OOPSLA 2019 TRIP REPORT

Athens, Greece

2019.10.20 ~ 2019.10.27

고려대학교 소프트웨어 분석 연구실

이명호

Royal Olympia Hotel(학회 호텔) 7층에서 바라본 Acropolis

1. 시작하기에 앞서

PL 분야 학회에서 최고 수준의 학회 중 하나인 OOPSLA에 2저자로써 있는 "Automatic and Scalable Detection of Logical Errors in Functional Programming Assignments"가 게재되어 연구실 사람들과 같이 학회에 참석하게 되었다. 덕분에 여러 경험을 할 수 있었고, 이 레포트를 통해서 다른 사람들에게 내 경험을 공유하고자 한다.

2. OOPSLA

- Poster

우리 연구실에서는 이번 OOPSLA 포스터에 총 4편에 대한 발표를 진행하였고, 난 또한 2저자로써 순범이와 같이 한 "VeriSmart: A Highly Precise Safety Verification for Ethereum Smart Contracts"와 이번 메인 컨퍼런스에서 발표한 도원이의 ""의 포스터에 대해 발표를 진행하기로 하였었다.

결론적으로는 VeriSmart에 대해서만 발표를 어느정도 진행하였는데, 포스터 세션이 시작한지 시간이 조금은 흐른 뒤라 사람이 많지는 않았지만 그래도 4명 정도가 포스터를 보러 왔었고 그들에게 내용을 설명할 수 있었다.

내용을 다 알고 있고, 사전에 어느정도 준비를 했음에도 불구하고 막상 실제로 설명을 짧고 흥미 있도록 하려고 하다 보니, 버벅이는 등의 내 스스로 부족한 점이 느껴졌었다.

짧은 포스터 발표였지만 질문들을 몇 가지 받았는데, 대부분 주로 invariant를 어떤 식으로 만드는지에 대한 질문이었다. 그리고, 지금도 유독 생각나는 질문은 integer overflow가 아니라 다른 것에 대해서도 검증이 가능하냐는 것과 가능할 경우, 이러한 특정 속성을 검증하는데 있어서 만약 코드 자체가 잘못되었고, 그로 인해 생성된 invariant가 원래라면 안전성 측면에서 만족하지 않아야 할 invariant일 때, 특정 속성에 대해서 잘못 검증하게 되는 경우가 있지 않냐는 질문이었다. 개인적으로 integer overflow를 대상으로 하고, 코드 자체는 의도대로 잘 짜졌지만, 부분적으로 해당 보안 취약점에 문제가 있는 부분들이 있을 거라고 가정한 상태였기 때문에, 이러한 질문은 나에게 있어서 신선하게 다가왔다.

이 외에도 재미있었던 것은 포스터를 보러 온 사람 중 같은 분야의 연구를 진행한 사람이 있었다는 것이다. 마지막에 보러 왔던 사람이었는데 본인도 smart contract에 대해서 분석 연구를 하고 있으며, contract library(https://contract-library.com)이라고 온라인으로 활용할 수 있도록 오픈화 했다고 하였다. 그 쪽 팀은 우리와는 달리 bytecode를 대상으로 진행한다 하였고 포스터를 보러 온 당사자는 분석보다는 bytecode를 decompile하는 파트를 맡고 있다고 하였다. "bytecode를 대상으로 한 이유가 여러 언어로 된 smart contract를 하기 위해서 인가"라는 질문에 맞다고 대답하는 등, 단순히 나만 질문을 받고 답변을 하는게 아니라 서로 여러가지를 물어보며 유익한 시간을 보낼 수가 있었다.

- Main Conference

OOPSLA 메인 학회는 10월23일부터 25일까지 3일에 걸쳐서 진행되었고, 내가 참석하여 발표를 들었던 논문들 중 몇 가지를 날짜순에 따라서 공유하고자 한다.

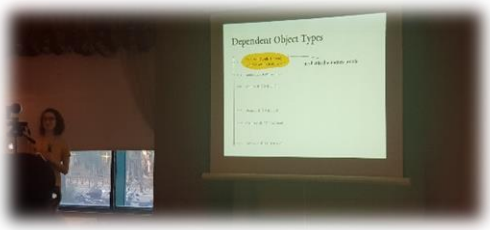
■ Static Analysis with Demand-Driven Value Refinement

이 논문은 JavaScript를 대상으로 하는 정적분석의 정확도를 높이는 것을 목적으로 하였는데, 그러기 위해서 기존의 non-relational 정적 데이터 플로우 분석 기술에 value refinement 기술을 합쳤다. 나는 JavaScript 언어에 대해서는 아직은 큰 관심은 없지만, 정적 분석을 하고 있는 입장에서 분석하고자 하는 대상은 다르지만 어떤 식으로 정확도를 높였는지가 궁금하여 이 발표를 듣게 되었다.

기본적으로 분석을 하면서 value refinement를 필요한 부분들에 적용하는 식으로 진행해가며 precision을 높이는데, precision이 떨어질 것 같다고 판단되는 부분들에 대해서 value refinement를 적용하게 된다. JavaScript를 대상으로 하는 분석인 만큼 해당 언어에 특화되어 있다는 느낌을 받았는데, 그 예로 value refinement를 적용하는 가장 큰 feature중 하나가 dynamic property read/write라는 것이다. 해당 명령문들이 정확히 어떠한 영향을 끼치는지 잘 몰라서 찾아보니 JavaScript 분석을 할 때 해당 명령어들을 처리하는 것이 상당히 중요한데 제대로 분석을 안 하게 될 경우, 다른 property의 값들을 합치게 됨으로써 그 정확도가 상당히 떨어진다고 한다. 이 논문은 이렇게 precision에 영향을 줄 부분들에 대해서 value refinement를 적용함으로써 precision을 높이게 되는데, 이 value refinement는 해당하는 부분들을 좀 더 정확도가 높은 집합 형태로 쪼개 변수와 property 값 간의 관계를 유지하도록 하여 정확도를 높였다.

내가 이해한 바로 쉽게 말하자면 합침으로써 정확도가 낮아지는 부분을 쪼갬으로써 그 정확도를 높이는 기법이었다. 재밌었던 점은 이렇게 실제로 value refinement를 하는 부분들은 그 수가 전체 분석 결과에서 매우 적었는데, 이러한 부분들을 함으로써 성능을 획기적으로 높였다는 점이었다. 결국, 분석에 있어서 모든 것을 자세하게 하는 것도 중요하지만 정말 중요한 부분들에 대해서만 정확하게 해도 전체 결과로 봤을 때 성능이 좋을 수 있다는 것을 다시 한번 느낄 수 있었다.

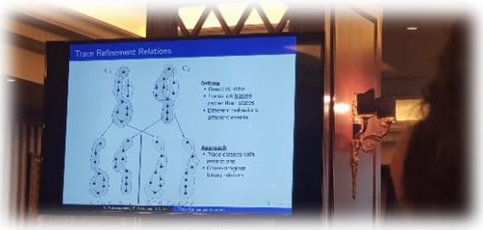
■ A Path to DOT: Formalizing Fully Path-Dependent Types



이 논문에서는 Scala 프로그래밍 언어를 path-dependent-type을 object로써 정형화하는 Dependent Object Types(DOT)가 가지는 한계점을 어떻게 풀었는지에 대해 설명하고 있다. 먼저, path-dependent-type이란 x.a.b.c...T 와 같은 형태의 타입으로 한 예로 x1.a.T와 x2.a.T라는게 있을 때, path가 x1.a와 x2.a로 서로 다르기 때문에 이 둘은 다른 타입이 되는 타입을 의미한다. 그리고 DOT가 가지는 한계점이란 path-dependent type을 처리한다고 하지만, 실제로는 variable-dependent type을 한다는 것인데, 이는 쉽게 말해서 x1.T와 같이 path(x1)의 길이가 1인 것은 가능하지만, x1.a.T와 같이 path(x1.a)의 길이가 2 이상인 것들은 못한다는 것이다. 이 논문에서는 이러한 DOT의 문제점을 해결하고자 기존 DOT를 확장하여 variable이 아닌 임의의 길이를 가지는 full path를 처리할 수 있도록 하였다.

논문과 발표의 내용을 떠나서 개인적으로 OOPSLA에서 들었던 모든 발표 중에 가장 신기하고 눈에 확 띄게 발표를 한 논문이 아닌가 싶다. 처음에 "로미오와 줄리엣"에 대한 이야기로 시작을 하는데, 이게 어떻게 논문 내용과 연결이 되나 싶어서 처음부터 집중하게 되었고 자연스레 로미오와 줄리엣의 가족 관계를 타입으로 표현하며 path-dependent type에서의 타입 에러에 관한 설명을 하는 부분에서 박수를 칠 뻔하였다. 또한 발표 슬라이드 자체도 발표와 엇물려서 눈에 잘 들어오게끔 구성이 되어 있었고, 하나하나 기존의 방식인 DOT가 어떤 식의 한계가 있는지 설명을 하고 그에 맞춰서 pDOT에서는 어떻게 처리를 하였는지 흐름에 따라 설명을 상당히 잘 하였다고 느꼈다. 발표 자체가 아직도 생각이 난다는 점에서 "발표를 이런 식으로 할 수 있구나"라는 생각을 하게끔 하는 발표였다.

■ Specification and Inference of Trace Refinement Relations



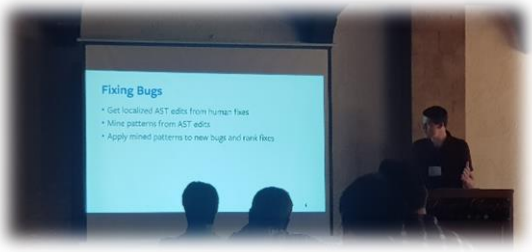
이 논문은 소프트웨어 공학 느낌이 나는 논문이었다. 이 논문의 대상은 버전이 서로 다른 같은 소프트웨어들로써 소프트웨어가 시간이 지남에 따라 최적화나 취약점 패치, 또는 refactoring과 같은 이유로써 업그레이드되며, 그렇게 해서 생겨나게 되는 예전 버전과 새로운 버전 간의 비슷한 점과 차이점을 잘 비교 분석해야 한다는 것이다.

서로 다른 버전의 소프트웨어들을 비교하는데 있어서 단순히 syntactic적인 비교만 하는 것이 아니라 좀 더 심층적인 부분을 해야 한다고 하는데, 그 이유로써 취약점 패치와 같이 지극히 작은 부분의 변화라도 실제 소프트웨어의 동작에 대해서는 큰 영향을 끼치거나 refactoring과 같이 많은 부분이 syntactic하게 바뀌더라도 실제 그 semantic은 변화가 없는 경우들이 있기 때문이다.

두 소프트웨어를 비교하기 위해서 이 논문에서는 두 소프트웨어 간의 state 관계 비교가 아닌 trace 관계 비교를 하였는데, 개인적으로 이해했을 때 trace는 단순히 각 소프트웨어의 execution path들의 집합으로 느껴졌다. 그리고 이 trace들을 KAT(Kleene Algebra with Tests) 표현식을 이용하여 표현을 하는 부분이 재밌게 다가왔다. 비교하고자 하는 소프트웨어들의 동작을 KAT 표현식으로써 표현하고, 이 두 표현식 간의 관계를 추정하는 방식 또한 신선히 다가왔다.

개인적으로는 두 소프트웨어를 비교하기 위해서 KAT(Kleene Algebra with Tests)를 활용하여 소프트웨어의 동작 구성을 나타내고, 그 intersection 관계를 이용한 것이 재밌게 느껴졌으나 구버전이나 신버전 상관없이 분석 자체를 잘한다고 하면 된다고 생각하기 때문에 서로 다른 버전의 소프트웨어의 비교를 해야 한다는 motivation이 크게 와 닿지는 않았다.

■ **GetaFix: Learning to Fix Bugs Automatically**



우리 연구실에서 비슷한 주제를 하는 팀이 있는 만큼, 이 발표는 재미있게 볼 수 있었다. 특히, 이 발표는 마지막 날에 있었는데, 그 전날 포스터 세션에서도 발표를 했었고, 그 때 나도 참석해서 몇 가지를 물어봤었기 때문에 더 재미있게 들을 수 있었다.

이 논문은 패치를 생성할 때 그냥 버그를 고치는 것뿐만 아니라, 사람이 쓸 법한 패치를 하는 것을 목적으로 하고, 그 수단으로써 기계 학습을 이용하였다. 비슷한 패턴들의 버그에 대해서 사람들이 패치를 하고, 이러한 버그 패턴과 그에 맞는 패치들을 학습하여 사람들에게 자동적으로 추천해주는 방식으로 크게 세가지 과정을 통해서 진행된다. 먼저 버그와 그에 대한 패치들을 AST 형식으로 변환을 하고, 그 AST 상에서 패턴들을 학습한 후, 마지막 단계에서 특정한 버그를 입력으로 받아 학습된 결과에 맞춰 top-k 개의 패치 후보군들을 추천해주는 방식이다. 사람들이 직접 쓴 패치를 학습하다 보니 사람이 흔히 쓸 수 있는 패턴의 패치를 제공해준다는 장점이 있었다. 보통 패치를 하는데 있어서 합성 기술을 쓰게 되면 버그를 고치긴 하지만 종종 이상한 형태의 코드를 생성하는 것을 많이 보게 되기 때문에 이 부분이 마음에 들었다.

포스터 때 질문을 하였던 아쉬운 점은, 아무래도 기존에 이미 사람들이 썼던 특정한 형태들의 버그와 패치를 학습하다 보니, 완전히 새로운 형태에 대해서는 패치를 할 수 없지 않을까 하는 것이었고, 이 질문에 대해서 답하길 정확히 실험을 해보지는 않았지만 논문의 접근방식상 못할 것이라는 답변을 받았다. 아는 것들에 대해서 잘하는 것도 중요하지만 결국은 새로운 형태의 버그들을 처리하는 것도 중요하다고 생각하는 입장에서 이러한 제한점은 아쉽게 느껴졌다.

3. ATHENS, GREECE

- **관광지**


Acropolis: 보수 공사를 하고 있기 때문에 철근이 많이 보인다.


The Temple of Hephaestus and the Roman Agora: 온전히 남아있는 건물은 몇 없었다.

OOPSLA 메인 컨퍼런스가 10월 23일부터 10월 25일까지였기 때문에, 그 전에 시간적 여유가 있어 아테네의 여러 관광지들을 둘러보게 되었다. 관광지는 아크로폴리스를 포함해서 크게 7군데가 있었고, 5일 동안 그 7군데를 마음껏 들어갈 수 있는 티켓이 30유로라 그 티켓을 사고 관광지를 둘러보았다. 유적지들은 대체로 비슷한 느낌으로 터만 남아있는 등, 세월의 흔적이 느껴지지만 딱 그 정도다 싶은 느낌이 있어서 뭔가 아쉬운 감이 있었다.

- **날씨**

Acropolis에서 본 아테네 도시 전경. 그리스에 있는 동안의 날씨는 상당히 맑았다.

10월 중후기의 아테네의 날씨는 우리나라의 늦여름, 초가을과 비슷한 느낌으로 아침 저녁에는 쌀쌀하고 점심에는 더웠으며, 습하지가 않아서 낮에 햇볕은 따갑더라도 그늘에 들어가면 서늘하였다. 우리가 머물렀던 기간에는 날씨 자체도 사진과 같이 상당히 맑아서 돌아다니는데 큰 무리가 없었다.

- **음식**

Acropolis 근처 음식점에서 Greek Traditional Brunch란 이름으로 팔던 것. 체리가 들어간 요거트가 특히 맛있었다.

좌상단의 곁들어진 야채 볶음은 상당히 짰고, 생선 자체는 담백한 맛이었다

Orzo: 오르소 버섯 파스타로 버섯 향이 강하게 나서 개인적으로 먹었던 음식 중 제일 맛있었다.

그리스 음식들은 대체로 짠 편이었는데, 짠 음식을 잘 먹는 입장에서 대체로 입맛에 잘 맞았다. 대부분의 음식들에는 치즈가 들어있었고, 그리스 치즈는 잘 부서지는 두부 같은 느낌으로 염분이 상당하였다. 그래서 그 자체만 먹기에는 부담스럽지만 샐러드와 같이 다른 음식에 곁들여 먹기에는 괜찮았다.

4. 마치며

OOPSLA는 처음 와봤었는데 내가 들었었던 발표와 포스터들 모두 상당히 수준이 높았었던 것 같다. 수준도 수준이지만 내가 잘 모르는 분야들의 경우들도 있어서 솔직히 마냥 발표만 들었을 때는 그 내용을 전부 이해하는 것이 쉽지는 않았다. 한 가지에 몰두하는 것도 좋지만, 그 외 다른 분야들에 대해서 공부하는 것을 쉬면 안되겠다는 생각이 들었다.

마지막으로, 포스터 및 학회 발표를 준비하느라 고생한 순범이, 준희, 도원이 그리고 민석이 형한테 고생했다는 말과 함께 OOPSLA라는 좋은 학회에 와서 여러 경험을 하도록 해 주신 오학주 교수님께 감사의 말씀을 드리고 싶다.