Trip Report on OPLSS 2010

Logic, Languages, Compilation, and Verification

June 15 – 25, 2010, Eugene, Oregon, US

임현승 (genilhs@postech.ac.kr)

Oregon Programming Languages Summer School (OPLSS) 에 포항공대 박성우 교수님과 박종현군과 함께 다녀왔습니다. Logic, Languages, Compilation, and Verification이라는 주제로 이름만 들어도 긴장되는 대가 분들께서 매우 훌륭한 강의를 해주셨습니다. 작년 여름학교에도 참석했었는데, 작년과 비교해서 프로그램 구성도 체계적이었고 대부분의 강의도 흡족할 만한 수준이었습니다.

그림 1 강의동 앞에서 왼쪽부터 박성우 교수님, 박종현군, 필자

1. Proof Theory Foundations by Frank Pfenning

이번 여름학교는 Pfenning 교수님의 강의와 함께 시작했습니다. 박성우 교수님의 지도교수님이라서 그런지 강의 스타일과 내용이 박성우 교수님의 Logic in Computer Science 수업과 매우 흡사했습니다. 덕분에 강의 내내 편하게 복습하는 기분이었습니다. 그러나 Natural deduction, sequent calculus, focusing[1] 등에

[1] Theorem proving 시에 proof search space를 줄이기 위해 사용하는 방법; 박성우 교수님께서 CMU에서 수업을 들었을 때만해도 아직 연구 중인 주제였다고 하시더군요.

대한 배경지식이 없던 친구들은 다소 어려워 했습니다. :)

2. Type Theory Foundataions by Robert Harper

Pfenning 교수님에 이어 Harper 교수님께서 logical relations 기법을 이용한 termination 증명에 대해서 강의해주셨습니다. Pfennig 교수님의 수업과 마찬가지로 박성우 교수님의 Programming Languages 수업의 연장선인듯한 느낌이 강했습니다. :) Harper 교수님께서는 먼저 simply-typed lambda calculus의 termination 증명에 대해 소개하시고 Girard의 System F termination 증명을 소개해주셨습니다. 특히 System F의 경우 termination 증명 방법이 logical relations을 이용하는 방법 밖에 없다고 하시면서 logical relations이 정말 우아하고 강력하다는 것을 시사하셨습니다. 개인적으로 module systems을 공부하면서 type equivalence 문제 때문에 logical relations을 공부한 적이 있었는데, 단순히 rule induction이나 structural induction으로는 증명할 수 없는 문제가 logical relations을 이용하면 깔끔하게 풀리더군요. 다소 어려운 내용이긴 하지만 배워두면 매우 유용한 툴인 것 같습니다. Logical relations에 관심 있으신 분들은 Advanced Topics in Types and Programming Languages Chapter 6을 보시면 도움이 많이 될 겁니다. :)

3. Essential Coq from Scratch by Andrew Tolmach

Coq 기초 강의. 이미 Coq에 대해 어느 정도 지식이 있어서 다소 지루한 감이 없지 않아 있었습니다. Coq tactics을 natural deduction의 introduction/elimination rules과 연관 지어서 설명했다면 Coq을 처음 접하는 사람들이 보다 쉽게 다가갈 수 있지 않았나 하는 아쉬움이 남는 강의였습니다.

4. Software Foundations in Coq by Benjamin Pierce

Types and Programming Languages(TAPL)의 저자이자 Harper 교수님의 제자로 PL을 공부하는 사람이라면 이름만 대도 알만한 Pierce 교수님께서 Tolmach 교수님 수업에 이어 Coq을 이용하여 programming languages의 meta-theory를 mechanize하는 방법에 대해 강의해주셨습니다. 전체 강의는 Software Foundataions in Coq 후반부 내용 위주로 진행되었습니다.

이번 여름학교에 KAIST 김지응군과 황준형군이 함께 참석했었는데 김지응군이 다소 깨끗한 TAPL 책에 Pierce 교수님의 사인을 받았습니다. 조금 부럽더군요. 제 지저분한 TAPL 책을 가져왔다면 정말 좋았을 텐데 하는 아쉬움이 남았습니다. :)

5. Programming Language Methods for Compositional Security by Anupam Datta

McBride, Constable 교수님 강의와 더불어 가장 힘들었던 강의입니다. (>.<) 여러 개의 안전한 컴포넌트(secure components)를 이용하여 하나의 시스템을 설계했을 때 (by sequential or parallel composition), 그 시스템의 안전성(security)을 logic system을 이용해서 증명하는 방법에 대해 설명해주셨습니다.

6. Proofs-as-Processes: Reasoning about Concurrency in Computational Type Theory by Robert Constable

Nuprl theorem prover를 설계하신 분이며 Harper 교수님의 지도교수님이신 Constable 교수님께서는 강의 초반에는 강의 주제와는 다소 동떨어져 보이는 type systems을 formal하게 formulate하는 방법과 Martin Löf type system에 대해서 이야기 해주셨습니다. (>.<) 마지막 강의에서 Nuprl theorem prover에서 간단한 theorem이 어떻게 증명될 수 있는지 보여주시고 computational type theory에 대해서 간략히 소개해주셨습니다. (>.<)


그림 2 열정적으로 강의 중인 Pierce 교수님 (좌) / 신의 손 서울대 영범이형 - 정말 명키퍼가 따로 없었다. (우)


그림 3 강의 중인 Constable 교수님 (좌) / 일요일 Eugene 근처 해안가에 가는 중에 - 뒤에 Xavier Leroy 교수님의 뒷모습과 Benjamin Pierce 교수님의 옆모습이 매우 인상적인 사진 (우)


그림 4 Eugene 근처 해안가가 아니라 호수 ㅜㅜ 태평양을 보고 싶었는데 다소 실망을 금할 수 없었다. (좌) / 왼쪽부터 황준형군, 필자, 박종현군 - 누가 누가 높이 뛰나 경쟁 중입니다.

7. Proving a Compiler: Mechanized Verification of Program Transformations and Static Analyses

by Xavier Leroy

OCaml 설계 최전선에 계시며 최근에는 compiler verification으로 명성이 드높으신 Leroy 교수님께서 compiler verification에 대해 훌륭한 강의를 해주셨습니다. Coq을 이용하여 간단한 imperative language 정의에서부터 시작하여 virtual machine으로의 컴파일, dead code elimination, register allocation 등의 optimization 기법 구현 등의 과정이 correct 함을 보이는 방법에 대해서 차근차근 설명해주셨습니다. 또

한 abstract interpretation을 도입하고 correctness를 증명하는 방법에 대해서도 강의해주셨습니다. 끝으로 realistic compilers의 formal verification에 대해서 연구하는 Compcert Verified Compiler 프로젝트에 대해서도 간략하게 소개해주셨습니다.

여담으로 다른 교수님 강의 시간에 Leroy 교수님 근처에 앉은 적이 있었는데 옆에서 보니 Coq 코딩을 마치 워드로 타이핑하듯이 하시더군요. Automated theorem prover가 따로 필요 없으신 분 같으셨습니다. 덜덜덜...(>.<)


그림 5 학생의 질문에 답변 중이신 Leroy 교수님 (좌) / University of Oregon 문패 앞에서 다 함께 (우)

8. Ynot Programming by Greg Morrisett

Harper 교수님의 또 다른 제자 Morrisett 교수님께서는 robust proof generation에 대해서 강의해주셨습니다.[2] Proof assistant를 이용하여 증명을 mechanize할 때의 주목적은 증명 그 자체 있지 남들이 증명을 이해하는 데에 있지 않음을 설파하시며 많은 논란을 불러일으키셨습니다. Simply typed lambda calculus와 Hoare type theory를 예로 들며 증명을 최대한 간단하게 하면서 syntax가 확장되거나 새로운 내용이 추가될 때, 또는 기존 내용이 변경될 때 기존 proof가 최대한 수정되지 않도록 하는 방법에 대해서 강의해주셨습니다. Coq으로 새로운 tactic을 정의해서 사용하시던데 그 tactic을 이용하면 마치 마법처럼 대부분의 lemmas가 손쉽게 증명되더군요. :)

9. Dependently Typed Programming by Conor McBride

여름학교의 대미를 장식한 McBride 교수님 강의시간에는 의지가 매우 강한 사람이라 해도 다소 멀리 갔다 올 수 밖에 없었을 것입니다. 총 다섯 시간 중 첫 시간은 Haskell을 이용해서, 남은 네 시간은 Agda를 이용해서 dependently typed programming에 대해 강의하셨습니다. Haskell과 Agda에 대한 지식도 별로 없는데 실험적인 내용도 많아서 강의 내용이 많이 어렵더군요. 강의 내용과는 별도로 McBride 교수님은 다소 특이한 분인 것 같았는데 OHP 필름을 이용해서 프레젠테이션을 하셨고 맨발로 강의실을 돌아다니셨습니다. 또한 Lisp을 이용하여 symbolic computation에 관한 phd thesis를 작성하셨던 아버지 때문에 6살 때부터 differential equation과 lambda calculus를 배워야 하는 아동학대를 당하셨다고 하시더군요. (>.<)

---

[2] 이번 여름학교는 Harper 교수님을 중심으로 강사진이 촘촘하게 엮여져 있는 듯한 느낌이었습니다. Harper 교수님의 지도교수님, 제자들, 같은 학교 친구분들이 주 강사진이었기 때문입니다.

맺음말

이번 여름학교는 최고의 강사진으로 구성된 만큼 총 80여명의 학생들이 참석하는 기염을 토했습니다. 한국 학생들도 총 7명이 참석했습니다. 강의 내용도 전반적으로 매우 훌륭해서 많이 배우고 올 수 있었습니다. 여름학교에 참석한 다른 외국학생들도 전반적으로 매우 만족하는 분위기였습니다. 작년에 이어 올해에도 여름학교에 참석할 수 있는 좋은 기회를 주신 박성우 교수님께 감사를 드리며 본 글을 마칩니다.

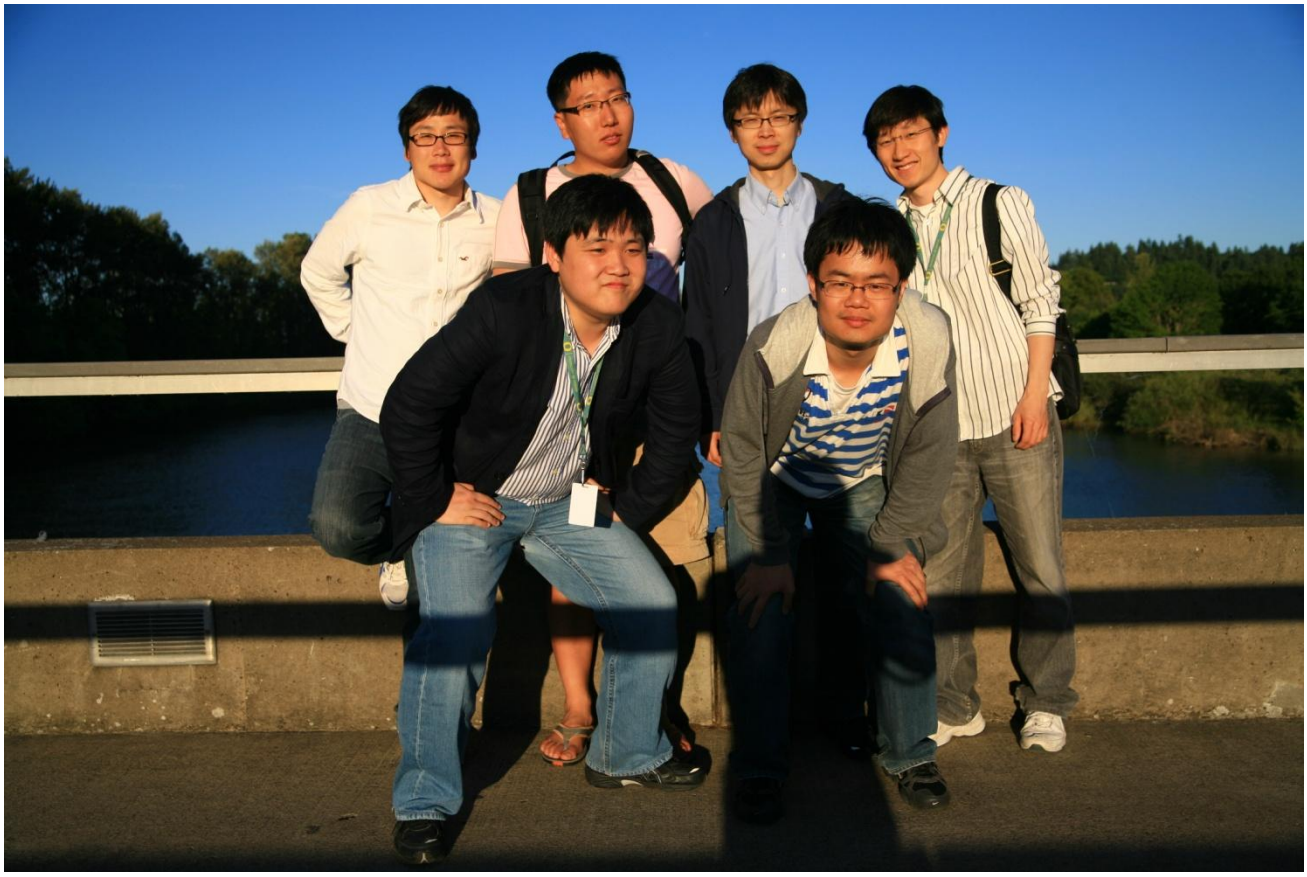

그림 6 단체사진: 뒤에서 왼쪽부터 필자, 서울대 영범이형, MIT 덕환이형, 서울대 학주, 앞에서 왼쪽부터 포항공대 종현, 카이스트 준형, 그리고 그림자는 사진을 찍어준 카이스트 지응이 :)