PLRG

PLDI 2024

Copenhagen, Denmark

김준겸

2024. 06

1 Introduction

PLDI는 명실상부한 프로그래밍 언어 분야의 Top-tier 학회 중 하나이다. 박지혁 교수님이 전민석 박사님이 쓰신 PL4XGL 논문에 도움을 주셔서 저자로 들어가셨기 때문에, 운 좋게 꼽사리를 껴서 PLDI를 구경할 수 있게 되었다. 안타깝게도 우리 연구실 사람들은 SE 쪽에 관심이 더 많고 PL에 더 관심이 많은 사람은 내가 유일하기 때문에 PL 학회, 그 중에서도 POPL과 더불어 제일로 생각하는 PLDI에 참석하게 된 것은 너무 행복한 일이었다.

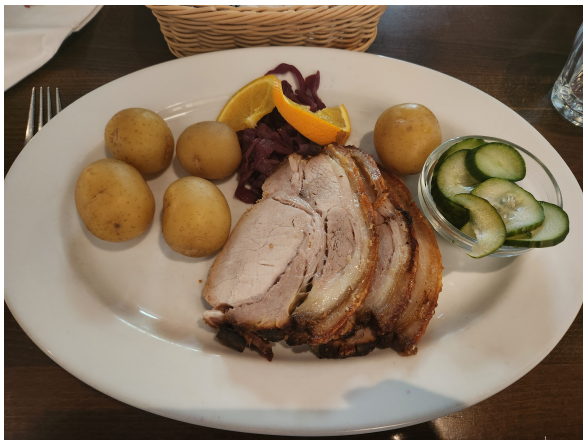
PLDI 2024는 내가 처음으로 참여하는 학회였다. 그래서 기대도 많이 했고, 실제로도 매우 재미있었다. PLDI는 프로그래밍 언어 관련 학회라서 SE 학회보다 훨씬 이론적인 내용이 많았지만, POPL이나 ICFP만큼 지나치게 이론적이지 않아서 현실과의 적절한 균형을 맞춘 느낌이었다. 그래서 다양한 분야의 연구를 접할 수 있었고, 학회 내내 “이런 연구도 있구나” 하는 새로운 깨달음을 많이 얻었다.

가장 인상 깊었던 점은 사람들이 적극적으로 말을 걸어왔다는 것이다. 간식을 먹고 있는데 미국 대학교 조교수님이 와서 테이블을 같이 써도 되냐고 묻더니 바로 무슨 연구를 하고 있는지 물어보는 식이었다. 처음에는 영어가 서툴러 당황스러웠지만, 점점 적응이 되면서 나도 먼저 말을 걸고 재미있게 대화를 나눌 수 있었다. 무엇보다 연구실에서는 정적 분석이나 자료구조에 대해 얘기하면 아무도 관심을 안 가져주는데, 학회에서는 관련 배경지식이 풍부한 사람들을 만나 활발하게 토론할 수 있었다는 점이 정말 좋았다. 아쉬운 점은 내가 아직 연구를 많이 못 해서 할 이야기가 많지 않았다는 것이다. 이번 학회에서 발표를 하냐는 질문을 많이 받았는데, 그때마다 “아니요”라고 대답하는 것이 매우 속상했다.

학회를 와서 제일 당황스러웠던 것은 발표를 듣는 것이 체력을 생각보다 많이 소모한다는 것이었다. 그냥 앉아만 있어도 되고 시차 적응도 1년 전에 끝내놓은 상황이라 편하게 보면 될 줄 알았는데, 항상 오후 3시쯤 되면 기진맥진해서 소파에 누워있기 일쑤였다. 진한 커피를 싫어하는데 어쩔 수 없이 마시느라 고생했다. 다음 번에 학회를 하면 한국에서 하면 좋겠다. 적어도 비행기 탈 체력은 아낄 수 있으니까.

가장 중요한 학회 밥은... 솔직히 실망했다. 호텔 밥이라고 해서 꽤 기대했는데 첫날 점심은 정말 극단적으로 말하면 ‘맛대가리 없었다’. 퍽퍽하고 느끼하고 짜고(본인은 라면에 물 조금 넣은 다음 소금을 뿌려먹는 사람인데도!) 전반적으로 끔찍했다. 갈수록 나아지기는 해서 마지막 날은 꽤나 맛있긴 했다만 평균적으로는 여전히 구렸다. 간식도 너무 달거나 너무 건강해서 건강보험이 3개쯤 있어야 먹을 수 있는 간식이 나오거나 시금치 오이 샐러리 주스 같은 것이 나왔다. 이런 걸 먹으면서 생존할 수 있다니 역시 북유럽인들은 대단하다.

(a) 청어 요리. 비렸다.



(b) 겉면이 바삭한 수육 비스무리한 것.

Figure 1.1: 한국인의 밤에서 먹은 밥. 이 두 접시가 5만원인건 범죄에 가깝다.

(a) 시금치감자조림

(b) 연어 스테이크

(c) 과일 콩포트 타르트

Figure 1.2: 뱅퀏. 그나마 먹을 만 했다. 비주얼적으로 끔찍한 건 여전하다.

(a) PLDI 개막식

(b) 최악의 프로그래밍 언어 시상식

Figure 1.3: 학회 사진. 밥 사진에 비해 찍은 양이 월등히 적다.

2 Interesting talks

2.1 Bit Blasting Probabilistic Programs

어떤 확률분포가 있을 때 깔끔하게 수식으로 표현할 수 있는 경우는 거의 없다. 따라서 대부분은 양자화 과정을 거쳐 표현하게 되는데, 당연히 양자화하는 구간이 세밀할수록 메모리와 시간을 많이 먹게 된다. 이 연구는 이 맹점을 PS에서 많이 볼 법한 방법으로 exploit하는데, 확률분포의 구간을 2^n 개로 쪼갠 다음 각 구간을 n개의 코인 토스의 합으로 표현한다. 이 과정에서 비트마스킹을 쓰고 그래서 Bit-blasting이라고 표현한 것 같다. PLDI에서 들은 첫 발표였는데, 굉장히 재밌는 연구가 나와서 기분이 좋았다.

2.2 Compiling with Abstract Interpretation

정말 재밌어 보여서 들었는데, 하나도 이해가 안 되어서 아쉬웠다. 주 골자는 임의의 분석기가 있을 때 프로그램 자체를 Abstract domain으로 표현하면 인터프리터를 자동으로 유도할 수 있는 내용인 것 같은데, 이론적으로 꽤나 심오한 내용을 담고 있는 것 같았다. 다만 현실적으로 적용 가능성이 있는지에 대해서는 매우 회의적.

2.3 Associated Effects

Magnus Madsen 교수님의 Flix 시리즈이다. Sparse analysis를 하신 이후로 자바스크립트 분석을 깔끔하게 손절하셔서 왜 그런가 싶었는데, Effect system에 빠지시면서 Flix를 풀타임으로 개발하시는 듯 하다. Flix에 대해서는 굉장히 재밌는 언어라고 생각하지만, 예전에 실행해보려고 튜토리얼을 할 때 Hello World 코드마저 버그가 있었던 관계로 꽤나 신뢰하긴 어렵다고 생각한다. 토이 언어에 그 정도까지 기대하는 것이 무리인가?

발표 자체는 굉장히 재미있었다. 어떤 Effect들이 존재할 때 Effect에도 Option 비스무리한 게 존재할 수 있는 것이 당연하다(Effect는 생길 수도 있고 없을 수도 있는 것이니). 따라서 여러 Effect가 발생하거나 발생하지 않을 수 있을 때 이를 적절히 추론하거나 합성할 수 있는 기술이 필요하다. Flix라는 언어를 만들면서 발생하는 문제를 차근차근 해결해나가고 있는 것 같은데, 좋은 연구의 줄기를 잡은 것 같아서 부러웠다. 물론 그렇다고 Flix를 유지보수하고 싶지는 않다...

2.4 Numerical Fuzz: A Type System for Rounding Error Analysis

실수 오차는 항상 문제가 된다. 하지만 실수 오차를 피하는 것은 현실적으로 불가능하니 차선은 실수 오차가 얼마나 날 수 있는지를 파악하고 문제가 되는 경우에는 오차를 줄이는 것이다. 하지만 실수 오차가 얼마나 나는지를 파악하는 것은 굉장히 비직관적이고 labor-intensive한 job이다. 특히 나 같은 경우에는 계산 기하 문제를 출제할 때 실수 오차 정당성을 증명해야 하는 경우가 많았어서 꽤나 골치를 썩었다.

이 연구는 실수 오차를 굉장히 신선한 관점으로 접근한다. 어떤 타입 $T[p]$를 실수 오차가 최대 p만 발생하는 연산으로 정의한 다음, 타입의 mapping을 정의해서 실수 오차를 효과적으로 bound한다. 아이디어 자체도 괜찮았는데, 벤치마크가 정말 압도적이었어서 기억에 남는다.

2.5 Stream Types

올해의 김준겸 어워드 수상작이다. Streaming, 그러니까 Incremental problem은 정보 기술이 발달되면서 정말 흔하게 볼 수 있는 문제가 됐다. 하지만 incremental algorithm을 직접 짜는 건 생각보다 까다롭고 귀찮을 수 있는데, 이 발표는 incremental algorithm이 실제로 잘 동작하는지, 또 자동으로 변환할 수 있는지에 대한 안전장치를 제공한다. PLDI에서 타입으로 서커스를 하는 분들이 많은데 그 중에서 제일 멋진 곡예를 보여 주신 것 같았다.

여담으로 발표자 분이 본인 연구에 정말 애정이 많고 재밌어 보이셨다. 그런 연구를 하면서 살면 재밌을 것 같긴 했다.

2.6 The Functional Essence of Imperative Binary Search Trees

흔치 않은 자료구조 연구이다. 기본적으로 함수형 언어는 명령형 언어보다 느리다(이유를 묻지 말자, 부정할 수 없는 사실이다!). 그런데 이 연구는 Koka라는 새로운 언어 위에서 로우레벨 최적화를 기반으로 함수형 언어로 작성된 이진 탐색 트리의 퍼포먼스를 명령형 언어급으로 끌어올렸다. 기본적인 아이디어는 Reference counting을 이용하여 가능한 모든 연산을 in-place로 진행해 추가적인 메모리 할당을 제거하는 건데, 생각했던 방향과는 달랐지만 재밌었다.

2.7 Bringing the WebAssembly Standard up to Speed with SpecTec

KAIST 류석영 교수님 연구실의 윤동준 박사(진)님이 하시는 SpecTec 연구이다. 저번 겨울 SIGPL 때 간단하게 소개를 들은 적이 있어서 전체적인 개괄은 알았지만 디테일에 대해서는 무지했다. 이번에 꽤나 자세하게 들을 수 있어서 좋았다. 전반적으로는 리버스-ESMETA라고 생각해도 될 것 같았는데, '스펙을 파싱하지 말고 애초에 분석할 수 있을 정도로 잘 쓰여진 스펙을 작성하면 되는 거 아니냐?' 라는 말로 요약할 수 있을 것 같다. Evaluation이 참 힘들었을 것 같다는 생각을 했다.

2.8 SuperStack: Superoptimization of Stack-Bytecode via Greedy, Constraint-Based, and SAT Techniques

별 기대 안하고 들은 발표였는데 생각보다 굉장히 재밌었던 발표이다. WASM같이 Stack을 사용하는 바이트코드 언어를 최적화할 때 탐색을 통해 semantic이 같은 코드를 합성함으로서 최적화를 할 수 있는데, 탐색 공간이 당연히 너무 크므로 이를 좀 줄여야 한다. 이 연구에서는 Greedy 알고리즘을 사용해서 bound를 개선한 다음, loose하게 정의된 SAT로 모델링해서 다이렉트로 때려박아버렸다. bound만 잡는거면 greedy 알고리즘이 아닌 meta-heuristic을 사용할 수 있지 않을까 싶어서 질문해봤는데, 생각보다 그렇게 간단하지 않은가 보다(SAT 모델링을 다시 해야 한다는 답변이 돌아왔다).

2.9 Floating-Point TVPI Abstract Domain

Abstract domain 연구는 항상 재밌다! 이 경우에도 실수 오차를 다루는 도메인이었는데, 정확히 말하면 coefficient가 range로 표현되는 상황에서의 2D-polygon abstract domain이라고 표현하는 편이 좋을 것 같다. 이런 경우에는 항상 도형이 convex하지 않을 수 있어서 예외처리 등이 까다로울 것 같았는데, join 등을 어떻게 한 지 몰라도 잘 정의했다는게 신기해서 좀 더 읽어보고 싶다.