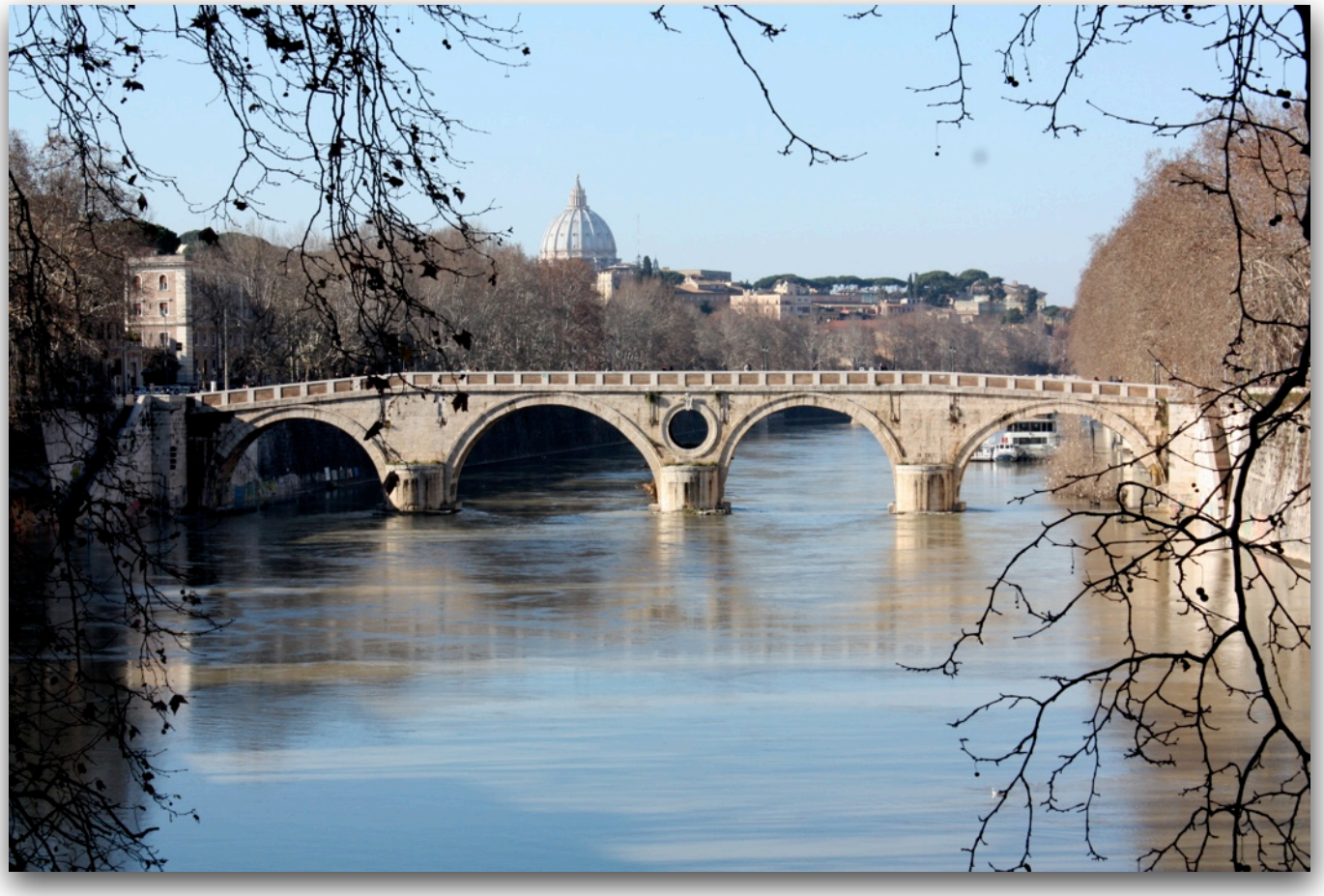


# POPL/VMCAI 2013

## ROME, ITALY

2013.01.20-2013.01.26


오학주

서울대학교 프로그래밍 연구실


로마에서 열린 POPL 2013에 다녀왔다. 올해로 40번째로 열린 POPL은 VMCAI, PADL, PEPM 등 다양한 위성학회 및 워크샵과 함께 열렸다. 다양한 사람들과 그들의 연구를 체험할 수 있었다. 일주일간 로마에 머무르면서 보고 느낀점을 공유하고자 한다.

# 1.개요

POPL은 수요일부터 금요일까지 진행되었고, 앞 뒤로 다양한 학회들이 열렸다. 먼저 매년 함께 열리는 VMCAI(International Conference on Verification, Model Checking, and Abstract Interpretation)가 일요일부터 화요일까지 열렸고, PADL(International Symposium on Practical Aspects of Declarative Languages)와 PEPM (Partial Evaluation and Program Manipulation)이 월,화에 걸쳐서 열렸다.

그 외 많은 워크샵이 있었다. 이번에 처음 열린 ID(Interference and Dependence)란 워크샵에서 malware관련 발표가 있길래 들어가 봤는데 10명정도가 모인 소규모임에도 매 발표마다 질의응답이 활발했다. 이번에 두번째로 열린 PLMW (Programming Language Mentoring Workshop)은 학생들을 대상으로 프로그래밍 언어 분야의 주요 이슈가 무엇인지, 연구는 어떻게 하며, 논문은 어떻게 쓰는지에 대한 워크샵이었다. Xavier Leroy, Peter O'Hearn 등 대가들로부터 직접 좋언을 들을 수 있는 기회였다. 이 워크샵은 학생들뿐 아니라 POPL참가자 거의 모두에게 인기가 많은것 같았다.  많은 사람들이 이 워크샵에 참석해서 마지막날  VMCAI 몇몇 세션들이 텅텅 빌 정도였다.

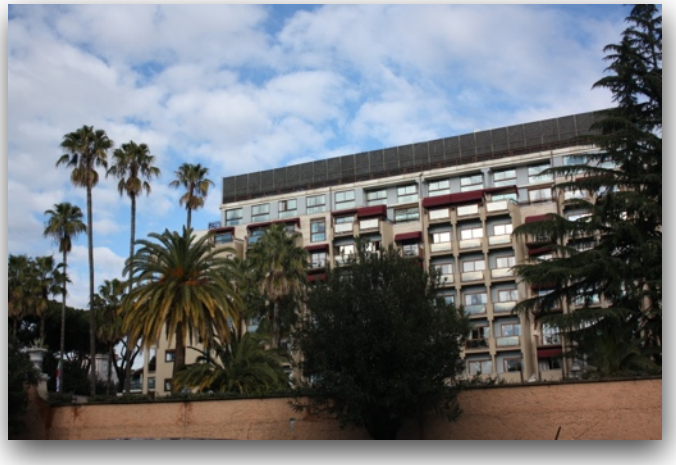
POPL이 진행되는 수요일부터 금요일까지 사람들이 가장 많았다. coffee break 시간은 사람들로 북적였고, talk이 시작되어도 남아서 못다한 얘기를 하는 사람들도 많았다. POPL에서는 총 43편의 논문이 발표되었고 A,B 두 세션으로 동시에 진행되었다.

# 2.VMCAI

로마에 19일 밤 11시 30분 경에 도착하였는데,VMCAI는 바로 다음날 아침에 시작했다. 학회는 로마에서 가장 비싼 호텔중 하나라고 하는 Parco dei Principi Hotel에서 열렸다. 우리가 묵는 호텔은 학회장에서 1km정도 떨어진 곳에 있었는데 학회장까지 가는데 10분정도 걸렸다. 멀지는 않았지만 로마시내 골목이 깔끔하게 정리되어 있지는 않아서 첫 며칠은 찾아가는데 애를 먹기도 했다.

VMCAI는 이름에서도 알 수 있듯이 검증(Verification), 모델체킹(Model Checking), 요약해석(Abstract Interpretation)의 최신 연구결과가 소개되는 정적 분석 분야 주요 학회중 하나이다. 이 세 분야는 모두 지향하는 목표가 소프트웨어 검증/분석으로 비슷하지만, 각 커뮤니티가 따로따로 성장하여 왔다.VMCAI는 세 분야의 기술들을 결합하여 시너지 효과를 내자는 취지에서 만들어졌다. 하지만 아직 만들어진지 오래되지 않아서인지, 각 분야의 기술들을 통합하거나 서로간의 관계에 대한 연구들은 많지 않고, 세 분야의 논문들이 세션별로 나뉘어서 발표되고 있는 느낌을 받기도한다.

지난 3년간 우리랩에서 VMCAI에 논문을 한편씩 발표했었는데, 올 해는 논문이 없어서 아쉬웠다. 대신 졸업생인 순호형이 발표한 논문이 마지막 세션에 있었다. 흥미롭게 들었던 논문들은 아래와 같다.



## Automatic Inference of Necessary Preconditions

Microsoft Research의 Francesco Logozzo, ENS의 Patrick Cousot, Radhia Cousot로 이루어진 팀에서 작년에 이어서 올해에도 주어진 명령문이 실행되기 위한 Precondition을 찾는 문제에 관한 논문을 발표하였다. 작년 VMCAI 논문에 대부분의 내용이 있었고, 올해 논문은 그 내용을 구현하고 실험한 결과였다. 요지는 자동화된 방법으로는 sufficient precondition대신 necessary precondition을 찾는게 더 유용한 precondition을 찾을 수 있다는 것이었다.

Sufficient precondition이란 프로그램이 옳바름을 보이는데 충분한 조건을 말한다. 즉, sufficient condition을 만족시키는 입력하에서는 프로그램이 올바르게 동작함이 보장된다. 흔히 말하는 weakest precondition도 sufficient condition이다. Necessary condition이란 그 조건을 만족시키지 않으면 프로그램 실행중에 에러가 발생하는 조건이다. 즉, 에러가 발생하지 않기 위한 필요조건이다.

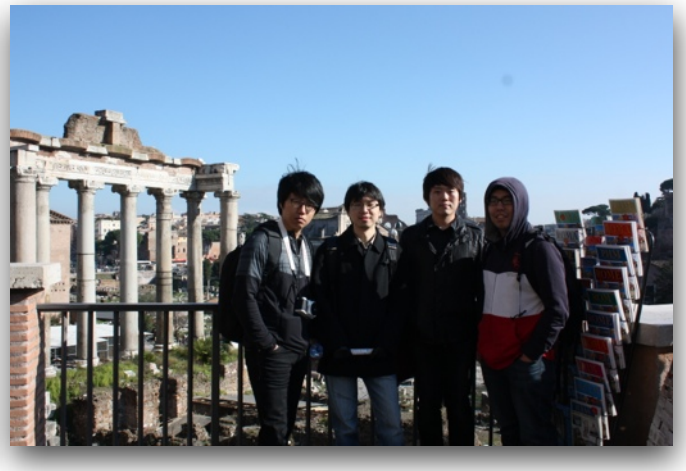
이 논문이 다루는 주제는 주어진 statement의 precondition을 구하는 것이다. 이 논문이 주장하는 바는 자동화된 방식으로는 necessary condition을 구하는것이 훨씬 유용하다는 것이다. Sufficient condition은 자동으로 유용한 것을 찾기가 쉽지 않다. Weakest precondition을 생각해보면, 룹이 있으면 weakest한 조건을 찾기가 쉽지 않고 대부분의 경우 근사하게 되는데, 근사를 하면 실제로는 대부분 유용하지 않은 precondition이 나오게 된다는 것이다.

이 논문에서는 necessary precondition이 왜 필요한지 상세히 설명하면서, 이를 자동으로 구하기 위한 알고리즘들을 제시하였다. Clousot라고 하는 MSR에서 개발중인 분석기에서 실험하였는데, 실제 Windows 코드에 적용하여 실험한 결과가 인상적이었다.

## Complete Abstractions Everwhere

익숙한 분야라 가장 듣기 편했지만, 과연 유용할지 의문이 드는 연구였다. 이 발표는 초청강연으로 이탈리아 Padova대학의 Francesco Ranzato 가 연사였다.

주요 내용은 정적 분석에서 완전성(completeness)를 많이 찾아볼 수 있다는 것이었다. 보통 정적분석은 안전하지만(sound) 완전하지는 않으므로 무슨 말인지 처음부터 궁금했다. 듣고보니 완전성에 대한 정의가 보통 정적분석에서 사용하는 의미와는 달랐는데, 주어진 요약 도메인에 대해서 가장 정확한 요약의미를 완전하다고 불렀다. 즉, 최대 정확한 요약의미(best abstract transformer)를 completeness라고 부르는 것 같았다. 정적 분석을 설계하다보면 주어진 요약 도메인에 대하여, 가장 정확한 요약의미를 가지지 않는 요약의미를 설계하는 경우가 흔하다. 가장 정확한 요약의미는 실제화(concretization)하고 실제 의미(concrete semantics)를 적용한 후 다시 요약(abstraction)한 것인데, 이를 계산하기 불가능한 경우가



많기 때문이다. 이 논문은 요약의미가 주어졌을때, 그것에 complete하면서 최소인 요약 도메인을 찾는 방식을 제시하였다. 최대한 정확한 요약의미함수를 정적분석에서 써서 정확도를 높이자고 주장하는것 같았는데, 이 방법이 실제로 얼마나 효과가 있을지는 의문이 든다. 어차피 요약 도메인 자체가 가지는 한계로 인한 정확도 저하를 해결하는 방법이 아니기 때문이다. 실제로는 굉장히 미미할 것 같은데, 이를 위해서 이론을 만들것까지 있나 싶은 생각이 들었다.

### Logic as the lingua franca of software verification

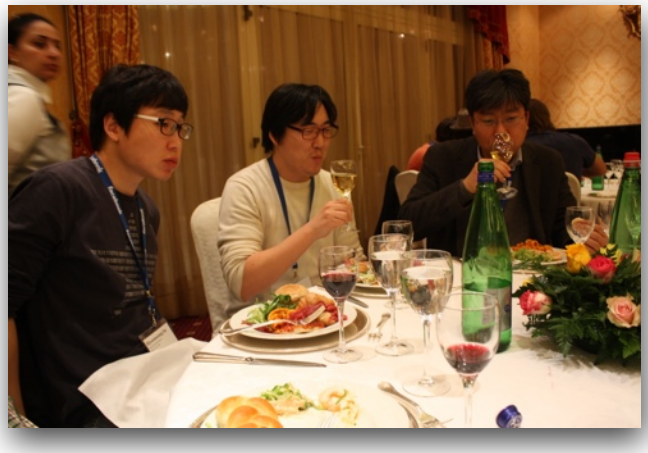
Ken McMillan의 초청강연이었다. Ken McMillan은 모델체킹 분야에서 Binary Decision Diagram, interpolation 등의 기법을 모델체킹에 적용함으로써 새 연구분야들을 개척한 스타이다. 난 이번에 Ken McMillan을 처음 봤는데, 명성에 비해서 나이가 많지 않아 보였다. 발표의 내용은 software verification 기법들이 다양한 방법으로 개발되고 있는데, 공통적인 프레임웍 혹은 공통 언어(lingua franca)로써 logic을 사용하자는 것이었다.

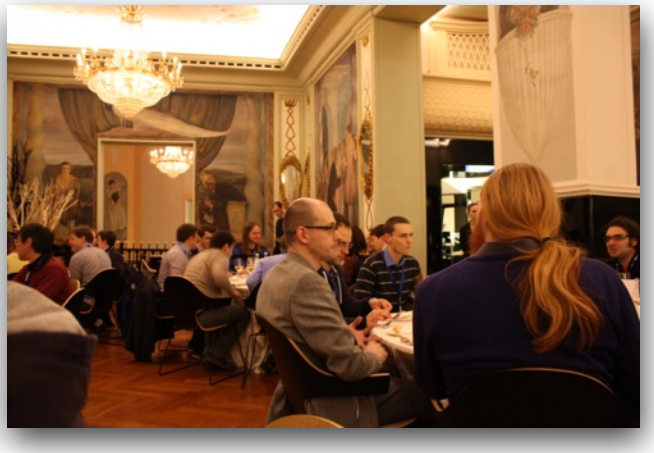
Logic 기반의 프로그램 분석이 좋은 이유에 대해 강조하였다. 하나의 분석기/검증기로 만족할만한 결과를 얻는것은 흔하지 않으므로, 여러개를 연결하여 서로 정보를 교환하며 결과를 향상시키는 방법이 많이 쓰인다. 이 때 각각의 분석기들이 사용/표현하는 언어가 다르다면 곤란할 것이다. 예를 들어, A란 분석기는 P란 특성을 분석하고 분석기 B는 특성 Q를 분석하는데, 이 둘을 결합하고 싶다고 할 때, A와 B가 내부적으로 사용하는 표현방식이 같아야 결합작업이 쉬울 것이다.

이 발표 뿐 아니라, 이곳저곳에서 logic 기반 검증/분석을 강조하는 것을 들었다. 요약해석을 lattice기반이기 때문에 나는 logic 기반 분석에 많이 익숙하지는 못하다. 그런데 Ken McMillan은 아예 대놓고 모든 분석을 logic기반으로 옮겨가자고 하는 것 같았다. PLMW에서 Peter O'Hearn의 발표에서도 비슷한 내용이 있었는데, 요약해석으로는 loop invariant를 찾고, interpretation은 logic 기반으로 설명하였다. 또한 최근에 포인터 분석이 logic기반으로 설계하려는 시도가 진행되고 있다고 한다.

## 3.POPL

POPL은 프로그래밍 언어 분야내의 다양한 연구 결과들이 발표되는 곳이다. 올해도 다양한 주제들(semantics, verification, static analysis, types, logic, security, concurrency 등)로 구성되었고, 두 세션이 동시에 병렬로 진행되었다. 나는 주로 static analysis, verification, abstract interpretation 세션에 있었다. 내 분야 혹은 인접분야의 발표들은 큰 흐름을 이해하는데 큰 문제는 없었지만, 역시 다른 분야의 연구들은 배경지식 부족으로 인해 문제가 무엇인지, 그것이 왜 중요한지에 대한 감을 잡기도 쉽지 않았다. 배경지식을 늘려야겠다는 생각을 했고 다른 분야의 사람들도 문제와 아이디어 정도는 이해할수 있게끔 쉽게 발표를 구성하는데 노력해야겠다는 생각을 했다.





정적 분석, 특히 요약해석(Abstract Interpretation) 관련 논문이 적은것은 아쉬웠다. 유럽에서 POPL이 열려서인지 세션이름이 Abstract Interpretation인 세션이 있었고 3편의 논문이 발표되었다. 하지만 이중에서 두 편은 요약해석의 개념이 일반적으로 쓰였을뿐 큰 관련은 없었다. 요약해석의 이론위에서 잘 정립된 정적 분석이 실용적으로 쓰이기 위해서 가야할 길이 아직 멀어보이는데, 이 분야 연구를 하는 사람은 많지 않아 보였다.

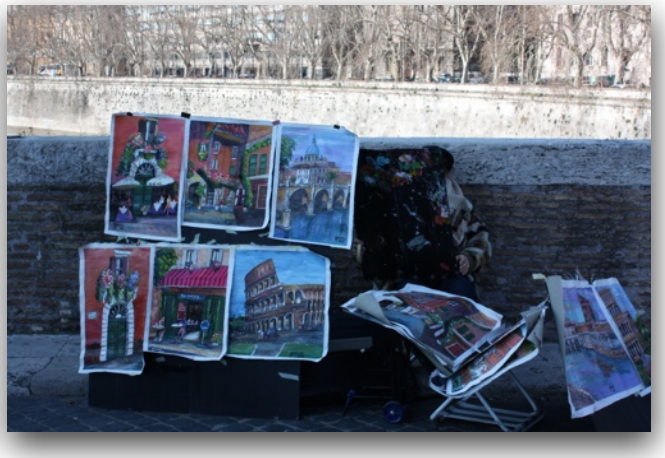
발표를 듣는것도 좋았지만, 매 세션이 끝나고 커피잔을 하나씩들고 이뤄지는 난상토론시간이 학회의 묘미인것 같다. 두 세 사람씩 모여서 이전의 발표에 대해서 질문하기도 하고, 서로의 연구내용을 교류하는 모습은 언제나 인상적이고 자극을 받는다.

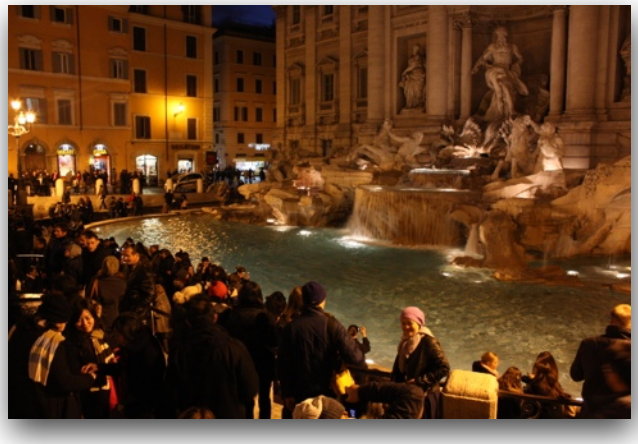
학회에 다닐수록 학계 사람들에 조금씩 익숙해지는것 같다. 예전에 학회갔을때 만났었던 사람들을 다시 만났다. 먼저 덕환이형, 임현승 박사, 어덕기 박사님을 만나서 오랜만에 많은 이야기를 나눴고 서로의 연구에 대해서도 교류하였다. Peter O'Hearn 교수님도 다시 만났는데, 옆에 있던 Byron Cook에게 나를 "localization guy"라고 소개하셨다. 덕분에 Byron에게 요즘하는 일들에 대해서 얘기하는 기회를 얻었다. UC Berkeley의 Vijay D'Silva는 원찬이와 우석이에게 하도 자주 들어서 직접 만나기 전부터 친숙했었다. Vijay와 쉬는시간, 점심시간에 종종 만나서 나눈 이야기들도 즐거웠다. 특히 어느 점심시간에 요즘 내가 하는 연구에 대해서 토론을 했는데, Vijay의 모든것을 abstract interpretation으로 설명하려는 시도가 신선했다. Vijay의 소개로 Josh Berdine과도 안면을 틀수 있었는데 비슷한 분야에 있어서인지 내 연구들을 잘 알고 있었다. 교수님의 소개로 David Monniaux과 이야기를 꽤 오래 하게되었다. 스파스 분석에 대해서 궁금해하셔서 설명했더니 자신도 예전에 시도해 보고싶었던 아이디어라고 했다. 지난 POPL의 student session에서 만났던 Zachary Kincaid를 다시 만났다. 캐나다 토론토 대학교의 박사과정 학생인데, 스파스 분석관련 연구를 하는 친구이다. 작년 student session에서 같은 조에 속해있었는데 서로 비슷한 내용을 발표하게되어 알게되었다. 이번 POPL에서 스파스 분석을 검증(verification)에 적용한 논문을 발표하였다. 발표후 쉬는 시간에 만나서 인사와 함께 근황을 물었다. 앞으로는 모양분석(shape analysis)에 스파스 분석 아이디어를 적용해 보는 것을 시도해볼 계획이라고 한다. 어떤 내용이 될지 궁금하다고 기대하겠다고 했다.

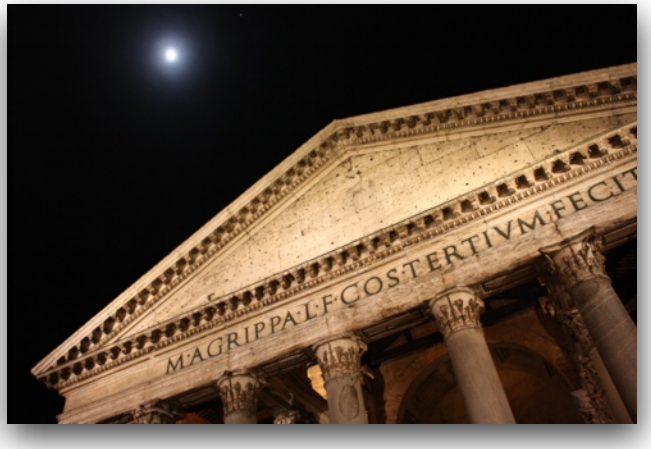
올해 POPL 발표중 몇가지를 아래에 소개한다.

## Fully abstract compilation to JavaScript

정말 훌륭한 일을 했다는 생각이 드는 발표였다. ML류의 함수형 언어로 짜여진 프로그램을 자바스크립트로 변환하는 컴파일러에 대한 내용이다. 특징은 full abstraction, 즉 소스 프로그램과 대상 프로그램의 동작이 모든 환경에 대해서 정확히 동일함이 보장된다.








자바스크립트 코드를 만들때, 자바스크립트로 직접 프로그램을 작성하는것이 아니라 ML과 같은 상위 언어에서 프로그램을 작성한 후 자바스크립트로 컴파일하는 경우가 많다고 한다. 그러나 기존 자바스크립트 컴파일 툴들은 full abstraction을 보장하지 않았다고 한다. 즉, 소스 프로그램(ML)에서 작성시 기대했던대로 정확히 컴파일된 자바스크립트 코드가 동작하지 않는 경우가 존재할 수 있다.

이 논문에서 발표한 컴파일러는 소스 프로그램과 대상 프로그램의 의미가 정확히 같음을 보장해준다. 그래서 프로그래머는 상위 언어로 프로그램을 작성하고, 그것의 옳바름을 상위언어에서 증명한 다음에, 자바스크립트로 컴파일한다. 그러면 상위언어에서 증명한 성질들이 모두 고스란히 자바스크립트 에서도 성립하게 된다. 이 과정을 ML류 언어인 f*를 JavaScript로 변환하는 과정이 옳음을 Coq으로 보였다.

프로그래밍 언어 이론이 실제 프로그래밍 환경에 적용될 수 있는 좋은 예인것 같다. 파급력 있는 좋은 문제, 단단한 이론과 구현을 보면서 부러웠고, 좋은 연구에 매진해야 겠다는 자극을 받았다.

## Abstract conflict clause learning

Vijay D'Silva의 발표였다. 내용은 SAT solver의 CDCL(Conflict Driven Clause Learning) 알고리즘을 abstract interpretation으로 설명한 것이다. Vijay의 최근 논문들은 SAT solver의 알고리즘들을 요약해석으로 설명하는 것인데, 그 연장선의 하나이다.

로마로 가는 비행기에서 Vijay의 예전 논문 몇개를 읽어보았는데 꽤 흥미로웠다. SAT solving을 요약해석으로 설명하는것이 흥미로운 이유는 최근 몇년간 이루어진 SAT solver의 성능향상 때문이다. 요즘 SAT solver는 주어진 식이 satisfiable인지 아닌지를 굉장히 빠르게 결정해준다. 이것이 놀라운 이유는 SAT solving 기본 알고리즘을 요약해석으로 설명하면 constant propagation 도메인을 가지고 고정점을 찾는 문제로 볼 수 있기 때문이다. 하지만 보통의 요약해석기는 SAT solver에 비해서 빠르지도 정확하지도 않다. SAT 커뮤니티에서 이루어놓은 성능향상기법들을 요약해석으로 이해해서, 요약해석기에 적용할 수 있도록 하자는 것이 Vijay의 궁극적인 목표이다. 이를 위해서 SAT 알고리즘의 대표적인 기법들인 DPLL, BCP, CDCL 등을 요약해석으로 설명해 오고 있다. 이번 POPL에서는 CDCL을 요약해석으로 설명하였다.

Vijay 발표 중 마지막 슬라이드에 향후 연구 내용이 있었는데, 그 중에서 스파스 분석과 SAT solving의 관계가 있었다. 시간이 없어서 발표때에는 언급이 없어서 나중에 물어보았다. 오래 생각해본것은 아니지만 SAT 최적화 기법중에 스파스 분석과 유사한 기법이 쓰인다고 한다. 요약해석에서의 스파스 분석과 SAT에서 대응되는 기법 사이의 관계에 대해서 앞으로 연구해볼 생각이라고 하였다. 기회가 되면 같이 생각해보자고 했다.

## Inductive data flow graphs

Concurrent 프로그램을 검증하는데 있어서 data flow graph를 쓰면 증명의 크기를 획기적으로 줄일 수 있다는 내용이었다. 여기서 말하는 data flow graph란 우리가 얘기하는 data dependency graph와 본질적으로 같은 것으로, 한마디로 스파스 분석의 핵심 아이디어를 concurrent 프로그램의 검증에 쓰자는 것이다.

기존의 검증은 프로그램의 control flow를 따라 이루어지는데 비해서, 이 논문에서는 data flow를 따라 이루어진다. control flow를 따라 가면 현재 statement와 무관한 변수와 식들도 pre/postcondition 계산에 수반되는데 이 때문에 증명의 크기도 커지게된다. 뿐만 아니라 concurrent 프로그램에서 두 쓰레드의 가능한 모든 interaction을 다 고려하면 생각해야할 상태가 쓰레드개수에 따라 지수적으로 증가하게 된다. 증명을 data flow graph위에서 하게되면 sequential프로그램의 경우에 증명의 크기가 당연히 작아진다. 이 논문에서 보인것은 concurrent 프로그램의 경우에는 프로그램 내에 존재하는 data dependency의 개수에 따라서 polynomial하게 증가한다는 것이다. 즉, 프로그램내의 명령문들간의 data dependency가 적은 경우에는 증명의 크기가 획기적으로 줄 가능성이 있다. 논문에서는 data flow graph를 이용하여 검증대상을 표현하는 방법과, 전체 검증알고리즘을 설명한다.

스파스 분석의 아이디어가 요약해석 기반 정적분석 뿐 아니라, 검증, 모델체킹 등에도 적용가능하지 않을까 어렴풋이 상상해보곤 했는데, 이 논문은 구체적인 응용을 보인 셈이다. 평소 인접분야의 지식이 있었더라면 비슷한 내용의 연구를 해 볼 수 있었을텐데하는 아쉬움도 남았다. 더 열심히 내 분야 말고도 다른 분야에 관심을 가지고 공부도 해야겠다고 느꼈다.

Zachary Kincaid라는 캐나다 학생이 발표했는데, 안면이 있는 사이라 논문 발표가 끝나고 쉬는시간에 인사나 해야겠다고 생각했다. 어떤 사람과 얘기중이었는데, 관련 연구에 대해서 거의 따지다시피하는 태도로 오랜시간 붙잡고 있었다. Zachary가 곤혹스러워하다가 결국 노트북을 꺼내서 데모를 보여주고 나서야 의문이 풀린듯 싶었다. 나도 인사나 하려다가 꽤 오래 기다리게 되었는데, 때문에 허충길 박사님의 발표를 놓치고 말았다.

## Reasoning about relaxed programs

POPL 논문 발표는 아니었고, student talk이었지만 인상적이었다. MIT의 덕환이형이 속해있는 연구실의 Michael Carbin이 발표했는데, 발표자체도 아주 좋았고, 내용도 흥미로웠다.

relaxed program이란 원래 프로그램이 하는 일의 일부를 생략하거나 변형한 프로그램이다. 목적은 프로그램의 기능중 일부를 변형하여 성능을 높이는 것이다. 특정 도메인의 프로그램들은 100% 정확도를 가지지 않아도 된다. 예를 들어, 컴퓨터 그래픽 관련 프로그램의 경우 화면의 일부 픽셀은 좀 잘못출력되어도 사람이 영상을 인식하는데 문제가 없을 것이다. 즉, 완벽히 정확하게 동작하지는 않더라도 소프트웨어의 성능을 획기적으로 높일 수 있다면 좋은 경우가 있다.

Micheal Carbin은 작년 PLDI에서 relaxed program을 증명하는 방법에 관한 논문을 냈다. 프로그램을 relax시키기는 하는데, 프로그램이 특정 동작만큼은 올바르게 수행한다는 것을 증명하고 싶은 경우, relax되기 전의 프로그램의 증명을 가지고 relaxed 프로그램의 증명을 어떻게 손쉽게 얻어낼 것인가에 대한 것이다. 작년 PLDI에서 다른 세션에 있어서 발표를 듣지 못했던것 같은데, 이번에 student session에서 듣게 되었다.

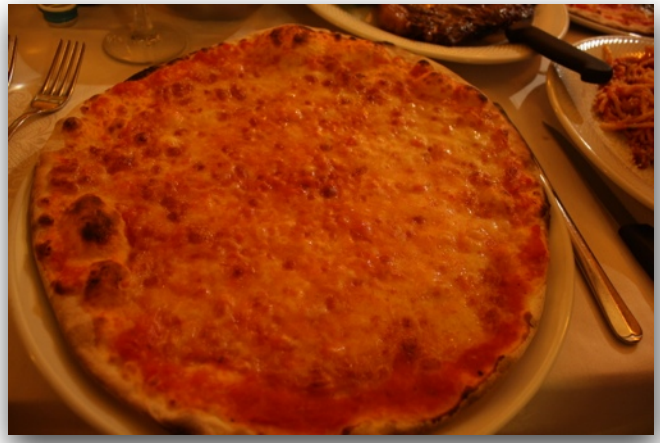
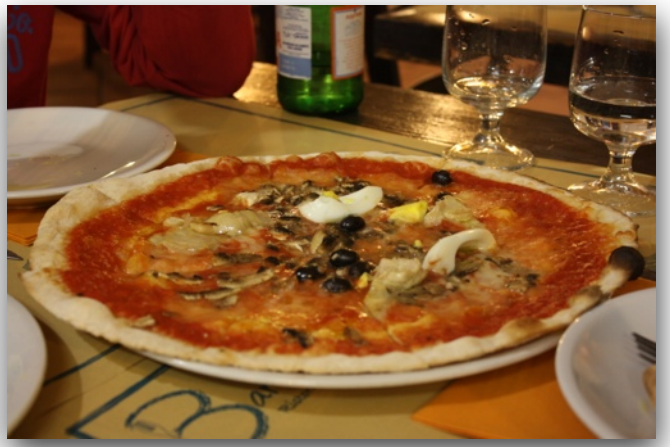
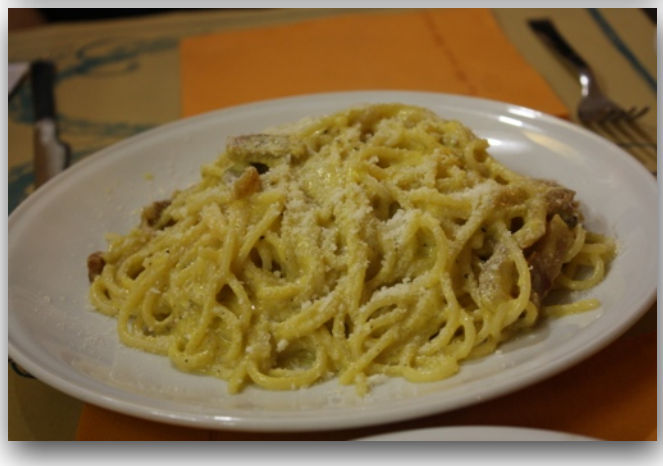
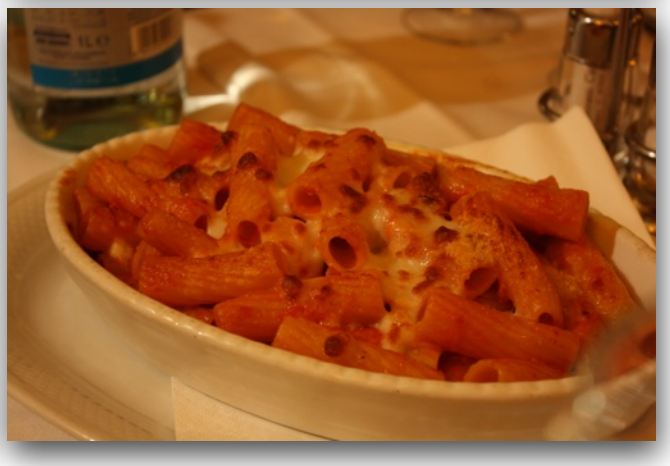
이러한 relaxed program의 아이디어를 정적분석에도 적용할 수 있지 않을까 하는 생각이 들었다. 정적 분석에서도 반드시 모든 프로그램 지점을 정확하게 분석할 필요는 없다. 언제나 관심있는 프로그램 지점과 변수가 있기 마련이고, 그 지점에서의 상태를 계산하는데 관계없는 부분은 분석이 부정확해도 된다. 그 대신 분석의 성능은 높아진다면 말이다. 이렇게 relaxed analysis가 원래 분석이 계산하려던 결과를 그대로 계산한다는 것을 증명할 수 있을것이다.

## 4.로마

로마는 과거와 현재가 자연스럽게 공존하는 도시였다. 2세기에 지어졌다는 판테온은 골목들과 상점, 집들로 둘러쌓여 있어서 어디서부터 유적지이고 생활터전인지 경계가 모호했다. 스페인 광장, 트레비 분수도 주변 거리와의 구분 없이 자연스럽게 사람들이 모이는 곳이었다.

로마는 생각보다 작았다. 6년전에 로마에 여행을 갔던적이 있는데, 유명한 관광지 위주를 다녔었다. 당시에는 항상 지하철과 버스로 이동해서 시내가 큰 줄 알았다. 하지만 이번에는 로마시내를 걸어서만 다녔는데, 대부분의 주요 유적지들 모두를 하루만에 걸어서 다닐 수 있었다. 덕분에 예전에 미처 보지 못했던 골목 구석구석을 둘러볼 수 있었다.

피자와 파스타는 원없이 먹었던것 같다. 아무리 이탈리아라지만 레스토랑들이 거의 다 피자집이었다. 마리게리타, 고르곤졸라 등의 피자는 우리나라에 있는 이탈리아식 피자와 비슷하면서 좀 더 바삭하고 얇았다. 특이한 점이라면, 주로 한 사람이 피자 한 판을 먹어서인지 우리나라처럼 피자가 조각으로 잘려 나오지 않는 경우도 많았다. 파스타는 우리나라식과 많이 달랐다. 까르보나라가 대표적인데, 소스가 크림이 아니라 달걀 노른자였다. 그리고 많이 짜서 내 입맛에는 맞지 않았다.

## 5.마무리

지원해 주신 이광근 교수님께 감사드린다. 학회에 참가하는 것만으로도 많은 자극이 되었고 유익했지만, 내 논문을 가지고 발표까지 했더라면 더 좋았을것이란 생각이 학회기간내내 머리에서 맴돌았다. 내년엔 논문을 가지고 다시 찾을수 있기를 바라면서 더 노력하는 계기로 삼아야겠다.