.

[11.18] DALEQ - Explainable Equivalence for Java Bytecode

- Java 에서 다른 버전의 컴파일러를 사용하면 같은 코드인데 결과가 다르게 나옴 → class 파일을 직접 비교하면 같은 코드인지 알 수 없다. → 바이트코드에서 Canonical Form 을 찾아 비교하기.

- Canonical Form 을 찾는 점에서 DEBUN 과 비슷하다.

[11.19] GlassWing: A Tailored Static Analysis Approach for Flutter Android Apps

- Android 대상 분석도구가 Flutter 까지 기능하도록 확장. Flutter 개발을 해왔어서 친숙한 주제였다.

[11.19] LSPFuzz: Hunting Bugs in Language Servers

- LSP 에 적절한 mutation (코드 누락, 오타 등 사람이 유발할 수 있는 각종 오류) + LSP 의 사용자 interaction(hover, click 등)을 적절히 연결한 Domain 에 잘 맞는 fuzzing 연구

[11.19] TEPHRA: Principled Discovery of Fuzzer Limitations

- Fuzzer 현상분석. Experiment family(int family, float family 등)를 정의하고 family 와 프로그램 복잡도를 조절하여 Fuzzer 의 능력을 평가한다.

[11.19] MIMIC: Integrating Diverse Personality Traits for Better Game Testing Using Large Language Model

- 역시 Minecraft (와 다른 게임들)이 연구 대상이라는 것이 신기함.

- 목표 + 인격을 정의해서 게임 내의 다양한 시나리오를 검증한다. 목표만 제공했을 때보다 더 많은 게임 상태를 탐색할 수 있었다.

5. Food


음식은 놀라울 정도로 맛있었다. '이게 국제 학회의 수준이다' 라는 것을 보여주는 것 같았다. 커피는 대량으로 내놓는 커피라는 생각이 들지 않을 정도로 깔끔하고 고소했다. 다과들 중에서는 저 과일절임 패스츄리가 막 달지도 않고 부드러워서 손이 계속 갔다.


점심은 도시락으로, 70%는 한식에 30%는 일식이나 양식이 적절히 합쳐진 구성이었다. 첫날 저녁은 리셉션이었지만 (리셉션/뱅퀫에 대해선 이번 학회에서 처음 알게되었다) 핑거푸드보다 좀 더 식사에 가까운 뷔페식 식사였다. 둘째날은 코스요리가 나왔는데, 뭔가 외국인들이 보면 이런 한식이 있구나 싶은 음식인데 한국 사람들이 봤을때는 이런 한식이 있다고? 라는 반응이 나올 것 같았다. 재료는 한식이 맞긴 한데 평소에 먹어본 음식은 전혀 아니었다.

6. Wrapping up

첫 학회 참여라 모든 과정이 새로웠다는 감상이 큰 것 같다. 연구실의 지원을 받아 참여하는 기회였기 때문에 최대한 시간을 최적화해서 많은 것들을 얻으려고 했지만, 아직 경험과 지식이 충분하지 않아 다른 사람들보다 이해력이 부족했을 것이고, 때문에 놓친 부분도 많이 있지 않았나 싶다. 이후 다른 학회에 참여하게 될 때는 더 다양한 시각으로 보고 느낄 수 있으면 좋을 것 같다.

마지막으로 이번 ASE 에서 SRC 와 Research Paper 발표 및 포스터를 진행한 성민이에게 수고 많았다고 전하고 싶다. 학회 전 몇 주 간 발표자료 준비부터 리허설까지 하는 과정을 보면서 어깨너머로 어떻게 좋은 발표를 하는 건지 배울 수 있었다. 이후에 좋은 기회가 있다면 직접 발표를 하고 싶다는 동기부여도 되었다.