FSE 2022

싱가포르

고려대학교 홍성준

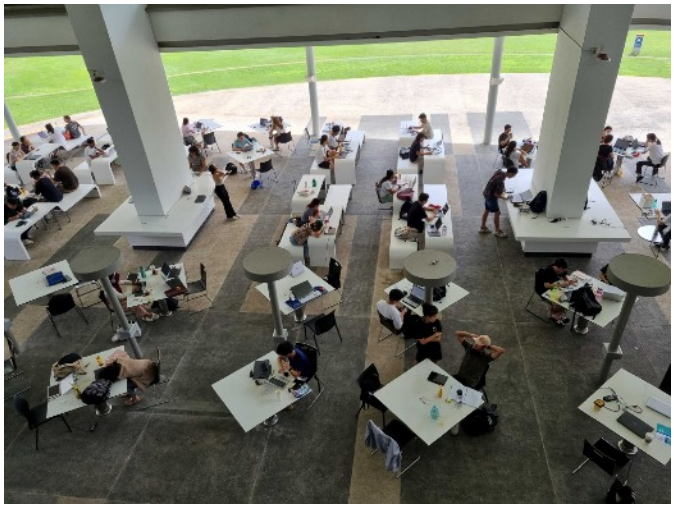
1. 소개

소프트웨어공학 분야에서 명망있는 학회인 FSE'22에 다녀왔다. 나로서는 2018년 FSE를 마지막으로 만 4년만에 참가하는 국제 학회였다. 게다가 연구실에서 여러명이 같이 가는건 또 처음이라 출국 전부터 설렌 마음을 안고 비행기를 탔다. 싱가포르라는 매력적인 도시 + 코로나 유행이 조금은 잠잠해진 덕분인지, 이번 FSE는 등록이 조기 마감될 정도로 성황리에 개최되었다. 졸업 준비 등 여러 사정으로 학회에 참가하지 못한 연구실 구성원들에게 경험을 공유하고자 이 글을 쓴다.

2. 학회

NUS

이번 FSE는 싱가포르 국립 대학에서 열렸다. 해외 대학은 처음가봐서 캠퍼스 구경을 꽤 열심히 했던 것 같다. 학교의 첫 인상은 돈이 많아보였고, 식물이 많아 되게 푸릇푸릇했다. 학교 곳곳에 분수대, 연못, 나무들이 무성했다. 캠퍼스 설명은 사진으로 대신하겠다.



NUS의 경쟁력!! (곳곳에 자고있는 학생도 보인다)

캠퍼스 내 한국 식당. 맞은편엔 술집(!!)도 있다.

학회 메인 빌딩. 우리 학교의 하나스퀘어같은 용도의 건물이다. 나중에 알아보니 Stephen Riady는 부동산업쪽의 재벌이신 듯 했다.

문어가 귀여워서 찍음. 무인스토어인데 회원 등록을 하고 태그해서 들어가야된다. 인심은 우리학교가 이겼다.

연구실 소개 사진

키노트

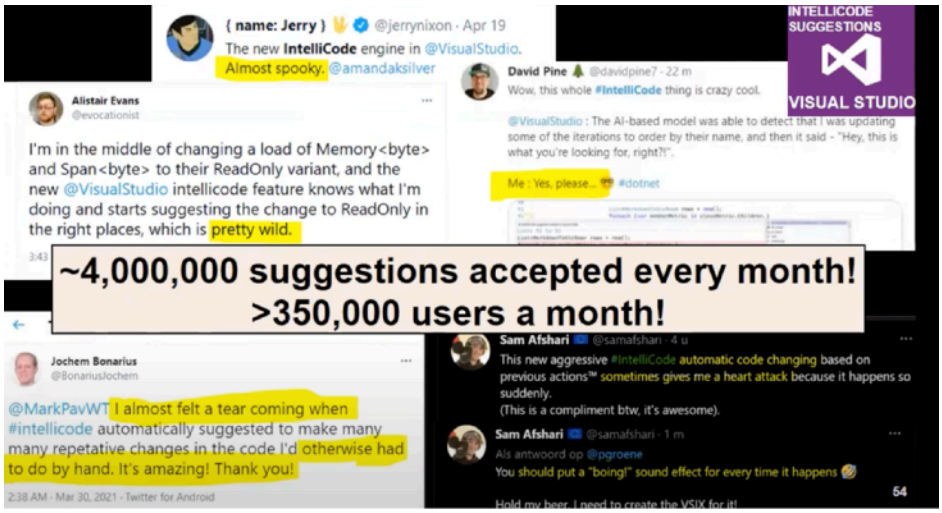
이번 키노트는 MS 리서치의 Sumit Gulwani가 발표했다. 발표 내용은 우리 랩에서도 친숙한 “AI의 도움을 받는 프로그래밍”이다. 한마디로 프로그램 합성 기술에 관한 발표였다. AI의 위세는 대단해서, 몇년째 SE 학회 키노트에는 항상 등정하는 것 같다. 그래도 학회 키노트를 보통 첫날 억지로 듣는 경우가 많았는데, 처음으로 관심가는 주제의 키노트여서 꽤 재밌게 들었다.

키노트에서는 유저의 다양한 의도에 따른 합성 기술들이 소개되었다:

- 입/출력 예시
- 자연어
- 코드 맥락
- Repair (컴파일 에러, 학생 오류 코드 수정 등)

연구가 여러개라 지루할 틈 없이 집중해서 듣게되었다. 발표를 들어보니 이 사람은 프로그램 합성에 진심이라는 생각이 저절로 들 수 밖에 없었다. 가끔 주제를 떠올릴 때 생각으로만 끝내는 것들을 이 사람은 직접 논문으로 완성시킨다는 느낌을 받았다.

Sumit Gulwani는 논문을 찍어내다시피 많이 쓰는걸로 익히 알고있어서, 연구 (논문 작성)만 할 것 같지만 의외로(?) 기술의 실용적인 측면도 고민을 많이 하는 것 같았다. 대단하면서 부러웠던 건



본인이 연구한 기술들을 액셀, Visual Studio등 자사 어플리케이션들에 적용해서 수많은 유저들에게 실제로 영향을 주었다는 점. 나도 일반 유저들은 아니더라도, 많은 개발자들이 실제로 사용하는 기술을 만들고싶다.

Repair 논문들

아무리 새롭고 재밌는 주제인들 내 관심 분야인 repair만은 못한듯하다. 학회를 여러번 참석해보니, 이제는 적어도 내 분야에서는 익숙한 얼굴들이 많이 보였다.

이번 학회에서는 두 발표가 특히 기억에 남는데, 하나는 내 주제와 기술적으로 직접적인 연관이 있는 연구이고, 다른 하나는 요즘 유행하는 딥러닝 기반 Repair였다. 두 논문에 대한 소개를 해보겠다.

Trident: Controlling Side Effects in Automated Program Repair

나랑 가장 비슷한 스타일의 연구를 한다고 생각되는 Sergey Mechtaev의 연구다. ICSE'18부터 학회를 갈 때마다 만나고 항상 발표를 챙겨듣는 편이다. 이 사람은 제약식 (constraints) 기반 repair로 생각할 수 있는 주제는 정말 다 하는것 같다. 원래는 TSE'21에 발표된 연구인데 Journal First로 초빙된 논문이다. 저널을 상대적으로 잘 안보다보니까 놓쳤던 논문이었다. 이 연구는 제약식 기반 repair에서 side-effect가 존재하는 대입문 (assignment), function call 패치를 합성하려면 어떻게 해야 하는지를 고민했다. 예를 들어 아래 예시를 보자:

```
int z;
□_l = □_r // PATCH
if (x < y)
  z = x;
return z;
```

```
int z;
switch (PATCH_ID)
        case 0: x = □_r
        case 1: y = □_r
        case 2: z = □_r
...
```

대입문을 합성하는 것은 얼핏 생각해봐도 까다로운데, 왼쪽 코드와 같이 대입문 □_l = □_r를 삽입한다고 할 때 □_l에 가능한 l-value에 따라 실행 경로가 구분되어야 하기 때문이다. 변수 x, y, z만 생각해봐도, 위 코드에서 왼쪽과 같은 스위치문을 분석하게 되는 식이다. 이러한 방식은 당연히 경로 폭발로 인한 scalability 문제를 야기한다. 이 연구는 정확히 이러한 문제를 해결하기 위한 효율적인 인코딩 방식을 제시한다. Trident는 위 패치에 대해 다음과 같은 조건식을 만든다:

$$\phi = (s_x \rightarrow x' = \alpha_x \wedge \neg s_x \rightarrow x' = x) \wedge (s_y \rightarrow y' = \alpha_y \wedge \neg s_y \rightarrow y' = y) \wedge \cdots$$

s_x는 변수 x가 정의되는지 여부를 의미하고, α_x는 그 때 x가 갖는 symbolic value를 의미한다. 제약식 ϕ를 통해 기호 실행 레벨에서 경로를 분기하지 않고 여러 대입문의 효과를 압축적으로 표현하게 된다.

지금 하고 있는 연구에서도 상당히 비슷한 문제를 겪고있기 때문에 좀 더 건질게 있나 싶어서 논문도 들여다봤는데 기술적으로는 저게 전부라서 좀 아쉬웠다. 지금 연구에서는 이 문제를 state merging 휴리스틱을 디자인해서 해결하는데, 여기서는 대신에 SMT 솔버의 파워를 이용하는 아이디어라고 이해했다. 해결 방식은 간단하지만 Angelix 스타일의 repair를 하지 않는다면 연구로 완성시키는 건 어렵겠다는 생각이 들었다. 지구상에 비슷한 문제를 고민하는 사람이 있다는 것이 반가웠던 연구이다.

Less Training, More Repairing Please:
Revisiting Automated Program Repair via Zero-Shot Learning

제목에 낚였던 발표. 요즘 나오는 최신 repair 연구들을 조금 후려쳐서 말하면 "그냥 딥러닝 한번 써볼까?" 하는 생각이 많이 든다. 나는 이 연구가 그런 세태를 꼬집어주는 연구인 줄 알았다.

실제로 그런 것은 아니고, 요즘 유행하는 pre-trained model인 CodeBert를 APR에 적용해본 연구로 정리될 수 있을 법한 연구다. 이 연구가 지적하는 문제는 다소 모호하다. 기존 딥러닝 기반 APR은 모델을 학습시키기 위해 (오류 코드, 패치 코드)의 쌍을 필요로하는데, 이러한 데이터는 대량으로 모으기도 어렵고, 좋은 품질의 데이터를 구하기도 어렵다는 것이다. 이렇게 Fix를 학습하는 대신, 그냥 수많은 코드 데이터로부터 학습된 CodeBert 모델을 써보자는게 주요 골자다. 패치 생성 과정은 상식적이다. 기존에 템플릿 기반 APR에서 잘 알려진 패턴들로 마스킹을 하고, CodeBert를 사용해 마스크에 들어갈 코드를 추론하는 식으로 패치를 생성한다.

성능은 SOTA보다 향상되었으나 비약적인 개선은 아니었고, 무엇보다 어프로치 자체의 soundness도 의심스러웠다. 첫째로, pre-trained model에는 분명 buggy program도 섞여있을텐데, 이러한 데이터를 정제없이 학습한 모델을 repair에 사용하는게 이상했다. 둘째로, 모델 학습 데이터에 벤치마크로 사용된 정답 패치 데이터가 포함되어있는데, 이 비율이 정답 패치를 생성한 버그의 약 15%에 이른다. 학습 데이터의 품질을 문제로 지적했는데, 정작 이러한 부분들에 대한 해명은 부실하다.

분명 프로그램 Repair/합성 세션인데 우리 연구실에 원석이가 발표한 PyTER를 제외하면 나머지 네 편의 발표가 모두 딥러닝과 관련된 발표였다. 딥러닝을 별로 안좋아하지만, 거스를 수 없는 시대의 흐름인지, 내가 모르는 뭔가가 있겠구나 하는 생각도 들었다. 앞으로도 특히 더 관심을 가져야 하는 연구 동향인 듯 하다.

뱅퀫

뱅퀫은 Mandai Night Safari에서 열렸다. 큰 동물원 내부의 식당까지 걸어가는 길에도 볼거리가 많았다. 나이트 사파리는 밥을 다 먹고 출발했다. 전동 열차를 타고 이동하면서 동물들을 보는데, 열차가 양 옆으로 뚫려있어서 처음엔 좀 무서웠다. 나는 사파리를 안가봐서 동물들이 내 바로 옆까지 오는건 줄 알았기 때문이다. 그건 아니고 멀리서 서식지를 구경하는 방식이었다.

밥먹고 기차를 타기까지 한 시간 넘게 기다리는 바람에 다들 원성이 자자했다. 기다린 시간에 비하면 볼거리가 크게 많지 않긴 했다. 나는 뭘 타고 구경하는걸 워낙 좋아해서, 개인적으로는 되게 만족스러웠다.

4. 싱가포르

편리한 교통

싱가포르 교통은 매우 편리했다. 해외 나가면 버스타는 것도 일인데, 싱가포르에서는 그냥 한국처럼 다녔다. 우선 신용카드를 교통카드로 쓸 수 있어서 삼성페이로 대중교통을 타고 다녔고, 환승 시스템도 상당히 잘되어있었다.

학회 기간에 애용했던 2층버스. 싱가포르는 중앙선이 오른쪽에 있어서 재밌었다.

횡단보도가 경계만 그려져있다. 페인트를 아낄 수 있겠다 생각했다.

음식은… 라면이 그리웠다.

특색있는 음식은 딱히 없었고, 음식들이 향신료 냄새가 많이 강해서 내 입맛에는 안맞았다. 처음으로 외국에서 라면이 생각났다. 칠리 크랩이 유명하대서 먹어봤는데 그 돈 받고 팔아도 되나 싶을 정도였다. SNS의 폐해라고 생각했다. 제일 괜찮았던 건 길거리 노상에서 먹었던 4천원짜리 오리덮밥.

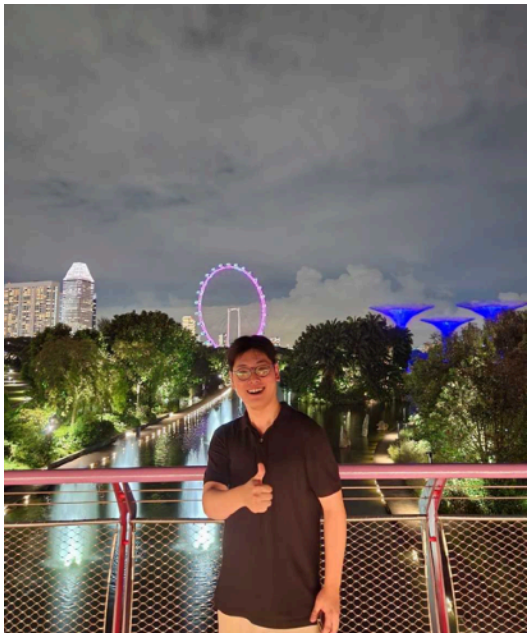
아름다운 인공물들

싱가포르는 빌딩 야경, 분수쇼 등 빛으로 만들어논 볼거리들이 많았던 것 같다.

돌아오는 날 창이 공항 쥬얼에 들려 본 실내 폭포. 어떻게 건물 안에 천장에서 쏟아지는 폭포를 만들 생각을 했을까? 장관이었다.

분수에 진심인 나라.

유명한 금융사들로 빽빽했던 빌딩 숲. 야근하는 사람이 정말 많이 보인다.



유명한 싱가폴 버섯 인증!

5. 마치며

반복적인 대학원 생활로 일상이 루즈해지던 찰나에 멘탈을 리프레쉬할 수 있는 좋은 기회였던 것 같다. 학회가서 내가 속한 커뮤니티의 사람들과 얼굴을 맞대는 것만으로도 뭔가 자극이 있는 것 같다. 또, 후배들과 같이가서 좋은 추억을 쌓고 올 수 있어 좋았다. 숙소, 일정 준비 등으로 고생한 후배들에게 고맙다. 마지막으로, 좋은 경험 할 수 있도록 지원해주신 오학주 교수님께 감사드립니다.