



SPLASH 2024

California, United States of America
2024.10.20. - 2024.10.25.

JOONGHOON LEE

Software Analysis Laboratory
Korea University

1. 들어가며

미국 캘리포니아에서 열린 OOPSLA에서 논문을 발표하였다. 첫 학회 참석이라 걱정도 많았지만 다행히 잘 마무리하고 돌아왔다. 이번 SPLASH 2024의 뒤 3일 동안 OOPSLA가 진행되었고, SAS도 등록하여 앞 3일간 다른 학회들도 함께 참석하였다. 학회 기간 얻은 경험을 본 글을 통해 공유하고자 한다.

2. 논문 발표

여러 사람이 내 말을 경청한다는 것은 정말 설레고 영광스러운 일이다. 아쉬운 부분들도 있지만 만족스러운 발표였다.

준비 과정. 원래 계획은 출국 전에 발표 준비를 완전히 마치고 발표 이를 전쯤부터 간단하게 연습 하려 했다. 몇 주 전부터 친구 선배님 도움을 받아 발표 자료를 만들었고, 과거 연구실 선배님들 발표 자료와 유튜브에서 학회 발표 영상도 많이 찾아보았다. 어느 정도 키노트 내용이 갖춰져 출국 며칠 전 교수님께 짧게 피드백 부탁드렸는데, 너무 자세한 내용은 빼고 좀 더 high-level에서 발표해도 괜찮을 것이라 조언해주셨다. 발표 시간이 짧아 내용을 전부 전달하기는 어려울 것이며, 학회장이 여러 개로 나뉘어 있어 관심 있는 사람들 위주로 올 것이라는 이유에서였다. 결국 발표 자료를 많이 줄이기로 마음먹었고, 시간이 부족하겠다고 판단해 가는 비행기에서나 밤에 현지 숙소에서도 발표 연습을 했다. SPLASH 첫날 학회장에 도착하자마자 한 일은 내 발표가 진행되는 행사장을 미리 가본 것이었다. 발표장 내 배치나 구조 등을 미리 확인해두면 이미지 트레이닝에 많이 도움 되기 때문이다. 다만 스크린도 크고 의자도 정말 많아 생각보다 규모가 많이 컼고, 발표 연습 많이 해야 겠다고 다짐하였다. 그래서인지 더 긴장되었고 출장 기간 내내 시차 적응을 못 하고 새벽에 깨어 있었던 것 같다.



Figure 1: 발표 모습

발표. 발표 현장에는 생각보다 사람이 많이 없었다. 학회 마지막 날 마지막 발표였고, 다른 양자 연구들은 하나의 세션에 모여 있었는데 내 발표는 테스팅 세션에 따로 배정되어 더 그런 것 같았다. 그래도 참석하신 분들은 이미 얼굴이 눈에 익은 양자 연구하시는 교수님들과 대학원생 분들이었고, 준비한 내용을 조금이라도 더 아는 청중들이 참석했다는 생각에 자신 있게 발표를 진행할 수 있었다. 개인적인 발표 팁들과 이번에 깨달은 것들을 공유하고자 한다. 첫 발표를 앞둔 누군가에게 도움이 되었으면 좋겠다.

1. 발표 시간은 짧다. 15분 발표 5분 질의응답 시간으로 구성되었다. 만약 학교에서 팀 프로젝트 발표였다면 다들 비슷하게 아는 내용 들이라 본론과 디테일에 집중해서 발표해도 충분히 전달되지만, 학회는 성격이 조금 달랐다. 워낙 다양한 분야 들을 연구하는 사람 들이 모이다 보니 아주 high-level에서 구성하는 게 좋아 보인다. 이건 개인적인 발표 팁인데, 시간이 정해진 발표에서는 레이싱 게임 타임 어택하듯 체크포인트를 만들어두면 아주 유용하다. 나는 이번 발표에서 *Born Rule*은 발표 시작 후 4분, *Fixed-Point*는 7분, *Evaluation*은 12분에 도달하도록 준비했다. 막연히 발표하기보다는 미리 시간을 설계해두면 연습에서나 실전에서나

안정적으로 말할 수 있다.

2. **발표 초반이 중요하다.** 여러 발표를 직접 들으며 느꼈는데, 발표 앞부분 문제 설명이 충분히 잘 전달되고 흥미롭게 들려야 발표에 집중할 수 있는 것 같다. 15분 발표 중 거의 앞 5분 안에 발표 성패가 결정되는 듯하다. 어차피 그 분야의 전문가가 아니면 자세한 내용은 따라가기 벅차다. 여러 발표를 들으면서 교수님께서 주셨던 조언이 옳았음을 백 번 느꼈다. 만약 또 발표하게 된다면 발표 초반 구성을 더 신경 쓰고 문제 설명까지 시간을 더 할애할 것이다.
3. **질문 내용은 대체로 비슷하다.** 사실 가장 우려했던 부분이 질의응답 시간이었다. 내가 못 알아듣거나 대답 못 하면 어쩌지 하는 걱정이 있었는데, 사람 생각은 거기서 거기라 하지 않는가. 다른 발표 들에서도 내가 궁금한 내용은 대부분 다른 사람이 먼저 질문하기도 했고, 내 발표에서 들어온 질문들도 어느 정도는 예상 범위 안에 있었다. 특히, 양자 연구하시는 교수님께서 주셨던 질문 중 하나는 노이즈가 존재하는 환경에서 이 방법을 적용할 수 있는가였는데, 논문 리뷰 과정에서 받았던 질문과 거의 유사해 매끄럽고 자신 있게 답변할 수 있었다. 다음 발표를 하게 된다면 훨씬 걱정 덜고 답변 준비할 수 있을 것 같다.

3. 발표 참석

다양한 분야의 발표가 동시에 진행되다 보니 계속 옮겨다니며 발표를 들었다. 인상적이었던 연구 세 가지 정도를 소개하고자 한다.

Quantum Control Machine: The Limits of Control Flow in Quantum Programming. 이 연구에서는 양자 프로그램의 control flow를 설명하기 위해 *reverse jump* 도입을 제안한다. 고전 컴퓨터에서는 conditional jump로 현재 레지스터에 담긴 값에 따라 다음으로 실행될 위치를 program counter에 설정할 수 있다. 하지만 양자 프로그램을 실행할 때는 각 레지스터가 중첩 상태에 놓이게 된다. 즉, conditional jump를 사용하면 분기마다 확률적으로 다음 program counter 값이 달라진다. 하지만

이 명령어를 양자 프로그램에서는 그대로 사용할 수 없다. 양자 게이트 기반 양자 컴퓨팅 과정에서 모든 연산자는 유니터리 행렬로 표현되는데, 이는 norm-preserving 연산으로 올바른 양자 상태는 그 norm 값이 1이 되어야 한다. 그리고 고전적인 conditional jump로 중첩 상태에 대한 state transition 을 따라가면 불가능한 norm 값이 발생하게 된다. 저자는 이 문제를 해결하기 위해 reverse jump 도입을 제안한다. 가장 관심 있었던 연구였고, 몰입력 있게 발표도 잘 구상하여 재미있게 재미있게 들었다. 사실 *QRAM(Quantum Random-Access Memory)*의 도입과 관련해서는 여러 관점이 있을 수 있지만, 어찌 되었든 기존에 존재하는 명령어들로는 한계가 존재함을 보인 인상적인 연구였다.

Quarl: A Learning-Based Quantum Circuit Optimizer. 양자 회로 최적화에 관한 연구다. 예전 PLDI에 비슷한 목표의 연구로 Quartz와 Queso가 있었고, *rule-based* 회로 변환을 통해 동일한 연산을 수행하지만, 더 적은 개수의 CNOT 게이트를 사용하는 회로를 찾는다. 이에 반해 Quarл은 학습 기반의 회로 최적화 방법을 제시한다. 우선 *rule-based* 회로 변환을 수행하다 보면 자원의 한계로 어쩔 수 없이 변환 깊이에 한계가 존재한다. 즉, 가까운 몇 번의 변환 중 가장 높은 점수의 회로를 선택하게 된다. 하지만 이 연구에서는 그 방법이 최선이 아님을 강조한다. 단기적으로는 오히려 CNOT 게이트 개수가 많아지는 비효율적인 변환이지만, 최종적으로 더 적은 CNOT 게이트를 사용하는 회로에 도달할 수도 있다는 것이다. 발표에서는 실행 시간에 관한 언급이 없어 따로 질문했는데, 같은 시간 동안 타 연구 대비 가장 좋은 결과를 보였다고 했다. 합성 중간 단계에서 greedy하게 변환하지 않는 게 더 optimal 한 최종 결과에 도달할 수 있지 않을까 싶었는데, 실제 결과와 학습 기반의 적절한 접근 방법을 통해 해결하여 흥미롭게 들은 발표였다.

Enhancing Static Analysis for Practical Bug Detection: An LLM-Integrated Approach. LLM 을 이용해 정적 분석을 수행한다는 어마어마한 제목에 이끌려 발표에 참석하였다. 이 연구에서는 *use-before-initialization* 버그를 찾는 데 집중한다. 이미 존재하는 리눅스 커널 UBI 버그를 찾는 기호 실행 정적 분석기 *UBITect*를 실제로 실행하면 메모리 및 시간 초과로 인해 약 60% 정도만 커버할 수 있다. 즉, 나머지 40% 안에 false positive 버그들이 존재할 수도 있고, 이 경우 들을 다

루기 위해 LLM을 이용한다. 예를 들어, `sscanf` 함수의 리턴 값을 비교하는 코드가 있다면, 적어도 그 함수의 인자들은 초기화된 상태여야 한다. 이 연구에서는 이러한 *post-constraint*를 이용해 취약 지점을 찾고자 한다. *UBITect*에서 *undecidable*로 결정된 경우 LLM을 단계적으로 실행하며 최대한 *decided* 결과를 도출한다. 만약 LLM을 실행한 결과 정확한 판단을 위해 함수 정의가 필요한 경우, 해당 정보를 제공하여 다시 LLM을 실행하는 등 몇 단계의 상호작용을 통해 결론을 낸다는 점은 인상적이었다. 이 연구에서는 이 과정을 *Progressive Prompt*라 말한다. 총 10개 이상의 버그를 찾는 등 좋은 결과도 나왔고 접근 방법도 재밌는 연구였다.

4. 맷으며

첫 학회 참석인데다 발표도 해야 한다는 생각에 걱정을 정말 많이 했는데, 열심히 연습한 덕분인지 만족스럽게 마무리할 수 있었다. 전 세계에서 온 연구자들과 열정적으로 대화하던 순간은 정말 인상 깊게 기억에 남았고, 스스로도 앞으로 연구하는데 많은 동기부여가 되었다고 생각한다. 특히 인터넷에서만 보던 연구자들을 직접 만나니 뭔가 연예인 본 듯한 신기한 무언가가 있었다. 이번에는 2저자 논문 발표였는데, 다음에는 1저자로 논문 발표하리라 다짐해본다. 연구 지도해주신 오학주 교수님, 함께 연구하는 강찬구 선배님, 그리고 발표 준비부터 학회 출장까지 도움 주신 연구실을 포함한 모든 분들께 진심으로 감사드립니다.

Appendix A 사진들



(a) 산타모니카 해변



(b) LA 야경



(c) 입국장



(d) 학회장 주변 풍경