2012 ROSAEC CENTER NEWSLETTER

소프트웨어 무결점 연구센터 뉴스레터

No. 3 March 2012

rosaec.snu.ac.kr

교육과학기술부 / 한국 연구재단 지정 선도연구센터

ROSAEC center

Research On Software Analysis for Error-free Computing

소프트웨어 무결점 연구센터 NRF ERC

No. 3 March 2012

목차

1	센터장 보고	01
2	1단계 발표평가 슬라이드	02
3	대표논문과 이야기	09
4	세미나와 워크샵	19
5	협력연구 이야기	27
6	방문기 및 에세이	30
7	블로그 : ROSAEC minutes	52
8	연구실, 유학생, 행정팀 소식	58
9	참여교수 연락처	68

1

센터장 보고

센터장 이광근
서울대학교 컴퓨터공학부 교수

안녕하십니까.

저희 소프트웨어무결점 연구센터가 선도연구센터(ERC)로 시작된 지 3년 반이 지나서 지난 2월 1단계를 마쳤습니다.

1단계 보고서 작성은 취한 듯한 작업이었습니다. 그동안의 연구결과를 모으면서 자랑스러운 것들만 보였기 때문입니다. 그런 것만 보고 싶은 마음이 앞선 탓일 거라고 봅니다. 돼지의 눈에는 모두가 돼지로 보이고 부처의 눈에는 모두가 부처로 보인다는데 말입니다.

다행히 연구재단에서는 질 중심으로 평가하려는 모습이었습니다. 보고서에는 대표 논문 10개만 포함하도록 요청이 왔습니다. 저희는 애초부터 외부 평가에 연연말고 제대로 된 연구에 집중한다는 모토였는데, 다행이고 감사했습니다.

발표평가는 2011년 12월 14일 100분간 서울 양재동 연구재단에서 있었습니다. 발표장에서 사용한 슬라이드를 추려서 이번 뉴스레터에 싣습니다. 살펴보시고 저희의 1단계 성과 보고로 삼아주시기 바랍니다.

2단계에서는 그동안 계획만이었던 일을 추진할 것 같습니다. 국내에서 로봇과 항공우주 분야에 장착되는 소프트웨어들에 특화된 검증기를 만드는 일입니다.

계속 많은 관심과 격려를 부탁드립니다.

감사합니다.

2 1단계 발표평가 슬라이드

소프트웨어 무결점 연구센터

1단계 성과 보고

이광근

센터장
서울대 교수

12/14/2011 @ 한국연구재단

내용

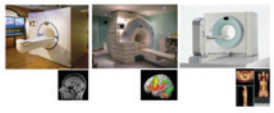
- 센터 개괄
- 1단계 성과
- 2단계 계획
- 마무리

센터의 목표

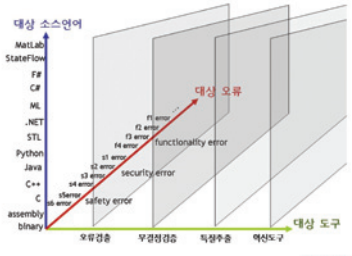
- 원천기술 선도
  정적분석 기술 심화 + 혁신 신기술 연구
- 실용적 도구 개발
  SW 오류 자동 검출/검증기 개발
- 특화된 산업화
  순수 SW, 지능형 로봇, 무인 비행체, 금융 공학에 특화시켜 산업화

센터의 목표 도구

"소프트웨어 MRI" "소프트웨어 fMRI" "소프트웨어 PET"



목표 도구의 공간



세부 초점 사항: 원천 기술

정적 프로그램 분석(static program analysis)[1]

프로그램의 실행 성질을
실행전에 자동으로
안전하게 어림잡는
일반적인 방법

- 엄밀히 예측
- 테스팅의 단점을 보완
- 실용성 확산 (국내외 성과)
  - 우리 예) 산업화한 SPARROW

Sparrow

[1] *static analysis, abstract interpretation, type system, program logic, theorem proving, model checking*

센터 인원 구성

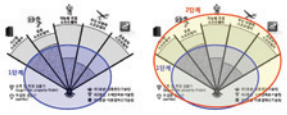
- 연구책임자(총괄 및 세부): 교수 10명
  - 서울대, KAIST, POSTECH, 한양대, 고려대, 경북대
- 교수연구원: 6 ~ 11명
- 박사연구원: 2 ~ 4명, 박사과정: 21 ~ 38명
- 석사연구원: 2 ~ 4명, 석사과정: 30 ~ 45명
- 학부과정: 2 ~ 5명
- 행정팀: 2명

ROSAEC center

이광근 1단계 성과 보고

단계별 목표

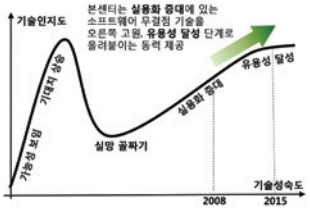
- 1단계: 오류 검출기(bug-finder) 단계
- 2단계: 무결점 검증기(verifier) 단계



ROSAEC center

이광근 1단계 성과 보고

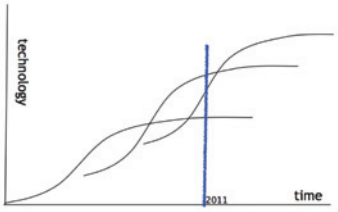
센터기술의 비전(우물안)



ROSAEC center

이광근 1단계 성과 보고

센터기술의 비전(우물밖)



ROSAEC center

이광근 1단계 성과 보고

센터 운영: 지향

- 탁월한가/선두인가
  "excellency, excellency, excellency"
- 존경할만한가/세계적인가
  "우리 주변의 문제" "선도적인 문제"
- 토양을 가꾸는가
  "강의자료, 전공서적/영상, 우리말자료/영상"

ROSAEC center

이광근 1단계 성과 보고

센터 운영: 전략

- "좋은 논문을 발표한다" (top-class)
  - 좋은학회, 좋은이론, 좋은실험
  - 세계적인 연구꺼리 = 우리 주변의 문제
- "좋은 도구를 만든다" (top-class)
  - 정말로 유용한 도구
  - 우리 이웃을 감동시켜야
- "좋은 자료를 보급한다" (top-class)
  - 쉽고 정확하게, 그리고 "몸"으로 파급되게
  - 모국어로 vs 영어로

ROSAEC center

이광근 1단계 성과 보고

센터 운영: 집단연구 시너지

- 내실있는 정기 ROSAEC Workshop
  - 매년 여름-겨울 2회, 3박4일/회
    - 성과발표(milestone talk)
    - 소그룹 난상토론(brainstorming)
    - 기술전수(tutorial)
    - 번개발표(lightning talk)
    - 세분야소개(invited talk)
    - 연구장터(poster)
- 총회샵을 통한 시너지 결실
  - 프로그램 분석과 소프트웨어공학(이광근/김성훈)
  - 알고리즘이론과 소프트웨어공학(정교민/김성훈)
  - 데이터베이스와 소프트웨어공학(황승원/김성훈)
  - 소프트웨어보안과 컴파일러(백윤흥/강병훈) 등

ROSAEC center

이광근 1단계 성과 보고

1단계 성과

- 성과 개괄
- 연구 성과
- 개발 성과
- 토양 성과

ROSAEC center

이광근 1단계 성과 보고

성과 개괄 (1/2)

확실한 질 중심의 탁월성 달성.

연구(논문) + 개발(산업화) + 토양(저변양성)
모든면에서.

ROSAEC center

이광근 1단계 성과 보고

성과 개괄 (2/2)

- 연구: 질 중심 논문 성과
  - 일급 학술지에 발표
  - 세계최고 학회에 발표
  - 선두를 다투는 그룹과의 활발한 교류
- 개발: 일급 산학 선순환 구축
  - 고부가 SW산업 발아
  - 신념: "우리 주변의 문제가 세계적인 연구를 드라이브"한다
  - 확인: 고부가 SW 산업화 = 세계 선도 연구
- 인력양성: 소수정예의 인력배출
  - 이론과 산업화 모두에 탁월한 인재배출
  - 미래에 큰 파급효과 기대

ROSAEC center

이광근 1단계 성과 보고

연구성과: 질중심 탁월성

- 영향력: SCI 논문 ≪ 프리미어급 학술회의 논문
- 각 총괄 모두에서 다수의 일급 저널 논문뿐 아니라
- 해당분야 "최고" 학술회의에 논문 발표
- 국내 컴퓨터분야에서 매우 드문 성과

(2008.9 – 2011.12)

프로그래밍언어 분야	POPL	1편	총괄 1팀
	PLDI	1편	총괄 1팀
	OOPSLA	2편	총괄 1팀, 총괄 3팀
	ICFP	2편	총괄 1팀, 총괄 3팀
소프트웨어공학 분야	ICSE	1편	총괄 1팀
	CAV	1편	총괄 2팀
	TACAS	1편	총괄 1팀
데이터베이스 분야	VLDB	2편	총괄 3팀
	SIGMOD	2편	총괄 3팀
인공지능 분야	AAAI	2편	총괄 3팀

ROSAEC center

이광근 1단계 성과 보고

학술회의 대표논문 10편

POPL 2011 Static Analysis for Multi-Staged Programs via Unstaging Translation
PLDI 2011 Verification of Semantic Commutativity Conditions and Inverse Operations on Linked Data Structures
OOPSLA 2011 A Syntactic Type System for Recursive Modules

CAV 2011 Program Analysis for Overlaid Data Structures
ICSE 2011 MeCC: Memory Comparison-Based Clone Detector
TACAS 2011 Predicate Generation for Learning-Based Quantifier-Free Loop Invariant Inference

VLDB 2011 QSkycube: Efficient Skycube Computation Using Point-Based Space Partitioning
SIGMOD 2011 A New Approach for Processing Ranked Subsequence Matching Based on Ranked Union
AAAI 2010 CosTriage: A Cost-Aware Triage Algorithm for Bug Reporting Systems
AAAI 2010 Coalitional Structure Generation in Skill Games

ROSAEC center

이광근 1단계 성과 보고

저널 논문성과: 질중심 탁월성

논문 발표된 일급 저널들 (2008.9 – 2011.12)

- *ACM Transactions*
- *IEEE Transactions*
- *Annals of Pure and Applied Logic*
- *Acta Informatica*
- *Information and Computation*
- *The VLDB Journal*
- *Journal of Combinatorial Theory*
- *Journal of Functional Programming*
- *International Journal of Robotics Research*

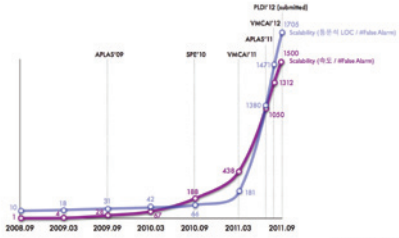
산학협력: 선순환. 산업 ⟸ 센터

산업화한 오류 자동검출 엔진 매출:

- (주)파수닷컴. SPARROW: 매출 ∼1M USD/2010
  (오류검출기 엔진)
- (주)지티원. CHANGEMINER: 매출 ∼2M USD/2010
  (문자열분석 엔진)

산학협력: 선순환. 산업 ⟹ 센터

시장의 문제로부터 연구성과로: SPARROW의 성능향상 과정



관학협력및 공공서비스: 기술 보급

- 소프트웨어 오류 자동검출 온라인 서비스
  - SW Clinic(2011.03): 센터[2]
  - Code Doctor(2010.12): 정보통신산업진흥원 SW공학센터[3]
- 정적분석기술 기반 SW저작권 확인 및 검증 서비스
  - CodeJam(2010.08): 문체부 한국저작권위원회[4]. 2010년 IT혁신상품
- 정부 소프트웨어 보안취약점 검출 도구
  - SW Security Scanner(2009): 행정안전부. 행안부장관표창

[2] rosaec.snu.ac.kr/clinic
[3] www.software.kr/consulting/code_01.jsp
[4] www.codejam.or.kr/codejam

인력양성 (1/2)

의미심장한 진로

OXFORD vs Sparrow

- Oxford대 박사후연구원 vs 산업화 책임자
  - 세계적인 연구성과 ⇒ Oxford대 박사후연구원 제의
  - 산업화 자신감 ⇒ SPARROW사업부 책임 선택
  - 정영범박사

인력양성 (2/2)

MIT Carnegie Mellon STANFORD UNIVERSITY Berkeley

국내 Computer Science분야에서 전례없던 진출. 미국 CS top 대학만으로 박사과정 유학:

- MIT, CMU, Stanford, UC Berkeley의 CS 박사과정 유학
- 모두 센터연구(석사과정)를 통해 주목받은 성과
- 김덕환(2009∼, MIT CSAIL), 공순호(2010∼, CMU CS), 신재호(2011∼, Stanford CS), 최원태(2011∼, Berkeley CS)

국제협력: 내실 + 선두그룹 + 세부집중

국제적 프리미어 리그와 나란히
견실한 풀뿌리 연구 넷트웍
겉치레 지양

34 inbound 방문연구, 27 outbound 방문연구, 45 공저논문

국제협력: 선두를 다투는 그룹과 쌍방향

54 ROSAEC Seminars, ≥13 seminars abroad

MIT
CMU
Oxford U.
Imperial College, London
Max Planck Institute
Aachen U.
Academia Sinica
Microsoft Research Cambridge
Bell Labs, Murray Hill

UC Berkeley
U. of Cambridge
U. of London
École Normale Supérieure, Paris
Kansas State U.
HKUST
Tsinghua U.
Microsoft Research Asia
Oracle Labs

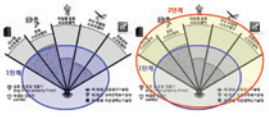
저변양성

- SW검증의 3세대 기술(theorem proving) 준비를 위한
  - CUK: Coq Users @ Korea 결성 및 활동
- 타분야와 융합을 위한 여름학교/겨울학교 개최
  - 소셜네트웍크 분석기법 여름학교(20011.08) 개최
  - 알고리즘과 조합수학 겨울학교(2010.01) 개최
- SW 원천기술에 대한 청소년의 관심과 포부형성을 위한
  - CS4HS(Computer Science for High School) 웍샵 매년 2월 개최 후원. 고등학교 교사 대상.
  - 서울대 핵심교양과목 [컴퓨터가 여는 세상(Computational Civilization)] 개발, EBS 다큐 콘텐츠제공
  - 초중등 대상 프로그래밍 교재 및 실습환경 개발: 바심

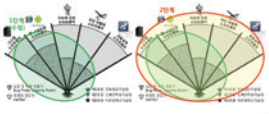
2단계 계획
&
결론

단계별 목표 수정보완

당초



변경



저희의 비전

- 중요한 문제
- 최고의 팀
- 높은 가능성
- 깊은 우물
- 광범위한 효과
- 고부가 SW산업 창출

소프트웨어 무결점 연구센터

저희는 저희의 연구 우위를 더욱 공고히 하고
한편으론 우리 산업과 튼튼히 연계시켜주는,
국부창출의 작지만 알찬 엔진 역할을 하는,
연구센터를 만들고 있습니다.

감사합니다.

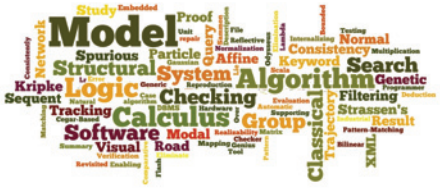
APPENDIX

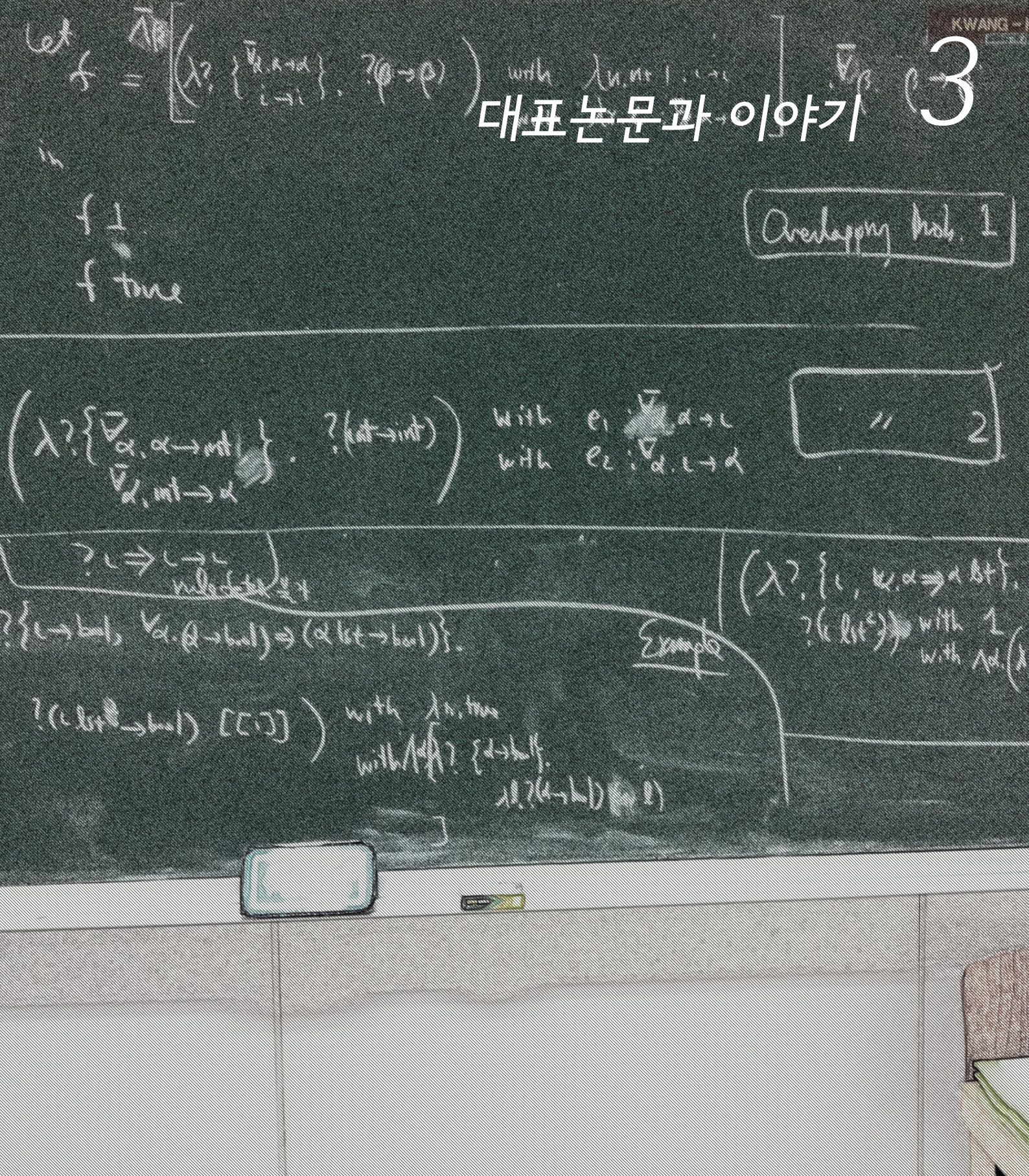
숫자 성과

- 논문
  - 국제 학술회의 논문집: 89편
  - SCI논문지: 35편 (IF총합: 53.92, Citation총합: 164)
- 인력양성
  - 박사 13명
  - 석사 47명
- 특허
  - 출원: 국내 19, 국외 4
  - 등록: 국내 10, 국외 1

학술회의 논문 키워드 ~ 센터목표

저널 논문 키워드 ~ 센터목표




대표논문과 이야기 3
Overlapping Inst. 1
// 2
Example

Design and Implementation of Sparse Global Analyses for C-like Languages

Hakjoo Oh, Kihong Heo, Wonchan Lee, Woosuk Lee, and Kwangkeun Yi

PLDI 2012: The 33rd ACM SIGPLAN Conference on Programing Language Design and Implementation

마음이 일었었다. 두어개 8000미터 봉우리를 오르고 나니 다른 8000미터급 봉우리들은 어떨까, 라는.

다보탑 꼭지에 올라 본 난장이라면 그 옆 석가탑에도 욕심을 내는 법인가. 프로그래밍언어 분야에는 양대 봉우리가 있다. 이론의 POPL과 실제의 PLDI. POPL에는 국내연구로 두 번 등정해 보았다. 옆에 선 PLDI는?

당장 유치한 꺼리리라 생각했다. 이솝우화 여우의 심리 그대로였다. 높은 넝쿨에 탐스럽게 매달린 포도, 딸 수 없는 능력을 위로하는 심리적인 면역시스템. 따먹어 봤자 시고 맛없을 거야.

유치한 마음 유치하게 접고 대신 우리는 원래 목표에 집중했다. 정확히(precise) 실행상황을 모두 포섭하면서(sound) 초대형 프로그램을(scalable) 한꺼번에 분석하기(global analysis). 누구도 달성 못 한 이 목표를 포기하지 말자, 이런 연구를 위해서 우리만큼 제대로 된 워크벤치(Sparrow system)를 갖춘 연구그룹은 전세계에서 드물다, 준비가 무르익었다. 독려하고 자신감을 심고 기회임을 상기시켰다. 슬로건은 "100만라인 통째분석."

이 여정에서 오학주박사는 묵묵히 끈질긴 걸음으로 맨 앞에서 루트를 개척했다. 4년간 집중한 박사과정 내내였다. 그 루트의 궤적은 APLAS'09, SPE'10, VMCAI'11, APLAS'11, VMCAI'12의 논문들이었다. 상용화된 분석기의 기본 엔진보다 무려 1500배로 분석성능을 급격하게 증가시키는 성과들을 쏘아올렸던 궤적. 놀랍고 흥분되는 스토리를 만들어갔다.

정상에 오른 날은 2011년 9월 23일 금요일이었다. 오학주군과 팀멤버들이 미팅에서 담담히 내밀었다. 100만라인 C프로그램 통째분석을 달성한 성능결과였다.

갈고닦은 아이디어는 담백했고 강했다: "필요한 부분만, 필요한 순간에." 프로그램 텍스트에 쓰인 문장들을 무턱대고 분석하지 않고, 그 실행의미 속에 숨어있던 고속통로를 낱낱이 발굴해서 이용한 것이다. A 문장 다음에 B 문장이 있지만, A 문장의 의미가 B 문장과 관계없다면 두 문장의 분석은 순서가 없어도 된다(temporal localization). 그리고 한 문장을 분석하는 데는 그 문장의 의미에 필요한 기계상태의 조각만 있어도 된다(spatial localization).

일부분 비슷한 시도(sparse analysis)가 여럿 있었지만 모두 초보적이었다. 우리는 단단한 최초의 성능결과뿐 아니라, 일반적인 이론까지 제시하였다. 어떤 수준의 정적분석이더라도 그 정확도를 떨어뜨리지 않고 고속통로를 이용하는 분석기로 변환하는 이론.

이 연구에서 다시 한 번 확인한 것은 튼튼한 등산장비(이론)의 힘이다. 과거에 등정했던 팀들이 사용한 장비들, 그 생각의 틀은 약했다. 허술한 카우보이식 정적분석이론 말고 가장 일반적이고 튼튼한 분석이론으로 채비를 하고 나섰다. 그 덕택에 중간 중간 우리의 방법들을 일반화시켜 이해하고 정비할 수 있었다. 리뷰의 한 토막이 즐겁다:

"An important strength of the paper is that the theoretical result is very general. It could be applied to many other analyses. PLDI papers have been accepted that were simply instances of this framework. The result should be highly influential on future work in sparse analysis."

-이광근

이 논문은 박사과정 기간동안 100만 라인 프로그램 분석에 도전했던 연구의 최종 결과로서 순수 국내 연구로는 최초로 PLDI(Programming Language Design and Implementation)에 채택된 논문이다. PLDI는 POPL과 더불어 프로그래밍 언어 분야 최고의 학회이다.

스패로우를 개발하던 때(2006~7년)에는 100만라인 분석은 상상하기 힘든 일이었다. 당시에는 전체분석으로 1만라인 정도 분석하는 데에도 비용이 상당했고 3만라인 달성이 목표였기 때문이다.

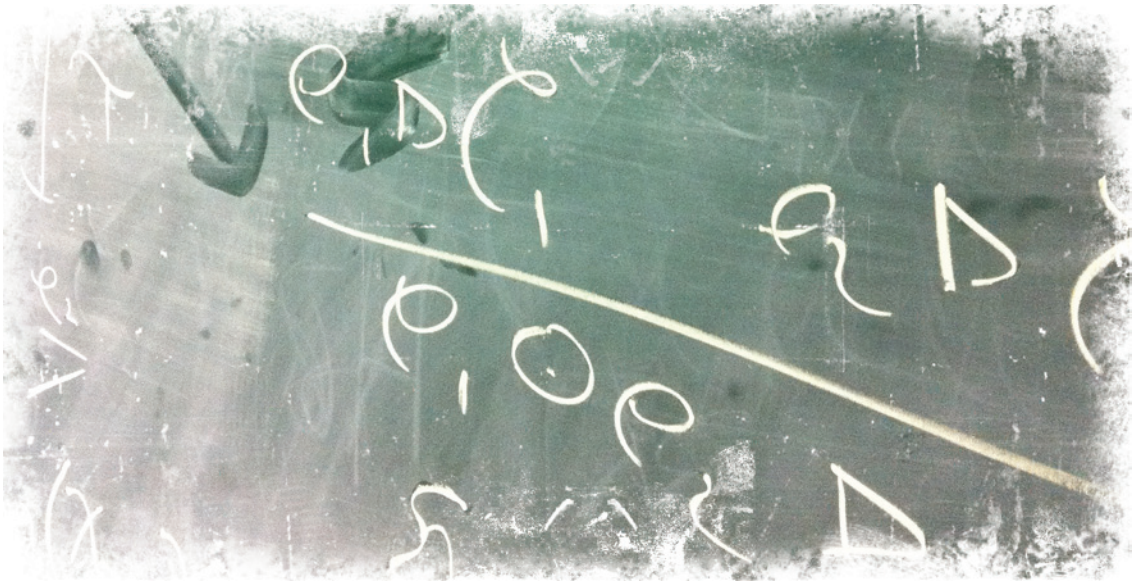
문제 해결의 실마리는 스파스 분석(sparse analysis), 즉 필요한 프로그램 지점을 필요한 메모리 영역만 가지고 분석하는 기법을 적용하면서 얻어졌다. 2009년 가을에 Lucas Brutschy가 인턴을 왔고 함께 초기 아이디어를 구현했다. 하지만 Lucas 버전은 몇 가지 문제가 있었는데, 어떻게 해야 올바른 구현이 되는지 확신할 수 없었고 아직 큰 프로그램을 분석하지 못했다. Lucas가 돌아간 후 윤석이형, 기홍이와 함께 팀이되어 1년넘게 구현 안정화 작업을 했고, 그러던 중에 기존의 연구에서는 발견하지 못했던 올바른 스파스 분석을 설계하기 위한 일반적인 이론적 기반을 찾아냈다. 2011년 여름이 되니 이론적 기반과 함께 구현체가 준비되었다.

내용에는 자신이 있었지만, 늘 그렇듯 논문마감에 맞춰 쓰기에는 시간이 너무 부족했다. 이때 원찬이와 우석이가 합류해서 깔끔한 이론과 함께 더욱 튼튼한 실험결과를 얻었다. 하지만 특히 논문의 전체적인 방향을 잡기가 쉽지 않아서, 1달을 넘게 뚜렷한 성과없이 시간을 보냈고, 제대로 쓰기 시작한 것은 논문 마감 두 주 전이었다. 모든 내용을 가까스로 완성하고 나니 제출이 삼십분 남아 있는 상황이 되었다.

사실 제출하고 나서는 논문이 채택된다는 것을 생각못했다. 왜냐하면, 제출 후 바로 이론상의 오류가 발견되었고, 철자, 문법 등 틀린 것들이 허다했기 때문이다. 전혀 기대 하지 않고 있었는데, rebuttal 기간에 드래프트 리뷰가 왔는데 생각보다 호의적이었다. 스파스 분석의 이론적인 틀과 좋은 실험결과 모두를 보였다는데 평가가 좋았다. 하지만 역시 논문의 구성 및 오류 등에 대해서 의문을 제시하면서 세 명의 리뷰어가 모두 B (weak accept)를 주었다. 의문들에 대한 반박문을 작성해서 보냈다. 주로 논문내용보다는 실험환경이나 구현의 올바름에 대한 의문이 많아서 구현의 신뢰성에 대해서 충분히 설명하였다. 그로부터 1달 후에 최종 채택 통보가 왔다. 내심 기대는 했지만, 채택되었다는 얘기가 믿기 어려웠다. 우리의 반박문이 효과가 있어서 세 명중 두명이 A (strong accept)로 바꾸었고, 새로운 리뷰가 하나 더 와서 AABB로 채택되었다.

무엇보다 오랜 기간 동안, 여러 사람들과 함께 연구한 내용이 좋은 결과로까지 이어져서 기뻤다. 2008년 PLDI에 논문을 내고 CCCD를 받고 처참하게 떨어진 기억이 있다. 또한 당시 PLDI에 참가했는데 논문들이 너무 훌륭하고 발표도 잘해서 은연중에 이곳은 나와 인연이 없을 것이라고 생각해 왔던 것 같다. 그랬기 때문에, 교수님께서 이번 PLDI에 내보자고 하셨을 때에도 사실 자신이 없었다. 하지만, 논문이 채택되었고 우리나라에서, 우리만의 연구로 세계적으로 인정받았다는 사실이 뿌듯했다.

-오학주



Sound Non-statistical Clustering of Static Analysis Alarms

Woosuk Lee, Wonchan Lee, and Kwangkeun Yi
VMCAI 2012: The 13th International Conference on Verification, Model Checking, and Abstract Interpretation

정적 분석기 사용자 편의성을 증대시키는 일반적인 방법에 관한 논문입니다. 제 첫번째 학회논문이고, 실제 산업 현장에서 문제를 찾은 연구입니다. 문제는 제가 인턴을 하면서 발견했습니다. 저는 여름방학에 연구실의 분석기 Sparrow를 상용화한 파수닷컴에서 인턴을 했었습니다. 파수닷컴은 분석기를 다른 소프트웨어 회사에 팔면서 알람 분류(분석 결과 중 실제 오류들만 찾는 일) 서비스도 제공하고 있었습니다.

저는 인턴을 하면서 종종 그 일을 도왔는데 재미도 없고 힘이 들었습니다. 왜냐하면 알람이 일반적으로 많이 나오고, 그 중에 허위 경보도 섞여 있는 가운데 진짜 오류를 찾는 것은 귀찮은 노동의 연속이었기 때문입니다. 알람을 빨리 검사하면 하루 종일 100개 정도를 봤는데, 분석 결과 몇백개의 알람이 나오는 일도 허다했습니다.

그래서 이 일을 어떻게 하면 편하게 할까 고민하다가, 같은 원인을 공유하는 비슷한 알람들이 많다는 걸 알았습니다. 이 중 안전하게 일부만 보여주는 일을 자동화시킬 수 있겠다 싶어서 당시 연구실에 계시던 김유일 박사님과 함께 이 일을 시작했습니다.

처음엔 간단한 일 같았지만 생각보다 해결해야 될 문제들이 많았습니다. 초반에는 알고리즘을 어떻게 설계해야 하는지 알기 위해서, 나중에는 구현이 제대로 되었나 확인하고 예제들을 찾기 위해 알람을 수천개는 본 것 같습니다. 일을 진행하는 동안 시간은 거의 1년이 지나면서 파트너는 김유일 박사님에서 이원찬 연구원으로 바뀌었습니다.

내용을 정리해서 VMCAI에 내게되었고, 감사하게도 이것이 제 첫번째 학회 논문이 되게 되었습니다.

-이우석

The Implicit Calculus: A New Foundation for Generic Programming

Bruno Oliverira, Tom Schrijvers, Wontae Choi, Wonchan Lee, and Kwangkeun Yi
PLDI 2012: The 33rd ACM SIGPLAN Conference on Programing Language Design and Implementation

"Implicit programming"? 센터 연구교수로 있던 브르노가 나와의 미팅에서 꺼내 놓았다. 이 프로그래밍 방식이 여러 언어에 도입되고 있지만 너무 중구난방이다, 잘 정리해서 그게 과연 무엇인지 그리고 그 맹수를 프로그래머가 안전하게 다루게 도와주는 방법(타입시스템)을 탐구해 보고 싶다.

안돼에! 라고 반응하려는데 구으뤠? 로 바뀐 것은 몇 번의 미팅 이후다. 연구를 위한 연구는 안된다, 그 연구동기가 실질적이면 좋겠다고 채근했다. 들어보니, 그런 프로그래밍 방식은 Haskell에서만 쓰이는 것이 아니고 Scala(Twitter 구현언어)에서도 쓰이고, 심지어는 C++의 거푸집 프로그래밍(template programming)에서 지원하려고 안달인 것이 결국은 "implicit programming"이었다.

암시적 프로그래밍(implicit programming)은 크게 보면 당연한 방향이었다. 프로그래밍은 점점 상위의 레벨로 오른다. 프로그래밍에서 지루한 코딩의 과정을 줄이려는 노력이 프로그래밍의 발전 단계가 아니던가. 프로그래밍 과정에서 컴퓨터가 자동으로 해 주는 일이 점점 많아지고 있다. 암시적 프로그래밍은 컴퓨터가 자동으로 프로그램 코드를 메꿔준다. 일종의 "computer-aided programming"이다.

몇 번의 미팅끝에 암시적 프로그래밍의 핵심을 정리하게 되었다: 암시적 프로그래밍은 필요한 타입의 데이터를 만드는 코드를 자동 생성하기. 이러한 자동 코딩 과정은 단순한 논리과정으로 가능한 정도만으로 한정되었다.

그러고는 본 게임이 시작되었다. 있을 수 있는 문제와 그 가능성의 끝을 파악하는 작업. 암시적 프로그래밍의 모델을 가지고 놀아 보자. "Implicit Calculus"를 정의해 갔다. 팀원들은 미팅중에 여러 아이디어들이 충돌했다. 그 언어모델의 문법구조(syntax)부터 서로 의견이 달랐다. 그 의미구조(semantics)를 정의하는 데도 이런 저런 의견이 부닥쳤다. 브르노는 전통적이지 않은 방식으로 접근하는 데 익숙했다. 학생들과 나는 전통적인 방식으로 접근하려고 했다. 브르노와 학생들이 다른 의견으로 분분할 때 어느 한 쪽의 손을 들어줘야 하는 난감한 시간이 몇 번 있었다. 나는 전통적인 방향의 것에 기울었다. 내용도 그렇게 채워지고 있었다. 중간에 브르노의 제안으로 팀 슈라이버 교수가 벨기에에서 합류했다. 이미 이쪽으로 많이 탐구한 친구였다.

우선 POPL논문 마감에 맞추어 박차를 가했지만 논문은 낙방했다. 전통적인 방식으로 구성한 논문 버전. 지치고 허탈해서 내버려두려 했는데, 브르노가 PLDI에 내자고 했다. 자신이 주장하는 방식으로 각색하면 될 것 같다고, 그 방식을 선호하는 연구진이 프로그램위원회에 많이 있다고. 그래보라고 맡겼다.

보란듯이 이번에는 PLDI에 게재승인되었다. 이 논문을 "champion"하겠다는 리뷰어가 4명중 3명이었다. 올해 PLDI에 우리나라 논문은 최초다. 그것도 이 논문과 함께 두 편이 한꺼번에. 경사였다.

-이광근

2010년 9월이었다. 브루노(Bruno C.d.S. Oliveira)가 이번에는 암시적 프로그래밍(Implicit Programming) 이라는 카드를 들고 나왔다. 프로그램 코드에서 자명한 부분을 생략하면 프로그래밍 시스템이 타입을 토대로 유추하여 구멍을 자동으로 채워 주는 기술을 암시적 프로그래밍이라고 부른다. 불필요한 코드를 줄이는데 도움이 되기 때문에 많은 언어가 (예를 들어, C++, Scala, Haskell) 암시적 프로그래밍을 지원한다. 그러나 각자 자신들의 구현에만 관심이 있을 뿐 핵심을 찾고자 하는 시도는 미비했다.

이광근 교수님이 브루노와 톰(Tom Schrjivers, University of Gent)과 이원찬 연구원으로 구성된 첫 번째 팀을 꾸렸다. 나는 한달 뒤에 팀에 합류했다. 목표는 암시적 프로그래밍의 핵심을 정확하게 드러내고 Haskell과 Scala의 핵심적인 예제를 포섭하는 작은 calculus를 만드는 것이었다.

첫 단계로 브루노가 수집한 예제에서 Scala와 Haskell의 껍질을 벗겨 냈다. 언어의 형식적인 껍질은 예상보다 두꺼웠다. 작업 결과물에 우리에게 익숙한 언어의 껍질이 덧입혀지기도 했다. 약 4개월간 갈아낸 끝에, 우리의 implicit calculus는 네 가지 부품만을 가지게 되었다. Haskell과 Scala의 모든 예제가 이 네 가지 부품과 lambda calclus로 간명하게 표현되었다.

다음 단계로 강력하면서도 결정가능한(decidable) 타입시스템을 찾아 나섰다. implicit calculus의 타입시스템은 Haskell과 Scala의 예제를 포섭할 만큼 강력해야 한다. 그렇다고 너무 강력해지면 타입검사(Type Checking)가 결정불가능(undecidable)해진다. 여기에 어려움이 있다. 작은 수정에도 타입시스템은 결정불가능해졌다가 다시 쓸모없어지기를 반복했다. 양 극단을 수차례 넘어다니며 점차 중간을 찾아갔다. 작업 속도를 높이기 위해 톰이 서울대를 찾아와 열흘 가량 머물기도 했다. POPL 마감을 한 달 앞두고 마침내 타입시스템이 자리를 잡았다.

마지막 난관은 논문을 쓰면서 찾아왔다. 우리는 담고 싶은 것은 너무 많았다. 브루노와 톰은 System F로의 변환을 통해 실행의미를 설명하고자 했다. Haskell 커뮤니티에서는 익숙한 방법이라고 했다. 이광근 교수님과 나, 이원찬 연구원은 과정을 드러내는 실행의미(Dynamic Semantics)를 갖추길 바랐다. 변환으로 실행의미를 정의하는 것이 익숙치 않을 수 있는 우리 분야 사람들에게까지도 잘 전달됐으면 했다. 이와 무관하게 Haskell과 Scala로 작성된 프로그램을 implicit calculus로 변환하는 방법도 보여야 했다. 욕심을 버리지 못하고 모든 것을 담은 첫 논문은 빡빡하고 거칠었다. 결국 POPL에는 채택되지 않았다. 리뷰중 아이디어를 깊이 이해한 듯 보이는 사람은 한명 뿐이었다. 설명이 직관적이지 않다는 증거였다. 아이디어 자체를 평가 받자는 취지로 내용을 줄이고 쉽게 쓴 버전을 PLDI에 제출하였다. 피드백을 취합해 논문을 재정비할 생각이었다. 놀랍게도 이번에는 굉장히 좋은 리뷰와 함께 채택되었다. 리뷰중

"The paper is well-written, if rather dense"

라는 구절에서 부끄러워졌다. 내용을 줄이고 풀어쓴 논문이 빡빡하게 느껴졌으면 첫 번째 논문은 오죽했을까. 논문의 기본은 커뮤니케이션이다. 처음부터 읽는 사람이 받아들이기 좋은 만큼 담아야 하지 않았을까.

다음 리뷰가 좋았다. 최초의 야심찬 목표를 향해 한 걸음 나아간 것은 확실하다.

"The paper brings clarity to a murky area of language design"

-최원태

GMeta: A Generic Formal Metatheory Framework for First-Order Representations

Gyesik Lee, Bruno C. d. S. Oliveira, Sungkeun Cho, and Kwangkeun Yi
ESOP 2012: The European Symposium on Programming

2009년 무더운 여름이 지나갈 무렵 Bruno가 ROPAS 그룹에 합류했다. 처음에 서로 소개를 하면서 generic programming과 Haskell이 주 전공이라 말하기에 논리와 Coq을 주로 다루는 나와는 좀 거리가 있다 싶었는데 어느 날 Bruno가 generic programming과 Coq을 접목시킬 수 있는 아이디어가 있는데 한 번 들어보겠냐고 물었다.

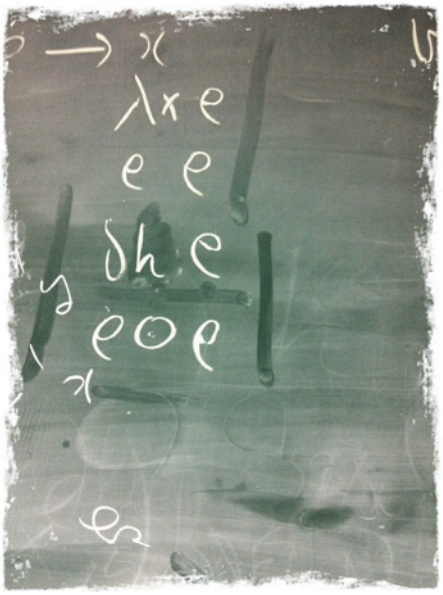
GMeta의 시작이었다. 이후 우리들은 처음의 아이디어를 구현하려고 하면서 서로 배움을 주고 받았다. 나는 generic programming을, Bruno는 Coq을. 잠시 후 프로젝트에 합류한 성근이는 generic programming과 Coq 모두를 익혀야 해서 보다 많은 에너지를 필요로 했다.

처음엔 Coq에서 어떻게 구현할 것인가에 집중하면서 많은 테스트를 할 수 밖에 없었다. 시간과 노력이 조금씩 쌓이면서 원하는 형태의 모습을 갖출 수 있었으며, variable binding에 관한 나의 경험을 바탕으로 해서 GMeta 라이브러리를 구현하기 시작하였다.

논문을 학회에 처음 제출하기까지 약 10개월여 준비하면서 마지막 석 달은 정말로 정신없이 구현에만 몰두하였다. Bruno는 이론과 논문 작성을 담당하고 Coq 구현은 나와 성근이가 전담하는 식으로 역할을 분담하였는데 성근이의 도움이 있었기에 기한에 맞추어 구현을 일단락 지을 수 있었다.

GMeta 이야기를 하다보니 Bruno, 성근이와 토론을 하며 보낸 많은 시간들이 소록소록 떠오른다. 내겐 아주 소중한 경험이다.

-이계식



석사 1년차에 여러 박사님들과 함께 연구를 할 수 있었다는 것은 아주 큰 행운이었습니다. 이 기간에 많은 것을 배울 수 있었습니다. 먼저 증명 보조기인 Coq에 대해 배우고, 데이터 타입이 정해지기 전에, 미래에 정해질 데이터 타입의 값에 대해 적용 가능한 함수를 정의하는 DGP(Datatype Generic Programming)에 대해 배웠습니다.

'내가 연구에 도움이 될 수 있을까?'

여러 박사님들과의 연구는 저에게 작지 않은 부담으로 다가왔습니다. 먼저 연구에 참여하기 위해서는 Coq 프로그래밍이나 DGP에 대한 개념을 갖추어야 했습니다. 아직 완벽히 작성되지는 않은 논문과 지금까지 이계식 교수님께서 만드신 GMeta 라이브러리를 참고하며 라이브러리 확장을 시작했고, 그렇게 부딪히며 연구에 대한 이해가 깊어짐을 느낄 수 있었습니다.

이 연구는 프로그래밍 언어를 Coq과 같은 증명보조기에서 표현/증명할 때 필요한 성가신 infrastructure들의 자동생성을 목표로 하고 있습니다. 손으로 증명할 때에 이런 infrastructure들을 가정하고 넘어가는 경우가 많은데, 증명 보조기를 이용할 때엔 이것들이 모두 증명되어야 하는 부담이 있습니다. GMeta를 이용하면 이런 부담을 줄여줌으로써 사용자는 좀 더 중요한 부분의 증명에 집중할 수 있게 됩니다.

이 연구가 accept되기까지 2년이라는 짧지 않은 시간이 걸렸습니다. 여러 번의 reject이 있었습니다. POPL, ICFP, CPP. 분명 문제도 명확하고, 해결법도 아름다우며, 결과도 나쁘지 않았지만, 좋은 평가를 받지는 못했습니다. 논문을 새로 제출할 때마다 어떻게 하면 연구의 내용을 잘 전달할 수 있을지 고민하고 수정하였습니다. 결국 2년만에 ISOP의 리뷰어들로부터 좋은 리뷰를 받을 수 있었습니다.

리뷰 중

"Not only does it provide a new application of datatype-generic programming, but it solves a real problem in mechanized metatheory."

"For a reader like me who has suffered from exactly the problem of tedious boilerplate infrastructure for programming-language proof developments that is addressed here, this is very welcome work."

이번 경험을 통해 우리 주변의 문제를 해결하는 것이 좋은 연구라는 사실을 다시 한 번 확인할 수 있었습니다.

-조성근

Type-Checking Modular Multiple Dispatch with Parametric Polymorphism and Multiple Inheritance

Eric Allen, Justin Hilburn, Scott Kilpatrick, Victor Luchangco, Sukyoung Ryu, David Chase, and Guy L. Steele Jr.
OOPSLA 2011: The Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages and Applications

Fortress 프로그래밍 언어는 다양한 기능을 제공하고 있는데, 여러 가지 기능을 안전하게 지원하기 위해서 타입 검사가 복잡해집니다. 특히 같은 이름을 갖는 여러 함수들을 정의하고 함수 호출 시에 동적인 타입을 이용해 함수를 선택하면서, 동시에 다중 상속을 지원하는 경우에는 모듈별로 타입 검사하는 것이 까다롭다는 것이 잘 알려져있습니다. 현재까지는 같은 이름을 갖는 여러 함수의 인자 타입이 상대적으로 간단한 경우만을 다룰 수 있었는데, 이 논문에서 처음으로 다형 타입을 갖는 경우로 확장하였습니다. 확장된 타입 시스템을 엄밀하게 정의하고 구현하여, 현재 Fortress 컴파일러에서 사용하고 있습니다. 2010년 여름 두 달 동안 Fortress 팀이 있는 보스턴 근교에서 같이 공동연구하고 논문 작업한 기억이 좋은 논문으로 마무리되어 기쁘게 생각합니다.

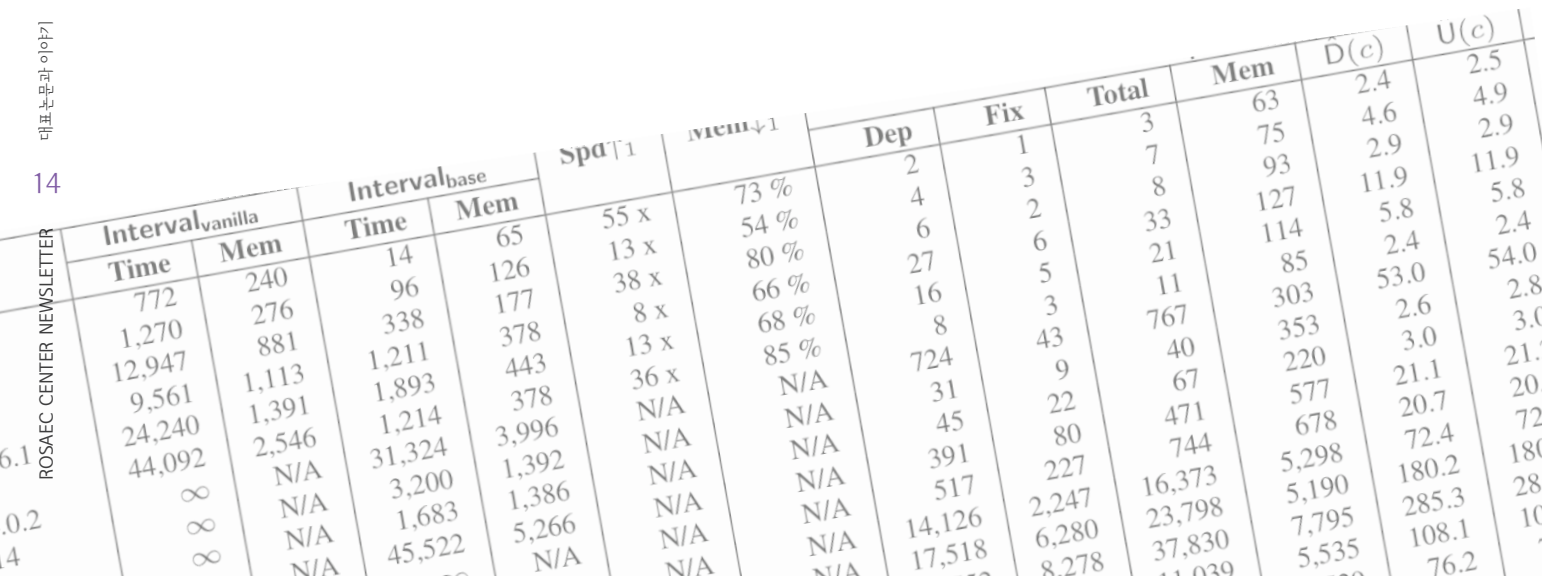
A Syntactic Type System for Recursive Modules

Hyeonseung Im, Keiko Nakata, Jacques Garrigue, and Sungwoo Park
OOPSLA 2011: The Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages and Applications

ML 모듈 시스템은 강력합니다. 중첩 모듈을 이용하여 이름공간을 효과적으로 구성할 수 있으며 추상 데이터 타입 및 signature sealing을 이용하여 미려하고 오묘한 데이터 추상화, 정보 은닉을 지원합니다. Functors를 이용한 모듈 단위의 강력한 코드 재사용 또한 일품입니다. 그러나 재귀 모듈을 완벽하게 지원하지는 못합니다.

보다 완전한 재귀 모듈 시스템을 설계하고자 했습니다. 재귀 모듈 시스템에 관한 연구로 박사 논문을 쓰신 Keiko 박사님과 OCaml 모듈 시스템 개발자이신 Jacques 교수님께서 함께 하셨습니다.

천군만마였습니다. 재귀 모듈 시스템 설계에 있어서 양대 산맥인 double vision problem과 순환타입문제를 해결하고 논문을 작성했습니다. ICFP 2011에 제출하고자 했습니다. 초록을 등록하고 논문을 계속 수정하고 있었는데, 프로그램 위원장이었던 Olivier Danvy 교수님에게서 메일이 왔습니다. Jacques 교수님이 프로그램위원회 위원이기 때문에 논문을 제출할 수 없다는 것이었습니다. 청천벽력과 같은 소리였고 잠시 좌절했으나 OOPSLA에 논문을 제출하기로 했습니다. 다행히 OOPSLA가 타입이론까지 영역을 확장했기 때문에 논문을 제출할 수 있었고 또 논문이 게재 승인될 수 있었습니다.



Program Analysis for Overlaid Data Structures

Oukseh Lee, Hongseok Yang, and Rasmus Petersen
CAV 2011: The International Conference on Computer Aided Verification

영국에 연구년을 가서 처음 맡은 연구이다.

양홍석 박사가 2년 정도 Rasmus Petersen과 작업하였는데도 분석하지 못하는 시스템 코드가 있다고 하였다. 문제는 겹쳐진 자료구조였다. 겹쳐진 자료구조를 분리해서 각각 분석하는 아이디어까지는 좋은데 각각 분석하면서 놓치는 정보가 있어 검증이 되지 않는다는 것이었다.

필요한 정보를 놓치지 않도록 지역 (region) 정보를 유지하고 이용하도록 하였더니 문제없이 검증이 되었다. 때에 맞추어 CAV에 제출하였고 문제없이 게재되었다. 연구 결과가 CAV에 "문제없이" 게재될 수 있었던 큰 이유 중 하나는, 양홍석 박사가 이 결과에 대한 세미나를 여기저기서 하여 내용을 미리 널리 알렸다는 것이다. 이 논문을 심사할 가능성이 높은 사람들은 양홍석 박사의 세미나를 통해 이미 내용을 알고 있었고, 그렇기 때문에 논문의 사소한 문제가 아닌 깊은 심사가 가능했고, 결과적으로 높은 점수를 획득할 수 있었다. 연구 결과의 소통이 중요함을 새삼 느꼈다.

Stability of the Max-Weight Protocol in Adversarial Wireless Networks

Sungsu Lim, Kyomin Jung, and Matthew Andrews
INFOCOM 2012: The 31st Annual IEEE International Conference on Computer Communications

본 연구는 2010년부터 정교민 교수님, Bell Labs, Murray Hill의 Matthew Andrews 연구원과 함께 진행하였습니다. 처음에는 SIGMETRICS 2011에 제출하였고 전체적인 평은 좋았으나 네 명의 reviewer중 한 명이 이 문제는 통신 네트워크보다 전산 이론에 더 가까운 문제로 보인다며 weak reject로 평가하여 최종 채택되지 않았습니다. Review를 감안하여 의미를 더 쉽게 설명하고 복잡한 증명은 technical report 등으로 넘겨서 증명 과정을 요약하는 등 수정을 거쳐서 통신 및 네트워크 분야 최고 학회인 INFOCOM 2012에 채택되었습니다. 관련 연구 결과를 제 6회 ROSAEC 워크샵에서 발표하기도 하였습니다.

Automated Analysis of Industrial Embedded Software

Moonzoo Kim and Yoonho Kim
ATVA 2011: The 9th International Symposium on Automated Technology for Verification and Analysis

이 논문은 ATVA 2011 에 영향력 있는 아시아 연구자로 선정되어 초대받은 논문이다. 처음 초대 메일을 받았을 때 '아 우리 연구가 이렇게 인정받고 있구나' 하는 생각과 함께 '아 우리가 옳바른 길을 걷고 있구나' 하는 생각이 들었다. 더 좋은 연구를 수행해서 더 많은 곳에서 초대받고 인정받는 연구자가 되고 싶다는 생각도 들었다.

논문 내용은 2006년부터 현재까지 진행해 온 사례 연구를 살펴보면서 각 검증 기법들이 어떤 장단점을 갖고 있는지 종합하는 것이었다. 특별히 새로운 내용을 더한 것은 아니지만 어떤 과정을 거쳐 우리가 현재 연구 방향을 설정하였는지가 명확하게 보였고, 그 과정이 사례 연구를 통해 탄탄하게 뒷받침 되고 있음을 다시 한 번 확인할 수 있었다. 우리가 발전시켜나가는 방향이 실제 현장에서 필요한 것이고, 우리 연구 결과가 실제 현장에서 유용하게 활용될 수 있음을 본 논문을 작성하면서 확신하게 되었다.

A Comparative Study of Software Model Checkers as Unit Testing Tools: An Industiral Case Study

Moonzoo Kim, Yoonho Kim, and Hotae Kim
IEEE TRANSACTIONS ON SOFTWARE ENGINEERING

이 논문은 ASE 2008 에 제출했던 모델 체킹 사례 연구를 확장한 논문이다. 기존 논문에서 SAT 문제 해결기에 기반한 모델 체킹 기법을 다루었는데 논문 발표 이후 다른 소프트웨어 모델 체킹 기법, 특히 Counterexample-guided abstraction refinement(CEGAR) with predicate abstraction 과 비교해서 어떤 게 더 쓸만하냐를 물어보는 질문을 종종 받았다. 서로 다른 두 소프트웨어 모델 체킹 기법이 어떤 차이점을 보여줄지 우리도 궁금해졌고, 내친김에 이 둘을 상세하게 비교 분석해보는 연구를 수행하게 되었다.

처음 비교 연구를 시작하기 전에는 내심 CEGAR with predicate abstraction 기법에 대한 기대가 있었다. SAT 기반 모델 체킹 기법이 소스 코드가 커질수록 SAT 문제 해결에 오랜 시간이 걸려 확장성에 한계를 보였는데 이런 한계를 abstraction 을 통해서 해결해 줄 수 있지 않을까 하는 기대였다. 하지만 막상 비교 연구를 수행하다 보니 CEGAR with predicate abstraction 기법이 C 언어의 포인터 구문을 정확하게 분석하지 못해 부정확한 결과를 내 주는 것을 확인할 수 있었다. 특히 임베디드 소프트웨어의 복잡한 포인터 활용을 생각해볼 때, 도저히 실제 활용 가능한 기법이라는 생각이 들지 않았다. 역시 SAT 기만 모델 체킹을 선택한 우리의 판단이 옳았다는 것을 다시 한 번 확인할 수 있었다.

2008년에 논문을 제출하고 심사평을 받았을 때도 느꼈지만 SE 커뮤니티가 실제 적용 사례 연구에 목말라 있다는 것을 이번 논문의 리뷰를 보면서 다시 한 번 느낄 수 있었다. 다음은 리뷰에서 발췌한 내용이다. SE 연구를 발전시켜 나가기 위해서 현재 기술 수준의 가능성과 한계를 평가하는 사례 연구에 목말라있음을 보여주는 리뷰였다.

"In general, I think this is a good TSE paper. It is a close case study, with important industrial software (components at least), using tools that many readers will be very interested to know more about -- including practical limitations and success."

"I enjoyed reading this paper. There is little "technical content" if by that term one means algorithms or theorems, but there is a severe shortage of papers that provide the service that this one does: detailed case studies that explore the possibilities and limitations of different software model checking technologies. Indeed it is difficult to make progress without these papers and so I definitely recommend acceptance."

COSTRIAGE: A Cost-Aware Triage Algorithm for Bug Reporting Systems

Jin-woo Park, Mu-woong Lee, Jinhan Kim, Seung-won Hwang, and Sunghun Kim
AAAI 2011: The Twenty-Fifth AAAI Conference on Artificial Intelligence

이 논문은 새로 생성된 버그레포트들을 "빠르고 정확하게 해결할 수 있는 개발자"에게 알맞게 할당하기 위한 문제를 해결하기 위한 방법을 제시하여, AI 분야의 최고학회 중 하나인 AAAI 2011에 채택되었습니다.

이 논문을 적기 위해서 밤새 연구하던 모습들이 주마등처럼 지나가네요. 지금은 버그레포트 연구와는 다른 연구를 하고 있지만, 이 논문을 적으면서 했던 경험은 저의 연구에 많은 도움이 되고 있습니다.

지도교수님이신 황승원 교수님과 홍콩과기대의 김성훈 교수님의 지도와 공동저자인 무웅이형과 진한이의 도움에 감사드립니다.

A Modal Logic Internalizing Normal Proofs

Sungwoo Park and Hyeonseung Im
Information and Computation

2005년에 지도교수님이신 Pfenning 교수님과 discussion 중에 찾은 문제로 문제를 풀고 논문으로 출판되기까지 6년이나 걸린 인고의 결과물이다. 양상논리를 이용하여 normal proof가 존재함을 judgment에 표현하는 방법에 대한 연구이다.

A New Approach for Processing Ranked Subsequence Matching Based on Ranked Union

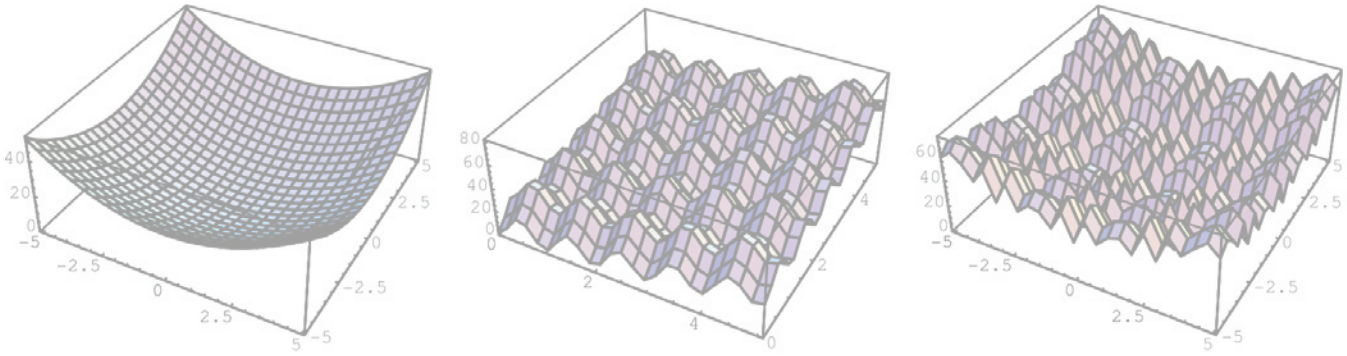
Wook-Shin Han, Jinsoo Lee, Yang-Sae Moon, Seung-won Hwang, and Hwanjo Yu
SIGMOD 2011: The 37th International Conference on Management of Data, ACM SIGMOD

순위를 지원하는 서브시퀀스 매칭은 데이터 시퀀스로부터 주어진 질의 시퀀스와 가장 유사한 서브시퀀스 상위 k개를 찾는 것이다. 기존에 제안된 HLMJ방법은 최소 거리 매칭 윈도우 쌍(minimum distance matching window pair-MDMWP)라는 개념과 전역 우선순위 큐를 사용하여 이 문제를 풀었다. MDMWP의 개념을 사용함으로써 HLMJ방법은 하한 거리값을 사용하여 데이터 시퀀스에 많은 불필요한 접근을 제거할 수 있다. 그러나 우리는 HMLJ방법이 중요한 유형의 질의에 대해서 심각한 성능의 오버헤드를 야기함을 알아냈다. 이 논문에서, 우리는 순위를 지원하는 서브시퀀스 매칭을 순위를 지원하는 합집합 질의로 봄으로써 이 문제를 풀기위한 새로운 체계적인 프레임워크를 제안한다. 특히, 우리는 matching subsequence equivalence class(MSEQ)의 개념과 MSEQ-distnace라고 불리는 새로운 lower-bound 를 제안한다. HLMJ방법의 성능 문제를 완전히 제거하기 위해서, 우리는 또한 cost-aware density-based scheduling 기법을 제안하고, 밀도와 우선순위 큐의 비용을 둘다 고려하도록 한다. 많은 실제 데이터 셋을 가지고 한 광범위한 실험의 결과는 제안된 알고리즘이 HLMJ와 adapted PSM보다 뛰어난 성능을 나타냄을 보였고, 최신의 연구인 non-monotonic distnace 함수를 지원하는 색인 기반 병합 알고리즘보다 수백배에서 수천배까지 더 좋았다.

iGraph in Action: Performance Analysis of Disk-Based Graph Indexing Techniques

Wook-Shin Han, Minh-Duc Pham, Yang-Sae Moon, Seung-won Hwang, and Hwanjo Yu
SIGMOD 2011: The 37th International Conference on Management of Data, ACM SIGMOD

그래프는 화학 구조식이나 단백질 상호작용, 그림, 그리고 프로그램 종속성 등과 같이 복잡한 구조를 모델링하는 강력한 방법을 제공한다. 그래프 색인 기술에서 실험하는 이전의 관행은 새로 제안된 기술의 저자가 자신의 것을 구현한 code를 기반으로 기존의 index를 구현하지 않고, 원 저자의 실행 파일을 사용하여 실험하고, 시간판을 보고해왔다. 우리는 이러한 관행이 몇몇 문제들을 낳는다는 것을 알았다. 이 문제를 해결하기 위해서 우리는 모든 대표적인 그래프 색인 기술을 iGraph라 불리는 하나의 공통 프레임 워크에 구현했다. 이 데모에서 우리는 iGraph를 보이고, 몇몇 실제 데이터셋과 그들의 workload를 사용하여 그림으로 나타낸다. workload로부터 선택된 질의들에 우리는 철저한 성능 분석을 포함하여 몇몇 고유의 특성들을 보인다.



Memory Access Optimization in Compilation for Coarse-Grained Reconfigurable Architectures

Yongjoo Kim, Jongeun Lee, Aviral Shrivastava, and Yunheung Paek
ACM Transactions on Design Automation of Electronic Systems

이 논문은 2010년에 게재한 LCTES 학회 논문을 확장한 논문입니다. CGRA가 포함된 아키텍쳐는 메인 프로세서와 공유하는 메인메모리 이외에도 CGRA가 독자적으로 사용하는 로컬 메모리를 가지고 있습니다. LCTES 논문에서는 이 로컬 메모리에 데이터를 로드하였을 때 최대한 많이 재사용을 하는 것을 목표로 하여 논문을 작성하였습니다. 이 논문에서는 로컬 메모리에서 CGRA의 Processing Element (PE) 로 로드한 경우 한번 로드했을 때 최대한 많이 재사용을 하는, 즉 데이터를 재사용을 하는 granularity 을 꿔서 CGRA의 데이터 재사용 기법을 연구하였습니다.

결과는 대성공이었습니다. 기존의 로컬 메모리에서 데이터를 재사용하는 기법을 통해서 bus traffic을 줄이는 것에, 추가로 로컬 메모리에 access하는 횟수를 줄임으로써 생성되는 코드의 사이즈를 줄이고 성능을 향상 시킬 수 있었습니다. 이 기법을 추가로 적용함으로써 CGRA에서의 데이터 재사용 기법에 대한 연구를 마무리 할 수 있었고 좋은 저널에 게재할 수 있었습니다.

High Throughput Data Mapping for Coarse-Grained Reconfigurable Architectures

Yongjoo Kim, Jongeun Lee, Aviral Shrivastava, Jonghee W. Yoon, Doosan Cho, and Yunheung Paek
IEEE Transactions On Computer-aided Design Of Integrated Circuits And Systems

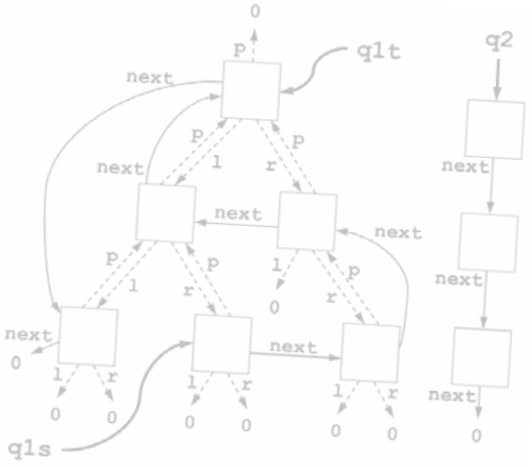
몇 년 전 UNIST에 이종은 교수님을 방문할 일이 있었습니다. 당시 저는 CGRA를 사용한 데이터 관리를 연구하고 있었는데, 수행하고자 하는 코드에 loop-carried dependency가 있을 때의 처리방법에 대해서 골머리를 앓고 있었습니다. 마침 이종은 교수님도 CGRA에 대한 연구를 같이 진행해 본 경험이 있었기 때문에 이 문제에 대해서 의견을 여쭤 보았었습니다. 아무런 기대도 하지 않고 했던 질문이었지만 이종은 교수님께서는 CGRA의 수행방식에 다른 견해를 가지고 계셨기 때문에 활발한 논의를 할 수 있었습니다. 당시 나는 모든것을 소프트웨어 적으로, 즉 컴파일러에서 모든것을 처리하려고 했었습니다. 하지만 이종은 교수님께서는 하드웨어를 약간 수정하는 방법을 제안을 하셨고, 결국 하드웨어에서 약간의 지원만 해 준다면 컴파일러에서 쉽게 처리할 수 있다는 결론에 도달할 수 있었습니다. 이후 일은 일사천리로 진행되어 논문 작업을 잘 마무리할 수 있었습니다.

이번 경험을 통해 연구는 혼자 하는 것이 아니라는 것을 깨달을 수 있었습니다. 혼자서도 해법을 제시할 수 있겠지만, 다른 사람과 논의를 통해 다양한 관점에서 접근한다면 훨씬 쉽게 그리고 더 좋은 해법을 찾을 수 있다는 것을 배웠습니다.

```
struct node { struct node *next;
              struct node *p,*l,*r;
              int key; };
struct node *q1s, *q1t, *q2;

void move_request() {
 struct node *c;
 c = list_remove_first(&q1s);
 if(c==0) return; //trans(list->tree)(c)
 tree_remove(c);  //move(c,gamma)
 list_add_first(&q2,c);
 c = 0;           //moveRgn(gamma,beta)
}
```

(a) C code



(b) a snapshot of data structure

4 세미나와 워크샵

Seminar

2011 10 20 **Software Development, Learning from Trenches**
Charles Hymans, EADS

2011 09 15 **How to Make Ad Hoc Proof Automation Less Ad Hoc**
Derek Dreyer, Max Planck Institute for Software Systems

2011 08 26 **Parametric abstract domains for shape analysis**
Xavier Rival, Ecole Normale Supériéuré

2011 08 22 **WebBlaze: New Techniques and Tools for Web Security & BitBlaze: Computer Security via Ninary Analysis**
Dawn Song, U.C. Berkeley

2011 04 14 **Regular Languages and Applications**
YoSub Han, Yonsei University

2011 03 15 **Type Inference for Bimorphic Recursion**
Makoto Tatsuta, National Institute of Informatics, Tokyo

2011 02 14 **Reasoning about correctness of software transactional memory**
Tarmo Uustalu, Institue of Cybernetics at TUT, Tallinn

2011 02 11 **SSLShader: Cheap SSL Acceleration with Commodity Processors**
Kyung Soo Park, KAIST

The 7th ROSAEC Center Workshop

기간: 2012-01-16 ~ 01-19

장소: Lo Ka Chung University Center, HKUST, Kowloon, Hongkong

Workshop 개회 및 센터 활동 보고
이광근, 서울대학교

양방향 상향식 모양 분석기
이욱세, 한양대학교

Automated Analysis of Industrial Embedded Software
김문주, KAIST

프로그램 분석: 학교에서 현장으로, 그리고 다시 학교로
박대준, 서울대학교

Cost-Aware Triage Ranking Algorithms for Bug Reporting Systems
박진우, POSTECH

Influence maximization in complex network
정교민, KAIST

Theorem Prover Approach to Semistructured Data Design
Scott Uk-Jin Lee, Hanyang University

Adding Modules and Functions to FFMM
김지응, KAIST

Formal Specification of a JavaScript Module System
강성훈, KAIST

접근 경로 분석
하동수, 한양대학교

ACSR을 이용한 계층적 실시간 시스템 분석
황대연, 고려대학교

Automatic Generation of Code-clone Reference Corpus
이효섭, 한양대학교

Concolic 테스팅 기법을 구현한 KLEE 테스팅 도구의 사례 연구
김용주, KAIST

x86 실행파일에서 악성코드의 행동을 정적으로 찾아내기
이승중, 서울대학교

다단계 언어의 단계 풀기 변환
최준원, 강지훈, 서울대학교

Coverage Guided Random Testing for Concurrent Program
홍신, KAIST

Sound non-statistical clusteing of static analysis alarms
이우석, 서울대학교

A theorem prover for Boolean BI
박종현, 서정봉, POSTECH

Gmeta의 평가와 보다 쉬운 정형증명에 대한 연구
이계식, 한경대학교

For More Usable This Type
나현익, KAIST

Concolic Testing to Real-Word Binary Programs
Duc Bui Hoang, 한양대학교

의미 분석에 기반한 겉변화 코드 검출
허기홍, 서울대학교

Analysis of information cascades in arbitrary networks
이슬기, KAIST

Tester-Centric Automated Testing: Bringing Humans Into the Loop
Mat Staats, KAIST

ScanDal: 안드로이드 앱의 개인정보 누출을 잡아내는 정적분석기
김진영, 윤용호, 서울대학교

100만라인 프로그램의 전체 분석
오학주, 서울대학교

효율적인 검증을 위한 컴포넌트 모델의 추상화와 가시화 기법
박민규, 경북대학교

성능버그 무결점 CPU 개발
노한얼, 채동주, POSTECH

성능버그 무결점 GPU 컴퓨팅
김영석, 김한희, POSTECH

네트워크 기반 시스템 명세 및 검증을 위한 Z 프레임워크
신제훈, 고려대학교

빠른 3x3 행렬 곱셈 방법
김진, 서울대학교

Scalable kernel k-means clustering via centroioid approximation
강병곤, KAIST

연구를 시작하며, 하이퍼바이저 Xen의 검증
조성근, 서울대학교

학습 알고리즘을 사용하여 반복문 종료 분석하기
이원찬, 서울대학교

안드로이드 앱의 전력소비량 상대적 비교를 위한 정적 분석
이영석, 서울대학교

기호실행을 이용하여 프로그램 보안 취약점 찾기
박원현 서울대학교

달빅 바이트 코드의 워터마킹 분석
유병준, 서울대학교

달빅 바이트 코드의 정수 범위 도메인 분석
정지수, 서울대학교

An Empirical Study on the Rewritability of the with Statement in JavaScript
박창희, 이홍기, KAIST

Instance Clone Search on Eclipse (system demo)
박진우, POSTECH

성능버그 무결점 시스템 개발 및 운영
김장우, POSTECH

요약파싱기법으로 동적으로 생성되는 HTML의 유효성 검사하기
김현하, 한양대학교

Measuring the integrity of operating system with hardware support
문현곤, 서울대학교

Tipping point of information spreading in random clustered network with arbitrary
임성수, 서울대학교

Recursive Types with Type Abbreviations
임현승, POSTECH

G/F FOYER
ROSAEC-HKUST CSE Joint Workshop on
Software Analysis for Error-Free Computing
HKUST January 16-19, 2012

The 6th ROSAEC Center Workshop

기간: 2011-06-25 ~ 06-28
장소: 경기도 파주 지지향

Workshop 개회 및 센터 활동 보고
이광근, 서울대학교
복잡계망에서의 정보 흐름 모델과 분석
정교민, KAIST
경북대학교 데이터베이스 연구
한욱신, 경북대학교
Dynamic Binary Translator
권용인, 서울대학교
Random Sampling Algorithms with Applications
정교민, KAIST
COQ과 객체 지향 언어 - 쉬운 타입 안전성 증명을 위한 COQ 표현법
김지응, KAIST
바심 프로그래밍 언어와 새로운 중학교 컴퓨터 교재 개발
정재성, KAIST
Malware Protocol Reversing and Botnet Take-Down
강병훈, George Mason University
Project Hugeti: 재미로 하는 프로그램 테스팅
김유일, 서울대학교
안전한 JavaScript프로그램 개발을 위해
류석영, KAIST
빠른 3X3행렬 곱셈법을 자동으로 찾아내기
김진, 서울대학교
Dalvik 바이트코드 분석을 위한 핵심언어 디자인
김진영, 서울대학교
웹 어플리케이션의 접근 제어 취약점에 대한 자동 탐지 기법
송호길, 한양대학교
Cop을 이용한 이분 그래프의 표현과 증명
배성경, KAIST
PLaSSE Research in 2011
도경구, 한양대학교
정형기법 연구실 연구 진행 상황
최진영, 고려대학교
The Implicit Calculus
최원태, 이원찬, 서울대학교
자바스크립트 프로그램에서 with의 사용에 대한 연구
박창희, KAIST
Desugaring을 이용한 JavaScript모듈 시스템
강성훈, KAIST
안전한 JavaScript 부분언어에 관한 조사
최재준, KAIST
정적 분석 경보 사이의 종속관계를 이용한 대표 경보 찾기
이우석, 서울대학교
Inheritance and Subtyping
나현익, KAIST
SAT/SMT Solver Summer School 2011(1)
조성근, 이우석, 서울대학교
SAT/SMT Solver Summer School 2011(2)
이승중, 윤용호, 서울대학교

교육과학기술부, 한국연구재단 지정 선도연구센터
ROSAEC center
제6회 소프트웨어 무결점연구센터 워크샵
날짜: 2011년 6월 25일(토)~28(화) 장소: 호텔지지향 대회의실
Coq Mechanizations @ KAIST
Bigraph L

안드로이드 앱의 개인정보 유출 여부 분석
이승중, 서울대학교

Model checking Trampoline Operating System
최윤자, 경북대학교

A Specialization Calculus for Pruning Disjunctive Predicates to Support Verification
Cristian Andrei Gherghina, 서울대학교

Lagrangian Multiplier 방법을 이용한 제약 조건하에서의 서브모듈러 함수 최소화
임용섭, 서울대학교

코드클론표본 집합체 자동 생성기
이효섭, 한양대학교

다단계 언어에 옮겨 적용할만한 테스트 메트릭
김택수, 서울대학교

복합값을 다루는 기호실행을 위한 SOS들
이주용, 고려대학교

다중 쓰레드 프로그램의 경쟁상태오류 검출기법 분류
홍신, KAIST

Boolean BI를 위한 중첩구조를 이용한 귀추계산법
박종현, POSTECH

Identifying Non-Essential Changes Using Semantic Code Clone Detection
정영범, 서울대학교

CPS Transformation of Lips-Like Multi-Staged Languages and Applications
Ludovic Patey, 서울대학교

Adversarial 통신망에서 Max-Weight 프로토콜의 안정성
임성수, KAIST

문자열 분석을 위한 요약파싱
김현하, 한양대학교

Integrating Code Search into the Development Session
이무웅, POSTECH

OCaml 일종 모듈과 재귀 모듈 프로그래밍
임현승, POSTECH

Sparse Analysis Framework
오학주, 허기홍, 고윤석, 서울대학교

Project Hugeti: Humans Generate Test Inputs
유병준, Saransh, Ludovic, 서울대학교

Parsimonious Algorithm for Distributed Ranking on Complex Networks
허우람, KAIST

A Hybrid, Directed Test Suite Augmentation Technique
김윤호, KAIST

TRM: Supporting pattern matching queries for road-network trajectory
박진우, POSTECH

Comparing two different map-reduce implementations:Hadoop and Plasma
김동원, KAIST

Cop Mechanizations @KAIST
김지응, 배성경, 류석영, KAIST

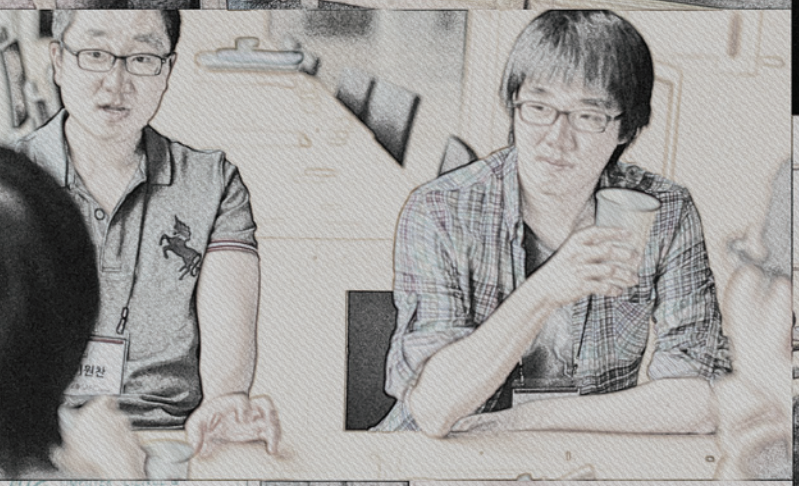
컴포넌트 모델 정제 과정의 행위 일관성 검증을 위한 모델 변환기
윤상권, 박민규, 경북대학교

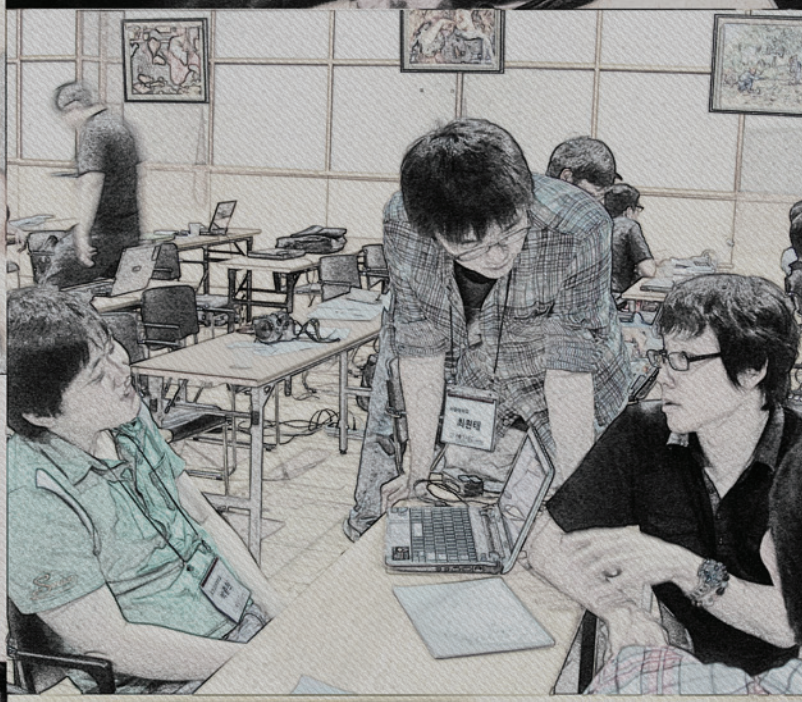
DOM Based Cross Site Scripting 취약점 분석
김세진, 한양대학교

예제로 보는 재귀 모듈 프로그래밍과 이를 위한 구문 기반 타입 시스템
임현승, POSTECH

CosTriage: A Cost-Aware Triage Algorithm for Bug Reporting Systems
박진우, POSTECH

협력연구 이야기 5





Department of Computer Science
and Engineering

田家炳廳
Tin Ka Ping Hall

My First Experience of Research and Korea

Visitor Saransh Srivastava
Sophomore-Undergraduate, IIT Kanpur, India
Period 2011-05-07 ~ 2011-07-19

Visiting ROPAS lab was like a milestone in my academic life. It was the first time I had ever worked in a research lab and that too with people who are far more knowledgeable than me in the area. Before my visit I was a bit worried because of the language barriers but soon I was made comfortable with the warm heartedness of Prof. Yi, Woosuk, Ms.Hana and other lab members. I respect and admire Prof. Yi who can manage so many things, right from research to workshops to weekly seminars, so efficiently.

Being a sophomore, I was not very academically knowledgeable to collaborate on the semantic analysis research front. Prof.Yi quickly understood this and included me into a group where I could learn and produce something meaningful. Throughout my stay everyone in the lab was so helpful that I never felt out of place. The research lab is a balanced combination of research and fun. It is like a big happy family and everyone is invited to join in.

I happened to have the opportunity to spend some weekends with Yoonseok and Ludovic who introduced me to the city and culture of Korea. Being myself from a multi-cultural country I was pleased to understand the Korean culture. The wonderful memories of the time spent with them is something I will cherish for the rest of my life. I hope that in near future I will see more interesting work produced from the research group and hope to work with them in future again.

A Wonderful Summer in Korea

Visitor Cristian Gherghina
Ph.D Student, National Unversity of Singapore
Period 2011-06-01 ~ 2011-09-05

I got the opportunity to visit ROSAEC for a three months summer internship under the supervision of Prof Yi.

My initial plan for the summer was to investigate the applicability to program analysis of the specialization calculus we developed for program verification with separation logic. To this end I interacted with several of the students involved in one of the analysis tools developed at ROSEAC. This lead to an interesting initial prototype the construction of which gave me a much better understanding of the practical issues behind such tools and the difference in the focus of analysis versus verification tools.

Extremely useful for my integration in the research group and projects were the weekly show and tell sessions and the follow-up discussions. These spurned some note-worthy questions, one in particular stuck with me: how can we verify programs with multistage constructs. Leveraging on the previous work of Prof. Yi on type-systems for multistaged languages, I spent the rest of my internship constructing a Hoare-logic for verifying such programs and proving it sound. I got a special satisfaction from picking-up COQ and using it to construct a machine checked soundness proof.

Overall thanks to Prof Yi and his group I spent the summer both investigating an older question related to the generality of previous work and also picking up a completely new set of skills. Also I got the chance to experience an extremely interesting culture in a country I will definitely visit again.

6 방문기 및 에세이

POPL/VMCAI 2011

글쓴이 최원태
U.C. Berkeley, Computer Science 박사과정
방문지 Austin, Texas, USA
방문기간 2011-01-22 ~ 2011-01-30

텍사스 오스틴에서 열린 POPL/VMCAI 2011 학회에 참석하였다. 지도교수님이신 이광근 교수님과 같은 연구실에서 박사과정을 하고있는 학주형, 석사과정인 성근이형, 우석이와 함께 한 여정이다. 우리 연구실에서는 학주형이 VMCAI에서 논문을 한편 발표하였고, 나 또한 POPL에서 논문을 한편 발표하였다. 공교롭게도 두 사람 다 스페인 마드리드에서 열린 작년 POPL/VMCAI때는 참관자 신분으로 참여했다. 달라진 입장에서 경험한 학회는 새로웠다. 우선 가는 비행기 안에서 부터 리허설을 했으니까. 작년에는 마드리드 가이드북을 보던 우리였다.

학회장소인 Omni Hotel은 유리로된 네모꼴의 건물 안에 삼각형모양의 작은 건물 두개가 들어가있는 형상이었다. 두 작은 건물 사이는 유리천장으로 가려졌지만 태양광으로 채광되고 위가 뚤린 실내광장이 자리하고 있었다. 놀라운 것은 그 넓은 실내광장을 에어컨디셔닝 하고 있다는 점이다. 미국답다는 말 이외에는 적합한 표현을 찾지 못했다.

POPL'11 논문 발표

논문

POPL'11 첫날 점심에 "Static Analysis for Multi-Staged Programs via Unstaging Translation" 논문을 발표하였다. 논문에서는 다단계 프로그램(프로그램 실행중에 프로그램 코드가 생성되고 생성된 코드가 다시 실행되는)을 의미가 동일한 단계없는 프로그램으로 변환하여 분석하는 내용을 다루고 있다. 이 논문은 터키 Ozyegin 대학의 Baris Aktemur 교수, 지도교수님이신 이광근 교수님, 일본 NII의 Makoto Tatsuta 교수님과 함께 작성하였다. 논문은 약 2년전에 UIUC의 박사과정이던 Baris가 자신의 박사논문 원고를 이광근 교수님께 보내면서 시작되었다. 당시 나는 다단계 언어를 위한 정적분석을 고안하다가 막다른 골목에 다다른 상태였다. Baris의 논문을 보고 눈이 번쩍했다. Baris는 "타입을 보존하는 변환"을 이미 고안한 상태였는데 조금만 고치면 실행의미도 보존하게 만들수 있을것 같았다. 이광근 교수님께서 Baris를 초청하면서 협력이 시작되었다. 1년 반에 걸친 고투 끝에 논문을 완성 할수 있었다. Baris의 어깨에 올라타고 이광근 교수님의 지휘를 따라가서 고지에 올랐으니 나는 운이 좋은 편이다.

발표준비와 발표

처음 해보는 학회 발표였다. 너무 긴장해서였는지 발표순간은 잘 기억나지 않는다. 머리를 비우고 리허설했던 내용을 리플레이하고 내려왔다. 한가지 기억나는 것은 세번째 슬라이드를 발표할 즈음 연구실 동료들과 Makoto 교수님이 갑자기 웃음을 터뜨린것. 발표중에는 뭔가 실수를 했나 싶었다. 후에 이우석 학생의 증언을 들어보았다: "목소리와 행동이 너무도 차분하여 처음에는 전혀 긴장하지 않은줄 알았어요. 스크린상에 비친 레이저 포인터 사시나무 떨리듯 흔들리는것을 보고 역시나 했어요" 큰 실수는 없었나보다.

긴장했음에도 발표를 무사히 마칠수 있었던 것은 발표 자료에 대한 자신감 덕분이었다. 내가 영어를 못해도 이 발표자료라면, 그리고 POPL 에 참여하는 사람들이라면 알아들을 것이라고. 이광근교수님과 발표자료를 보름정도 두들겼고, Baris에게도 두차례에 걸쳐 긴 피드백을 받았다. 간단히 넘어가도 되는 부분과 자세한 설명이 필요한 부분은 어디인지를 알았고 그에 따라 발표도 준비할수 있었다. 따로 대본을 만들지 않았다. 대신 직접 말하면서 녹음해보고 빠르게 들어보면서 표현이 이상한 부분을 찾고 시간을 조절했다. 한번에 한시간 정도걸리는작업이었는데, 세번쯤 반복하면 한결 나아졌다는 느낌을 받을 수 있었다. 학회에서 편안하게 발표하듯 보였던 선배들은 발표전에 얼마나 노력했을지를 생각하니 세삼 대단하게 느껴졌다.

발표가 끝나고 다섯명 정도가 질문을 했다. 이때부터 정신이 돌아왔는데, 다단계 언어에 관심을 가진 사람이 생각보다 많아 한번 놀랐고, 질문을 듣기가 쉽지 않아 다시 한번 놀랐다. 끝내 질문을 알아들을 수 없었던 두명에게는 답변이 길어질 듯 하니 세션이 끝나고 만나자고 얼버무렸다. 다행히도 세션이 끝난후에는 제대로 답을 해줄수 있었다. 그중 한명은 Scheme에 관련된 연구를 하는 학생인데 처음에 같은 어휘를 서로 다른 어휘로 사용하여 의 사소통에 어려움이 있었다. 서로 vocabulary를 맞춰가면서 의사소통을 이루어 나가는 것도 재미있는 경험이었다.

느낀 것

우선 내용을 확실하게 소화해야 한다. 당연하게 들릴수도 있다. 그렇지만, 연구를 하는 도중에는 의외로 세심한 디테일을 점검하지 않고 넘어가는 일이 많다. 논문을 쓸때까지도 교수님과 동료들의 도움을 받거나 이해하지 못한 부분을 얼버무리는 식으로 넘어 갈수가 있다. 하지만 발표때는 도망칠 수 없다. 자료를 만들때, 처음에는 도무지 25분짜리 슬라이드로 내용이 압축되지 않아 당혹스러웠다. 준비를 하면서 내가 낸 아이디어라고 내가 소화하고 있는 것은 아닐수도 있으며, 소화하지 못하면 25분안에 전달할수 없다는 사실을 알았다. 어떻게 설명할지 고민하고 제대로 이해하지 못한 디테일을 채우기를 반복하다가 출발 4일 전쯤 비로소 필요한 디테일은 다 들어있으면서 장황하지 않다는 느낌이 들었다. 차이는 자신의 아이디어를 둘러싼 디테일들을 온전히 이해하고 있느냐에 있는듯 했다.

또 한가지. 듣기가 잘되야 한다. 긴장감 때문인지 아니면 긴장풀린 때문인지, 발표 직후 사람들의 질문을 잘 알아들을 수가 없었다. 연구자들은 대체로 심성이 착해서 느리게 말하고 쉬운 단어만 써서 더듬더듬 말해도 뜻이 통하면 잘 참아준다. 그런 반면에 본인이 하는 말이 알아듣기 쉬운지 어떤지는 별로 신경쓰지 않는다. 제대로 듣고 그에 대해 제대로된 내용으로 답을 한다면 틀린 영어를 사용하더라고 일단 대화는 이루어진다. 듣는 능력이 중요하다. 다양한 억양으로 다양한 빠르기로 하는 말을 일단 알아들을 수 있어야 커뮤니케이션이 올바르게 이루어질것 같다.

학회 발표 자체에는 그다지 강한 인상을 받지 못했다. 대부분의 발표가 발표 내용을 전달한다는 느낌보다는 통보를 받는다는 느낌이었다. 발표를 잘하는 것이 무척이나 어렵다는 사실을 반증하는 셈일 것이다. 유익하지 않았던 발표의 대부분은 30분의 시간동안 기술적인 세부사항만을 다루는 우를 범하고 있었다. 내가 발표에서 기대했던 것은 연구의 목적과 동기, 그리고 핵심이 되는 연구자들의 발견 혹은 직관에 대한 이야기였다. 어차피

연구 내용을 이해하려면 그들이 쓴 논문을 붙잡고 씨름해도 부족하다. 그렇다면 발표의 목적이란 응당 1) 자신이 연구과정에서 발견한 중요한 직관을 사람들과 나누고 2) 자신이 쓴 논문을 읽게끔 유도하는 것이 되어야 한다고 생각한다. 30분 간 알아듣지도 못하는 수식을 늘어놓는 발표자들에게는 야속한 마음과 안타까운 마음이 일었다. 같은 분야의 연구자의 마음도 훔칠 수 없는데 더 많은 사람에게 영향을 미치는 훌륭한 연구란 있을 수 없다고 생각했기 때문이다. 내 자신은 그런 실수를 하지 않도록 발표의 목적을 잊지 말아야겠다고 생각했다.

보고 들은 것들

한국의 약진

이번 POPL/VMCAI에서는 한국인의 발표가 각각 두팀이 있었다. VMCAI에서는 우리 연구실의 학주형과 카이스트의 김세원 박사님이 발표를 하셨다. 학주형은 Local Reasoning을 적용해서 실행흐름을 고려하는 함수간 분석의 비용을 획기적으로 줄이는 내용을 발표하였고 김세원박사님은 String Analysis를 오토마타연산을 사용해 요약해석틀 안으로 끌어들인 이론적인 성과를 발표하였다. POPL에서는 MPI에 계시는 허춘길박사님이 ML프로그램과 그 컴파일 결과인 바이너리 프로그램 사이에 contextual equivalence가 있음을 보인 내용을 발표하셨다. 내 발표 바로 다음인데 방이 달라서 우리 연구실 사람들은 우루루 이동해야했다. 직접 발표하지는 않았지만 양홍석 박사님도 POPL에 논문을 한편 내셨다.

> 발표의 목적이란 응당
>
> 1) 자신이 연구과정에서 발견한 중요한 직관을 사람들과 나누고
>
> 2) 자신이 쓴 논문을 읽게끔 유도하는 것이 되어야 한다고 생각한다.

VMCAI

Access Nets : Modeling Access to Physical Spaces : 건물의 접근 권한 및 관리 정책이 의도한대로 작동하는지를 정적분석 기법을 사용하여 검증하는 내용을 소개했다. 어느 방에 몇명의 사람이 들어있는지가 프로그램의 상태에 대응되고, 방의 연결상태와 관리정책이 상태전이 함수에 대응된다. 문제의 크기를 줄이기 위해 주어진 건물을 "의미가 동일" 한 다른 건물로 변환하기도 하고, constant propagation과 비슷한 아이디어를 사용하기도 하는 등 여러가지로 흥미로웠다. 관리정책과 건물의 생김을 xml로 옮기는 번잡스러운 과정에 대한 해법을 제시하지 않았다는 점에서 완성도가 높지는 않았다.

Proving Stabilization of biological systems : 이 논문에서는 세포가 가질수 있는 상태를 정적분석으로 추적하는 내용을 다루고 있다. 핵심 아이디어는 Local Reasoning을 이용해서 세포 단위로 상태를 분석하고 그렇게 모인 정보를 인접한 세포들에게 전달하여 다시 분석하는 것이다. 생명체를 구성하는 세포들은 외부환경과 가능한 독립적으로 작동하도록 진화해왔기 때문에 Local Reasoning이 아주 잘 어울린다고 한다. 생명체를 분석하는데 Semantics를 정의하고 정적분석을 돌린다니 기분이 이상했다. 이광근교수님께서 종종 "Syntax와 Semantics를 정의할 수 있으면 프로그램을 짤 수 있는 것이고 분석도 가능하다"는 이야기를 하시는데 그것이 얼마나 강력한 아이디어인지 세삼 느껴지는 발표였다.

Distributed and Predictable Software Model Checking : 이 논문은 CEGAR기반 모델체킹을 병렬계산을 이용해 빠르게 만드는 기법을 소개한다. 모델체킹의 경우 하나의 모델에서 다양한 반례를 찾을 수 있다. 또한 어떤 반례를 사용해 모델을 정제하느냐에 따라 분석기 빠른 시간에 끝나기도 하고 메모리부족으로 죽을수도 있다. 그리고 통상적으로 반례가 간단하면 분석이 끝날 확률이 높다고 알려져있다. 이 논문에서는 여러 노드에서 각자 다른 반례를 찾도록 한 다음 그중 가장 간단한 반례를 이용해 모델을 정제 한다. 노드의 수가 많을 수록 더 간단한 반례를 찾을 가능성이 크기 때문에 노드가 증가할수록 검증시간은 점점 빨라지게 된다. 쉬는시간에 발표자에게 이것저것 물어보면서 제대로 이해할수 있었다. 비싸서 할 수 없었던 계산을 병렬적으로 빠르게 수행함으로서 전체계산을 더 빠르게 만드는 아이디어가 내게는 새로웠다. 정적분석에도 그런 여지가 있지는 않을지.

POPL

작년과 다르게 이번 POPL에서는 프로그램 분석 관련 논문의 약진이 두드러졌다. 특히 첫재날에는 Patrick Cousot 교수가 발표마다 질문을 하는 모습이 정적분석학회인 SAS를 연상시켰다. Verified Compiler 관련된 논문이 많았던 것도 인상적이었다. 무엇보다도 학회를 시작하는 첫번째 Invited talk이 Xavier Leroy의 Verified Compiler 연구개괄이으니까. Type을 위한 학회라고 불리던 과거와는 다르게 Verification이 점점 치고올라오는듯해서 기분이 좋았다.

들었던 많은 발표중에 "민감성" 이라는 테마로 묶을 수 있는 논문 네편을 소개한다. 분석 기술이 점점 발전하면서 함수간의 호출관계와 실행흐름 고려하기 시작했다. 호출 관계를 고려하는 분석은 "호출문맥민감" 분석이라 부르고, 실행흐름을 고려하는 분석을 "실행흐름에 민감한" 분석이라고 부른다. 호출문맥이나 실행흐름을 정직하게 따라가면 너무 분석 비용이 비싸다. 그렇다고 무시하면 정확도가 떨어진다. 적당한 중간지점을 찾아야한다. 이번 POPL 에 나온 분석과 관련된 논문들중에 중간지점을 찾고자 하는 논문들이 다수 있다. 그리고 그 결과물은 그 동안 많은 사람이 "최대한의 민감함"을 싼 값에 지원하는 방법을 찾는데 집중해 온 노력을 무색하게 만든다. 발표를 들으며 프로그램분석 분야가 점점 발전하고 있다는 확신이 들었고 아직 할일이 많다는 생각도 들었다. 무술의 달인들은 평상시에는 힘을 빼고 있다가 정확하게 필요한 만큼만 힘을 사용한다고 하는데 분석분야는 이제 막 그러한 필요성을 배운듯 하다. 개인적으로 석사를 시작하면서부터 관심이 있었던 주제라 아주 흥미로웠다.

Pick Your Contexts Well : Understanding Object-Sensitivity : 절차형 언어로 작성 된 프로그램을 분석할 때는 호출지점을 이용해서 호출문맥을 구분한다. 객체지향 언어에는 또 다른 옵션이 있다. 객체지향 언어에서 함수는 객체에 종속되어있으므로 호출된 함수를 소유한 객체를 통해 호출문맥을 구분하는 것이다. 이런 아이디어를 object- sensitivity 라고 부른다. 이 논문에서는 객체를 보는 대신 객체의 타입 보는 아이디어를 제시한다. 타입을 사용하면 경우의 수가 줄어들어서 분석이 빨리 끝나게 된다. 흥미로운 점은 정확도는 큰 차이가 없더라는 것이다. Java프로그램을 계속해서 분석하던 사람이 낼 수 있는 아이디어인듯했다. 발표를 듣고 숙소에 돌아와 생각하면서 아이디어의 단순함과 효과에 감탄하였다. 관찰이 면밀하고 정확할수록 아이디어는 간단하면서 강력해지는 것 같다.

Learning Minimal Abstractions : 모든 함수가 동일한 만큼 문맥을 구분하면서 분석될 필요가 있는 것은 아니라는 것은 분석을 해본사람이면 누구나 알고있을 것이다. 그렇지만 정확하게 어디서 얼마 만큼 민감해져야 하는지 아직 제대로 알고있는 사람이 없었다. 이 논문에서는 몇가지 프로그램을 분석하는데 필요한 최소한의 민감성정도를 기계학습으로 찾아서 보고하고있다. 결과가 놀랍다. 찾고 보니 평균적으로 100개중 2-3개의 함수만 민감하면 (pointer 분석에는) 충분하더라는 것이다. 논문의 주된 내용은 슈퍼컴퓨터로 수십년 돌려야 찾을 수 있는 값을 기계학습을 동원하여 어떻게 빨리 찾는지 테크닉을 다루고 있다. 정작 우리에게 의미있는 것은 결과가 던지는 한줄의 메세지 : 민감해져야 하는 정확한 지점을 안다면 많은 민감성이 필요하지 않다.

Calling Context Abstraction with Shapes : 호출문맥을 고려하는 분석분야에서 풀리지 않던 난제가 한가지 있으니, 어떻게 하면 재귀함수를 싼 비용으로 정확하게 분석하느냐는 것이다. 함수호출 문맥을 아무리 자세하게 고려해도 재귀함수를 분석하다 보면 문맥민감성이 부족해서 결국 분석이 부정확해지기 마련이다. 이 논문에서는 함수호출이 만들어내는 call-stack을 재귀적 자료구조로 생각하고 모양분석기법을 적용하여 호출문맥을 요약하는 기법을 제시한다. 실험이 부실하여 실제로 얼마나 분석을 정확하게 만들지는 알 수 없지만, 문맥민감분석이 가야할 길을 한가지 제시했다는 점에서 의미있다고 생각되었다. INRIA 사람들 특유의 이해하기 어려운 슬라이드와 발표자의 프랑스 억양덕에 발표 자체는 따라가기는 쉽지 않았다.

Points-To Analysis with Efficient Strong Updates : 이 논문은 실행흐름을 얼마만큼 존중해야 하는가에 대한 논문이다. 실행흐름을 항상 고려하면 분석 비용이 비싸니 strong update가 발생하는 경우에만 실행흐름을 고려하자는 것이 아이디어이다. 그랬더니 분석비용은 실행흐름을 무시한 분석과 거의 같고 정확도는 실행흐름을 고려한 경우와 거의 같더라는 것이다.

맺는말

multi-session으로 진행된 학회에는 처음 참석해 보았는데, 참석자로서 준비가 부족한 감이 있다. 최소한 어떤 발표들을 따라서 들을지 미리 고민해보고 스터디를 한번 해 봤어야 하는데. 작년 POPL/VMCAI때는 이해가 되지 않더라도 논문을 한번씩 읽어보고 발표를 들어서 꽤 많은 내용을 건질 수 있었는데 올해는 그렇지 못했다. 반성. 또 반성. 아는만큼 얻어간다.

홍콩과기대 방문연구

글쓴이 허기홍
서울대학교 프로그래밍 연구실 박사과정
방문지 홍콩과학기술대학교, 홍콩
방문기간 2011-09 ~ 2012-02

홍콩 구룡반도. '영웅본색'의 향기가 아직 진하게 배어있는 그 곳에서 연구 할 기회가 생겼다. 홍콩의 무더위와 과기대의 살을 에는 에어컨 바람 속에서 지낸 6개월은 값진 시간이다. 넓은 세계를 보고, 좋은 사람들을 만나 생각하고 공부할 수 있던 좋은 계기였다.

홍콩과기대

아시아의 신흥 명문대학인 홍콩 과기대의 첫 느낌은 굉장히 복잡했다. 학교가 큰 건물 하나로 구성되어 있어서 길을 잃기 십상이다. 첫날 무거운 이민 가방을 끌고 연구실을 찾아 이리저리 헤메던 기억이 생생하다. 곳곳에 있는 경비원들도, 교수님으로 보이는 한 서양 여자분도 각 방의 세세한 위치는 몰라서 나와 함께 헤멨다.나중에 알고보니 비단 초행자만의 문제가 아니었다. 과기대 학생들도 스마트폰의 홍콩과기대 앱을 이용해 방 번호를 검색한 후 찾아가곤 했다. 나는 한달 쯤 지나서야 비로소 학교 구조가 조금씩 이해가 가기 시작했다.

학교 시설이 기대만큼 좋지는 않았다. 연구실이 비좁고 창문이 없어서 좀 답답했다. 놀랍게도 홍콩과기대의 상당수 교수님들도 창문이 없는 좁은 방에서 연구를 하고 계셨다. 기숙사나 학교밖 건물들을 돌아다녀 보아도 집을 보는 한국인들의 높은 기준에 홍콩의 집은 약간 못미쳤다.

허나 그 이외 다른 것들은 아주 훌륭했다. 무엇보다도 학내 여러 식당에서 제공하는 다양하고 푸짐한 음식. 또 일사천리, 철두철미한 학사행정과 연구지원. 무료로 제공되는 여러 체육시설도 방문기간동안 홍콩과기대에서 제대로 누린 혜택이었다. 그리고 중국 본토, 한국 등 여러 나라에서 온 열정적인 대학원생들이 진지하게 연구에 임하는 모습도 인상깊었다.

> 산을 대하는 산악인의 자세를 배웠다.
> 섣불리 덤벼들지 않고 차근차근 밟아
> 올라가는 자세. 값진 경험이었다.

방문연구

홍콩과기대의 PAM (Program analysis and Mining) 연구실에서 김성훈 교수님의 지도를 받으며 6개월간 연구를 하였다. 서울대에서 공부하고 있는 프로그램 정적 분석 기술을 이용해서 소프트웨어 공학 분야의 문제를 해결하는 것이 목표였다. 늘 가까이서 조언을 구하던 지도 교수님과 선배들의 품을 떠나는 것은 긴장되는 일이다. 허나 내 수준을 스스로 적나라하게 벗겨볼 수 있는 좋은 기회이기도 하였다.

정적 분석을 이용하여 소프트웨어 개발 과정에서 나타났던 겉변화 코드들을 찾아내보기로 했다. 개발과정에서 새로운 기능을 추가하기도, 오류를 고치기도 하면서 소프트웨어는 계속 변한다. 하지만 그 중에는 속 내용은 변하지 않고 겉 모양만 변하는 경우도 많다. 주로 코드를 보기 좋게하여 유지보수를 쉽게 하려는 목적이다. 그런데 이런 겉변화 정보는 개발 과정에서 일어났던 일을 관찰하여 의미있는 정보를 추출(Software mining)하는 데 걸림돌이 된다. 개발과정에서 많은 사람들이 관찰하고 싶은 정보는 새로운 기능 추가나 오류 수정 등인데 기존 버전 관리 도구는 각 버전의 의미를 구분해서 기록하지 않기 때문이다. 따라서 개발과정에서 일어났던 별 의미 없는 겉변화 정보를 찾아내서 의미있는 변화만 취할수 있도록 해줄 필요가 있다.

시작은 순조로웠다. 기존 방식 보다 프로그램의 의미 변화를 더 효과적으로 감지해 낼 분석기가 있었기 때문에. 허나 시간이 갈수록 헛점이 많이 보였다. 현실 세계의 프로그램은 우리의 상상 이상으로 복잡했다. 또 그동안 연구실에서는 C 프로그램만 들여다 보고 있었던 터라 이 연구에서 대상으로 하던 Java 프로그램에 대한 경험과 지식이 부족했다.

시간은 흐르고 해야할 일은 점점 많아지면서 순간의 판단실수가 논문 제출 실패로 직결되는 상황이 되었다. 소프트웨어 개발 과정을 모두 훑어야 하는 작업이었기에 새로운 기능을 추가하고 결과를 보는데 하루정도 걸렸다. 결과가 좋지 않으면 우리의 분석 방식 중 어떤 부분을 어떻게 고쳐야할지 판단을 내려야 했다. 모든 경우를 만족시키는 절대적인 답은 없었고 그동안 실험한 결과를 바탕으로 최적을 결정해야 했다. 서버 여러 대에 분석기 여러 개를 동시에 돌려놓고 그때 그때 나오는 결과를 기록하고 기억했다. 결정을 내리려면 '감'이 필요했고, 그 감은 관찰과 실험으로부터 나온다는 것을 여실히 새겼다.

치열하고 간절하게 논문을 준비했지만 결국 마감기간 내에 좋은 결과를 내지 못했다. 실험이 한창이던 새벽, 중앙 집중형이라 꺼지지 않는 연구실 에어컨 아래가 너무 추워서 건물 밖에 나가 몸을 녹이고 오는 웃지못할 상황을 겪으며 사투했지만 실패했다. 마감을 눈앞에 두고 나온 결과가 좋지 않은 것을 확인 했지만, 동료들에게 그만하자는 말이 왜 그리 나오지 않았던지.

아쉬운 실패를 뒤로 하고 보완하여 얼마 후 저널에 논문을 제출했다. 모든 실험 환경과 결과를 다시 정리하고 분석기를 개선했다. 동료들, 교수님들과 함께 논문을 써서 다른 사람에게 보여주고 다듬고 제출하였다. 산을 대하는 산악인의 자세를 배웠다. 섣불리 덤벼들지 않고 차근차근 밟아 올라가는 자세. 값진 경험이었다.

홍콩 음식

홍콩의 다양한 음식을 맛보는 것도 머무는 동안 빼놓을 수 없는 즐거움이었다. 홍콩에서도 쌀밥을 매끼 먹을 수 있지만 우리 나라 쌀과 달리 윤기가 없고 매우 거칠었다. 대신 맛있는 국수와 딤섬에 반해서 방문 기간 동안 이를 주식으로 삼았다. 해물육수와 꼬들꼬들한 면빨이 특징인 홍콩 전통 국수 완탕면, 사천지방에서 유래한 탄탄면, 짜장면의 원조 작장면 등을 번갈아가며 먹곤 했다. 또한 수십가지에 달하는 딤섬을 종류별로 먹고 이름을 외우는 것도 재미 있었다.

여러 국수 중에서 기억에 남는 두 가지가 있다. 몽콕 (Mong Kok) 야시장 주변에 서문정(西門町)이라는 식당은 약간 특이한 탄탄면을 판다. 보통 탄탄면은 사천식 매운 국물에다 견과류를 갈아넣어 고소한 맛을 내는 국수이다. 헌데 이 집은 매운맛이 전혀 나지 않는 특이한 탄탄면을 팔고 있었다. 가게 이름에서 유추해보건대 대만식이 아닌가 싶다. 다른 탄탄면보다 고소한 맛이 진하고 약간 단맛도 난다. 다른 곳에 비해서 자극적이지 않고 정갈하여 며칠 있으면 다시 생각이 났다. 같이 시켜먹은 가지 요리와 만두도 훌륭했다. 내 가 갈때마다 "감사합니다" 하며 인사하시던 친절한 아저씨 때문에도 더욱 다시 가고 싶은 집이다.

또 한 가지는 이름만 듣고 경악을 금치 못했던 토마토 국수이다. 홍콩섬 어느 길거리에 천막과 탁자를 놓고 장사를 하는 쌩홍윤(勝香園)이라 는 집이었다. 도착하니 이미 수많은 사람들이 줄을 서서 기다리고 있었 다. 그날 준비한 재료가 다떨어지면 2~3시에도 장사를 그만하는 집이라고 한다. 면은 그냥 인스턴트 라면인데, 토마토와 햄 등이 들어간 국물이 아주 좋았다. 소금에 절인 레몬을 넣은 사이다와 레몬 빵도 팔았는데 이도 특이했다.

한국인들이 홍콩에 많아서 그런지 큰 식당 메뉴판에는 한글도 많았다. 허나 잘못된 표현도 많았는데, 그 중 "먹으십시요"는 충격이었다.

사람들, 그들의 연구, 소통

머물면서 각지에서 온 여러 좋은 사람들을 만나고 다양한 이야기를 나누었다. 홍콩과기대에서 김성훈 교수님의 지도를 받는 연구원들, 북경대 박사 과정학생으로서 나와 같이 방문 학생으로 온 량광태 (Liang, Guangtai), UC Santa Cruz 에서 온 이제헌박사님과 Caitlin Sadowski 모두 소중한 인연들이다.

만났던 모든 사람들의 큰 목표는 같았지만 가는 길은 다 달랐다. 무결점 소프트웨어와 효율적인 소프트웨어 개발을 향해 가는 여러 길을 보았다. 어떤 이는 프로그램 개발 과정을 주로 살펴보고, 어떤 이는 개발자의 행동 양상과 성향을 살폈다. 동적으로 혹은 정적으로, 통계적으로 혹은 비 통계적으로 각자 목표를 향해 나아갔다. 어느 한 기술도 결코 완벽할 수 없기 때문에 서로의 연구에 꾸준히 관심을 갖는 것이 중요할 수 밖에 없다.

특히 그들과 함께하면서 소프트웨어 개발 과정 기록을 파헤쳐 본 것은 신선한 경험이었다. 그동안 프로그램 분석, 버그 같은 말을 입에 달고 살았지만, 부끄럽게도 오픈소스의 버그리포트 한 번 열어본 적이 없었다. 여러 프로그래머들이 겪은 어려움, 고민, 해결책들이 모두 담겨 있는데도 어렴풋이 그 존재만 알고 있었을 뿐 관심을 갖지 않았다. 앞으로 더 유용한 연구와 그 와중에서 더 나은 판단을 내리기 위해 염두에 두어야할 부분이다.

여러 사람들과 유익한 이야기를 나누기 위해서는 당연히 소통 능력이 필수이다. 영어 실력 자체도 물론 중요하지만, 자기의 연구를 제대로 설명할 수 있는 연습을 해야할 것 을 다시금 느꼈다. 특히 연구가 아직 마무리되지 않았을 때, 걱정과 헛점 투성이인 현상태를 어떻게 다른 사람들에게 설명해야 하는지 계속 고민해야 할 필요가 있다. 연구 진행 상황이 뒤죽박죽이라 앞이 보이지 않을 때, 그때가 바로 다른 사람의 조언이 절실한 상황이기 때문이다.

보통 내가 연구했던 순서 그대로 남에게 말하는 경향이 있는데, 때로는 그것이 원활한 소통을 막을 수가 있다는 것을 깨달았다. 내가 겪었던 일의 순서, 내 생각의 흐름을 상대 방은 전혀 납득하지 못하는 경우가 있기 때문이다. 따라서 언제나 가던 길을 멈추고 돌아서서 그동안 한 일을 논리적으로 재구성한 후 남에게 설명하는 연습이 꼭 필요하다.

연구란?

방문기간 중 Stanford 대학교의 Monica Lam 교수님이 홍콩과기대를 방문해서 그 분의 강연을 듣게 되었다. 컴파일러 책으로 유명한 분이기도 했거니와, 홍콩 오기 전 분석기 성능을 높이는 연구를 하던 중 그 분의 논문을 참고한적도 있어서 어떤 분인지 궁금 했다. 컴파일러나 프로그램 분석 쪽에서 어떤 문제를 갖고 있는지. 허나 강연을 소개하 는 이메일을 보고는 놀랐다. 무수비 (Musubi) 라는 소셜 네트워크를 만들고 있고 이를 소개하는 강연이었다. 강연에 들어가서는 고등학교까지 홍콩에서 나온 사람이라는 것을 알고 또 한 번 놀랐다.

무수비는 기존 SNS에 대한 불만에서 시작되었다고 한다. 페이스북같은 기존 SNS는 그 회사에서 제공하는 서버에 사용자가 올린 모든 자료가 저장되고, 저작권도 회사가 갖는 경우가 많다. 또한 소통과 개방을 표방하지만 역설적이게도 기존 SNS는 폐쇄적이다. 그 SNS에 계정을 만든 사람들만 소통할 수 있고, 점점 그들끼리만 게임, 채팅, 자료전송 등을 하고 있다. 여러 회사들도 자신들의 SNS계정으로 제품을 소개하고 홍보한다. 인터넷이라기보다 인트라넷과 같다.

대신에 무수비는 울타리를 없애고, 당사자만 데이터를 갖게 한다. 마치 페이스북에 카카오톡을 결합한 느낌이다. 특별한 플랫폼에 가입할 필요가 없으며 본인의 단말기에 있 는 전화번호로 서로 소통한다. 내가 올린 정보도 내가 지정한 사람들에게만 보낼 수 있고, 이는 중앙 서버에 저장되지 않으며 수신자와 송신자만이 소유한다.

재미있는 발표였다. 현재 SNS가 어떻게 보면 폐쇄적일 수도 있다는 생각, 기꺼이 개인 정보를 갖다 바치는 사람들에게서 모은 수많은 정보가 한데 모여 있는 상황이 위험하다는 경고. 발표자도 말을 조리있고 시원스럽게 해서 더욱 집중이 잘 되었다.

헌데, 재미있긴한데 돈이 될까? 무슨 연구거리가 될까? 발표를 다 듣고 이런 질문이 떠 오르던 차에 마침 어느 학생이 같은 질문을 했다. 그러자 그 분의 거침없는 대답. 1) 주변 상황에 대한 불만을 토대로, 2) 더 나은 대안을 만드려는 시도를 하고, 3) 그 과정속에서 세부 문제 해결해 나가는 것이 연구라고. 이미 통신 보안과 분산 컴퓨팅 쪽으로 좋은 연구가 진행되고 성과도 나오고 있다고 한다. 답변을 듣고보니 우리 교수님께서 연구실 학생들에게 늘 강조하시는 부분이었다. 다시 한번 아로 새길 수 있었다.

맺음말

방문 기간 정말 많은 것을 배우고 느꼈다. 7년간 서울대라는 울타리를 벗어난 적이 없어서 매너리즘에 빠질뻔 했을 찰나, 많은 긴장거리를 얻어온 좋은 경험이었다. 값진 기회를 주시고 먼 곳에서 늘 격려와 지원 아끼지 않으신 이광근 교수님, 홍콩에서 한 단계 성장할 수 있도록 잘 지도해주시고 보살펴주신 김성훈 교수님, 6개월간 동고동락했던 홍콩과기대 동료들, 소프트웨어 무결점 연구센터의 지원 덕택에 가능했던 일이다.

CPP 2011

글쓴이 김지응
KAIST PLRG 연구실
방문지 Kenting, Taiwan
방문기간 2011-12-07 ~ 12-09

이번 APLAS (Asian Symposium on Programming Languages and Systems) 와 CPP (International Conference on Certified Programs and Proofs) 두 학회가 대만의 켄팅이라는 곳에서 열렸습니다. 이 두 학회 중 저와 한명희 학생, 그리고 배성경 학생은 수요일부터 금요일까지 3일간 진행되는 CPP학회만을 참석을 하였습니다. 이 학회는 제가 처음으로 참석해 보는 국제 학회임과 동시에 처음으로 논문을 발표하는 국제 학회였습니다. 이런 뜻 깊은 경험을 할 수 있는 기회가 생긴 것에 감사를 드리며 제가 관심 있게 들은 발표들을 정리해 보았습니다.

관심 있게 들은 발표들

Simple, Functional, Sound and Complete Parsing for All Context-Free Grammar

CPP Session 3: Formalizations 에서 다뤘던 논문입니다.

발표에 대한 느낌을 기억해 보면 이 발표는 처음에는 간단한 CFG 예시들을 보여주고 문제들을 설명함으로서 처음에 내용들을 따라가기가 매우 쉬웠던 기억이 납니다. 하지만 본론에 들어가면서부터 논문을 그대로 옮긴듯한 화면을 계속 보여줌으로서 약간 실망을 하게 되었고, 한 슬라이드에 너무 많은 텍스트가 있다 보니 집중하기가 힘들었던 것으로 기억합니다.

내용에 대하여 살펴보면, 우선 이 논문의 목표는 CFG에 대한 효과적인 parser generator를 만드는 것 입니다. CFG에 대한 parsing을 만드는 방법 중에 naive한 방법은 left-recursive grammar와 같은 것에서 termination을 보장하지 않고 문제가 생깁니다. 따라서 이 논문에서는 이러한 문제가 발생하지 않으며 CFG grammar specification에 대한 간단하고 (simple), 안전하고 (sound) 완전한 (complete) parser를 만들어 주는 방법을 제안하였고, 이에 대한 성질들을 HOL4 theorem prover로 증명하였으며 또한 다른 parser generator들과 자신들의 방법을 비교해 주었습니다.

기억에 남는 내용은 이 논문에서 제안하는 parser generator는 combinator parser를 만들어 주며 bad parse tree (nontermination 하는 parse tree)를 만들어 주는 parser를 만들어 주지 않기 위해 이를 체크하는 function을 만들어 주었다는 것이 있었습니다. 하지만 이 check function이 어떠한 방식으로 수행되는 지는 정확히 이해를 하지 못 했습니다. 논문을 살펴보니 논문의 Section 5, 6에서 이 내용들을 다루고 있는 것 같습니다.

그 다음 이 논문에서 말 하고 있는 soundness와 completeness 가 무엇인지, 그리고 그것을 어떻게 HOL4로 표현하였는지에 대한 내용을 이야기 하였으며, 마지막으로 이 논문에서 제안한 parser generator로 만들어 진 parser와 유명한 Happy parser generator로 만들어진 parser 로 parse time을 비교해 줌으로서 이 논문의 방식이 뛰어나다는 것을 보여주었습니다.

발표에 조금 더 집중을 하였으면 훨씬 더 많은 내용을 이해할 수 있었을 것 같았는데 그러지 못해서 아쉬움이 남는 발표였습니다.

Teaching Experience: Logic and Formal Methods with Coq.

CPP Session 5: Teaching with PA's 에서 다뤘던 논문입니다.

이 session은 논문 2개에 대한 발표로 구성이 되어 있었으며, 두 논문 다 proof assistant tool을 사용한 강의에 대한 논문들이다 보니 다른 session들에 비하여는 이해하기가 상당히 쉬웠었습니다. 이 두 논문들 중 더 관심이 많이 갔던 논문인 "Teaching Experience: Logic and Formal Methods with Coq"에 대하여 간략하게 제가 느낀 내용들을 요약해 보려고 합니다.

이 논문은 National University of Singapore에서 Coq을 이용한 Logic 수업에 대한 내용을 담고 있는 논문입니다. 이 논문과 관련이 있는 webpage는 http://www.comp.nus.edu.sg/~henz/cs3234/ 이며, assignments나 exam의 solution이 필요할 경우 이 논문의 저자들에게 contact을 하라고 하고 있습니다. Faculty가 아닌 학생이나 연구원에게도 자료를 줄 지 모르겠지만 이 trip report를 쓴 후 메일을 보내 자료들을 요청해 보려고 합니다.

이 논문은 기존의 Software Foundations, CPDT, 그리고 칭화 대학교의 Coq summer school과 관련된 Coq 강의 자료들과는 다르게 논리학을 가르치는 데 있어서의 보조 도구로서 Coq을 사용하는데 중점을 두었다는 것이 특이한 점 이었습니다. 논리학에 중점을 두며 Coq을 사용한 강의라는 점에서는 오히려 박성우 교수님의 전산 논리학 수업과 더 가까워 보이지만, 이는 박성우 교수님의 수업에서의 Coq을 이용한 과제들 보다도 조금 더 Coq을 논리학을 배우는 도구로서 효과적으로 이용하는 데 포커스를 맞추었다는 느낌을 받았습니다.

조금 더 자세하게 적으면, 기존에 있던 Coq 관련 강의 (공부) 자료들은 초반에 functional programming language에 초점을 둔 것이 많았었습니다. 아니면 최소한 Coq에서 type을 어떻게 만들어 주는지에 대하여 설명을 하고 이후에 다른 내용을 다루는 것 들이 많았었습니다. 하지만, 이 논문에 대한 발표를 들어본 결과, 그리고 논문을 본 결과 logic을 가르치는 데 있어서 기존의 방법과 비교해 보았을 때 보다 적은 overhead를 가지고 Coq을 배울 수 있는 방법 또한 존재한다는 것을 알게 되었습니다. 이 학교의 강의는 초반부터 Coq에 대한 자세한 설명 없이 traditional logic에 대한 강의로 시작합니다. 이로부터 propositional logic, predicate logic, formal induction, modal logic, 그리고 마지막으로 hoare logic에 이르기까지 각각의 topic 당 1~2주씩을 할애하여 강의를 합니다. 또한, Coq을 배우는 데 있어서의 overhead를 줄이기 위하여 각각의 강의 주제에 필요한 Coq 정의들을 몇 가지 해 놓은 것을 볼 수 있었습니다. 특히, traditional logic과 modal logic을 강의하는 데 있어서 몇 가지 필요한 정의들을 해 놓은 것을 볼 수가 있었고, 이를 논문에서도 언급해 주고 있습니다. 어떻게 보면 Coq을 logic 수업을 위한 도구로서 사용하는데 특화 시켰다고 생각할 수도 있지만, 이러한 수업들을 통해서 Coq에서의 증명의 흐름 또한 익힐 수 있고, 학생들에게 Coq에 대한 거부감 또한 줄여주며 Coq을 익히게 할 수 있는 방법이 아닐까 하는 생각을 하게 되었습니다.

아무리 Coq을 배우는 데 있어서의 overhead를 줄였다고 하더라도, 일반적인 logic 수업보다 다루는 양이 많아진 것은 당연한 일 이라고 생각을 합니다. 하지만 발표와 논문에서 언급한 학생들의 feedback 중 인상이 깊었던 것 중 하나가 Coq을 처음 배울 때는 힘이 들었지만, 이후에는 손으로 증명하는 것 보다는 오히려 좋은 점이 있었다고 하는 부분이 있었고, 저도 이런 logic 수업에 있어서 Coq을 사용하는 것은 학생들이 증명의 흐름을 파악하거나, 자신이 한 증명을 다시 따라가는 데 있어 많은 도움을 주지 않을까 하는 생각을 합니다.

Automatically Verifying Typing Constraints for a Data Processing Language.

CPP Session 6: Programming Languages 에서 다뤘던 논문입니다.

이 논문은 type-checker와 동일한 성질들은 체크해 주는 generic한 (범용적인) 검증 도구를 만드는 것을 목표로 하고 있었으며, 이러한 연구를 통하여 범용적인 검증 도구를 위한 연구들에 도움을 줄 수도 있다고 하였습니다. 이 논문에서 한 일을 더 자세히 살펴보면 이 논문에서는 M이라는 MS Research에서 만든 data processing language의 subset 언어인 DMinor 라는 언어를 정의하고, 이 언어에 대한 대한 검증 도구인 DVerify를 만들었으며 이러한 자신들이 제안한 방식과 type checker 를 사용한 방식 (DMinor 방식이라고 이야기 하였었습니다.) 의 차이점을 Z3 SMT Solver로 서로 비교해 주었습니다.

DVerify는 DMinor를 translation 해 주는 부분과, Boogie라는 기존에 있었던 범용적인 프로그램 검증 도구를 사용하는 파트로 나뉘어 져 있다고 하였으며, 또한 translation을 효과적으로 하고, translation이 정확하다는 것을 증명하기 쉽게 하기 위하여 Bemol 이라는 Intermediate Verification Language (IVL - 검증용 중간 언어) 를 만들어 주었다고 하였습니다. 발표에서는 이 언어들의 syntax나 semantics, 혹은 translation technique을 보여주기 translation에 대한 몇 가지 예시들 (Factorial 프로그램 같은 것들) 을 보여주었습니다.

그리고, 발표의 마지막 부분에서 발표자는 DMinor 언어에 대하여 DVerify (이 논문에서 제안한 방식) 를 이용하여 검사한 방식과, 기존의 type checking 방식으로 검사한 방식 두 가지에 대한 비교를 보여주었습니다.. 정확도에 대한 결과에서는 DVerify 방식이 조금 더 정확 (precise) 하다고 하였습니다. 또한 성능에 대한 비교에서는 DVerify 방식이 프로그램 검사에 걸린 전체 시간으로 놓고 봤을 때는 DMinor 방식 (type checking 방식) 보다 상당히 시간이 오래 걸리지만 이렇게 시간이 오래 걸리는 이유는 initialization (parsing 과 .NET framework을 시작하는 데 걸리는 시간) 때문이라고 하였습니다. 그리고 다른 비교를 통하여 이러한 initialization time을 제외하고는 DVerify 방식과 DMinor 방식에 대한 평균 소모 시간은 비슷했다고 합니다. 하지만 두 번째 비교에서 initialization time을 포함시키지 않아도 되는 이유에 관해서는 자세히 설명을 안 했던 것 같습니다.

이 발표를 들은 후 제가 가지게 된 이 발표에 대한 의문은 크게 두 가지입니다. 첫 번째는, 이것이 과연 범용적인 프로그램 검증 도구를 위한 연구인가 하는 의문입니다. 결국 type checker 만들지 않는다고 하더라도 검증용 중간 언어 (IVL) 로 검사를 하고자 하는 프로그램에 대한 프로그래밍 언어 (여기서는 DMinor) 를 잘 표현해 주기 위한 방법이 필요합니다. 결국 검증하고자 하는 각각의 프로그래밍 언어를 translation 해야 하며 이와 관련하여 translation이 정확하다는 증명 또한 해 주어야 할 것 같습니다. 그러면 이러한 것이 과연 type checker를 만드는 것에 비해서 들어가는 cost가 적을지는 생각해 봐야 할 문제라 봅니다.

두 번째로 DVerify가 type checker를 사용한 방식보다 사용자를 위한 측면에 있어서 불편한 점이 하나 있다고 느꼈습니다. 이 논문에 대한 발표 초반에서 자신들의 방식은 type checker를 사용하는 것과 같이 프로그램이 "correct 하다" 혹은 "incorrect 하다"는 것 까지는 알려주지만 type checker를 사용하는 것과 다르게 프로그램이 incorrect할 경우 이에 대한 counter example을 알려주는 기능은 좀 떨어진다고 하였던 것 같습니다. 이러한 것은 결국 프로그래머들에게 프로그램이 왜 잘못됐는지를 직접 찾도록 하는 것이 되 버리기 때문에 만약 이 것이 맞다면 이러한 점은 DVerify가 가지고 있는 커다란 단점일 것이라는 생각을 하였습니다.

Certified Security Proofs of Cryptographic Protocols in the Computational Model an Application to Intrusion Resilience

CPP Session 6: Miscellaneous 에서 다뤘던 논문입니다.

이 논문은 security protocol (특히 session key 생성과 같이 인증과 연관이 많이 되어 있는 security protocol들) 들에 대한 공격 모델들을 확률적인 측면까지 고려하여 만든 다음 이러한 공격에 있어서 security protocol들이 안전한가 안전하지 않은가에 대한 증명을 mechanize를 했다는 것이 주 내용 이었습니다. 이 논문 발표자는 자신들의 연구가 electronic voting 등과 같이 여러 커다란 security 적으로 민감한 프로그램 (혹은 시스템)에 접목될 수 있다고 했습니다.또한 이 논문에서는 현실적인 공격 모델을 만들기 위하여 하나의 security protocol을 black box와 같이 간주를 하고 공격자가 input과 output만을 유추해 낼 수 있다는 가정을 하였다고 들었습니다.

이러한 연구를 하게 된 배경은 자신과 같은 접근을 통해서 security protocol에 대한 안전성을 증명하려고 하는 paper-and-pencil 을 이용한 연구들은 많이 있지만 증명 자체가 매우 복잡하고, 이에 대한 검증이 쉽게 이루어 질 수 없다는 단점이 있기 때문에 이를 mechanize하려고 하였다고 하였습니다. 이러한 측면에서 봤을 때 proof mechanization을 하는 이유들 중 가장 기본적인 이유에 충실한 연구였다는 생각이 듭니다.

이러한 security protocol의 안전성과 관련한 증명들을 Coq으로 mechanize 하기 위해서 이 논문은 크게 두 가지를 하였다고 말 했습니다. 첫 번째는 Coq에서 CIL (Computational Indistinguishability Logic) library 를 만들어 주었다 (기존에 있는 것을 확장한 것인지 아니면 자신들이 직접 새로 이 library를 만들어 주었는지는 더 알아봐야 할 것 같습니다.) 고 하였고 두 번째는 이 논문에서 모델링 한 security protocol에 특화된 Coq mechanization part라고 하였습니다. 이 두 가지 부분들 중에서 아무래도 CIL 부분이 범용적으로 사용할 수 있고 다른 관련된 연구들에 있어서도 재사용이 될 수 있기 때문에 조금 더 관심을 기울이게 되었는데, 이는 공격자가 security protocol과 관련 된 정보 등을 얻는 데 있어서 확률적인 측면을 고려할 수 있도록 해 주는것을 도와주는 Coq library라고 하였으며, 이를 통하여 현실적인 공격 모델을 만들 수 있다고 하였던 것으로 기억합니다.

> 이러한 수업들을 통해서 Coq에서의 증명의 흐름 또한 익힐 수 있고, 학생들에게 Coq에 대한 거부감 또한 줄여주며 Coq을 익히게 할 수 있는 방법이 아닐까 하는 생각을 하게 되었습니다.

1년간 영국에서 공부를 하다

글쓴이 이욱세
한양대학교 컴퓨터공학과 조교수
방문지 런던, 영국
방문기간 05/07/2010-05/08/2011*

한양대학교 교수가 된지 6년, 학과 교수님들의 도움으로 지연되지 않고 7년차에 연구년 승인을 받아 영국으로 향할 수 있었다. 연구년을 런던으로 가기로 결정한 것은 2007년, 그 때 Peter O'Hearn과 양홍석 박사가 있는 Queen Mary, University of London에 5주간 단기 방문을 했었고 나름 의미있는 성과[1]를 거두었기에 다시 돌아오겠다고 약속했었다. 사실 국내파인 나로써는 첫 외국 장기거주라 두려움도 많았다. 연구에 관해서는 양홍석 박사가 있기 때문에 걱정 없지만, 가족을 모두 데리고 낯선 곳에서 1년 넘게 거주해야 하기 때문이었다. 하지만 2007년에 5주간 그때도 가족을 모두 데리고 런던에 거주한 경험이 있었기에 그리 힘들지 않게 즐겁게 1년을 보내고 왔다고 생각된다.

1 연구환경

퀸매리 대학 (Queen Mary, University of London) 시설 자체는 우리의 여느 대학들보다 우수하다고 할 수 없다. 런던의 동부 지역, 중심은 아니지만 중심에 가까운 곳[2]에 위치하고 있고, 학생들은 런던의 특성상 다양한 인종으로 구성 되어 있다. 특히 인도, 파키스탄, 방글라데시 이주민들이 동부에 많아 퀸매리 대학에도 인도 계열 학생들이 많다. 건물 자체는 그리 오래 되지 않지만 공간 자체는 크지 않다. 특이한 것은 가구가 별로 없고, 개인 물품을 연구실에 잘 두지 않는다. 심지어 대부분 연구원 또는 학생들은 PC를 놓고 사용하는 것이 아니라 노트북을 들고 다니면서 공부를 한다. 필자 또한 한국에서 가져간 맥북만이 유일한 연구장비였다. 늦게까지 연구하는 이가 별로 없다. 양홍석 박사 말로는 학교가 있는 동네가 그렇게 안전한 곳이 아니니까 저녁식사 전에 퇴근하는 것이 좋다고 한다.

퀸매리 전산과 내에 이론 그룹(theory group)이 있어 연구가 활성화되어있다. 세미나도 함께 하고 식사도 함께 한다. 우리 대학들과는 달리 대학원생 수는 많지 않고 대부분 구성원이 교수 아니면 포닥, 즉, 박사후 연구원이다. 아무래도 Peter O'Hearn, Dino Distefano, 양홍석, 그리고 겸직교수인 Byron Cook 까지 분리논리와 프로그램 검증 쪽의 교수가 많아 이 주제와 관련된 연구원들이 많다. 특히 1년 거주하면서 친하게 지낸 친구는 Noam Rinetzky라고 이스라엘 Tel-Aviv University의 Mooly Sagiv 교수의 박사 졸업생으로, 포인터의 의미구조론을 연구하고 있었다.

퀸매리 대학 출신의, 특히 Peter O'Hearn과 함께 작업을 한 이름 있는 연구자들이 많다. Peter O'Hearn 교수는 분리 논리(separation logic)를 Carnegie Mellon University의 John Reynolds와 함께 만들고, 이를 실제 프로그램 검증 에 적용한 입지전적인 인물이다. 최근에 University College London으로 이직 하였다. 양홍석 (Hongseok Yang) 박사는 분리 논리를 완성하는데 큰 공헌을 하였으며 분리 논리를 기반으로 하는 모양 분석기인 SpaceInvader 개발에 주 역할을 한 학자이다. 최근에 옥스포드 대학 (Oxford University)로 이직하였다. Cristiano Calcagno와 Dino Distefano 또한 SpaceInvader 개발을 하였으며 현재 각각 Imperial College, Queen Mary University of London의 교수이다. Josh Berdine과 Byron Cook은 종료분석기 SLAyear/TERMINATOR를 개발한 연구자로, Josh는 퀸매리 출신이고 현재 Microsoft Research at Cambridge에 재직 중이다. Byron은 Microsoft Research at Cambridge 연구원이고 퀸매리 대학 겸직교수를 하고 있다.

---

* 날짜 표기법이 영국식이다 (dd/mm/yyyy)
1 Scalable Shape Analysis for Systems Code. Hongseok Yang, Oukseh Lee, Josh Berdine, Cristiano Calcagno, Byron Cook, Dino Distefano, and Peter O'Hearn. International Conference on Computer Aided Verification (CAV 2008), pp. 385-398, July 2008.
2 런던에서는 시내 중심부터 Zone 1-6로 나누는데 퀸매리는 Zone 2에 있다.

퀸매리 멤버들 뿐만 아니라 Cambridge University, Imperial College, Microsoft Research at Cambridge의 관련 연구원들은 비정기적으로 Yak이라 불리는 세미나를 열어 새로운 연구 주제들에 대해 의견을 나눈다. 내가 있을 때는 Imperial College와 퀸매리 대학에서 3-4번 정도 세미나를 했었다. 세미나 스타일은 자유로운데, 슬라이드를 특별히 만들지 않고 화이트보드에 써가면서 발표를 한다. 나는 발표하지 않았지만 양홍석 박사와 함께 한 세 번째 연구주제를 양박사가 발표하였다.

연구 분위기 자체는 한국과 별로 다르지 않다. 물론 한국에서도 활성화된 그룹과 활성화되지 않은 그룹이 다르겠지만, 연구실 내의 연구원들이 토론하고 조용히 공부하고, 가끔씩 SIGPL 학회나 ERC 워크샵을 통해 외부와 정보를 교류하고 별반 다를 것은 없다. 아마도 우리 선배 교수님들이 한국에서 열심히 노력해서 만들어 놓은 토양 위에 내가 있었기에 영국에 와서도 특히 부러울만 한 것을 찾지 못하는 것인지도 모르겠다. 단지 차이점을 찾으라고 한다면, 조금 이름 있는 연구자들이 주변에 많이 있다는 것, 토론을 더 즐긴다는 것, 국적이 다양하다는 것, 박사후연구원이 많다는 것 정도이다.

2 연구주제

2.1 겹쳐진 자료구조

이야기의 시작은 양홍석 박사의 해결되지 않은 고민이다. 내가 영국에 가기 2년 전 쯤에 양홍석 박사는 Microsoft로부터 희한한 자료구조를 가진 디바이스 드라이버 코드를 입수하였다. 문제의 자료구조는 겹쳐진 자료구조이다. 예를 들어, 리스트와 트리가 겹쳐져 있는데, 즉, 셀에 리스트에 대한 필드와 트리에 대한 필드가 함께 있는 것이다. 이러한 자료구조는 두 가지 이상의 방법으로 셀에 접근하고자 할 때 쓰이는데 Microsoft 윈도우즈 뿐만 아니라 Linux에서도 폭넓게 쓰이고 있다. 포닥으로 있던 Rasmus Petersen과 함께 양홍석 박사는 코드를 분석하여 이러한 겹쳐진 자료구조에 대한 검증 방법론을 찾으려고 노력 하였다. 최종적으로 두 자료구조를 분리해서 분석하자는 아이디어를 사용하여 구현하였으나 성능이 나오지 않았고 일부 예제는 제대로 검증되지 않았다.

영국에 가자마자 이 주제를 진행하였다. 처음에는 양홍석 박사의 방법론에 동의하지 않았으나 토론을 통해 방법론이 먹힐 수 있다고 믿게 되었다. 문제는 두 겹친 자료를 분리했을 때 두 자료 간의 관계를 잃는 것이었다. 그렇다고 두 자료 간의 관계를 정확히 유지하면서 분석하기에는 두 자료의 관계가 약하다. 예를 들어, LIFO 큐와 우선순위 큐가 겹쳐있는 경우 LIFO 큐의 순서와 우선순위 큐의 순서는 아무런 관련이 없다. 하지만 그 두 자료구조가 동일한 메모리 영역을 차지하고 있다는 것은 알고 있어야 한다. 왜냐하면 코드에서 LIFO 큐에서 셀을 꺼낸 후 우선순위 큐에 관련된 필드를 수정하는데 그 관계를 모르면 엉뚱한 셀을 수정한다고 분석할 수도 있기 때문이다.

문제의 핵심을 영역(region)을 표시하는 것으로 해결하였다. 두 자료구조를 따로 분석하되 어느 영역에 속하는지 별도로 표시해 두면 LIFO 큐에서 꺼낸 셀이 다른 영역이 아닌 우선순위 큐 내에 있는 셀임을 알 수 있는 것이다. 물론 이 아이디어를 구현하기 위해서는 영역 유추가 필요하였고 여러가지

$$\frac{\Sigma_0\ ,\ \Sigma_1\ ,\ \mathsf{ext}(\epsilon, f_0, e_1, f) \rightsquigarrow \Sigma\ ,\ \epsilon'\ ,\ \delta_0\ ,\ \delta_1}{(\mathsf{ls}\, k\, e_0\, f_0) * \Sigma_0\ ,\ \Sigma_1\ ,\ \epsilon \rightsquigarrow (\mathsf{ls}\, \mathsf{PE}\, e\, f) * \Sigma\ ,\ \epsilon'\ ,\ \delta_0 \cup (e_0, f_0)\ ,\ \delta_1}\ \text{PE-left}$$
$$(\text{when}\ (e_0, e_1, e) \in (\epsilon \cup \mathsf{EQ}) \wedge e_1 \notin \mathsf{MayAlloc}(\Sigma_1) \wedge \mathsf{comb}_\epsilon(f_0, e_1) = f)$$

$$\frac{\Sigma_0\ ,\ \Sigma_1\ ,\ \mathsf{ext}(\epsilon, e_0, f_1, f) \rightsquigarrow \Sigma\ ,\ \epsilon'\ ,\ \delta_0\ ,\ \delta_1}{\Sigma_0\ ,\ (\mathsf{ls}\, k\, e_1\, f_1) * \Sigma_1\ ,\ \epsilon \rightsquigarrow (\mathsf{ls}\, \mathsf{PE}\, e\, f) * \Sigma\ ,\ \epsilon'\ ,\ \delta_0\ ,\ \delta_1 \cup (e_1, f_1)}\ \text{PE-right}$$
$$(\text{when}\ (e_0, e_1, e) \in (\epsilon \cup \mathsf{EQ}) \wedge e_0 \notin \mathsf{MayAlloc}(\Sigma_0) \wedge \mathsf{comb}_\epsilon(e_0, f_1) = f)$$

기법들을 활용하여 그럴듯하게 영역을 유추하는 알고리즘을 개발하였다. 사실은 이러한 대충 유추하는 알고리즘을 개발하는 것이 내 특기인 듯하다. 결과적으로 분석기의 성능도 문제가 없었고 예전에 검증하지 못하던 예제도 검증할 수 있었다.

이 결과는 예전의 내 경험과는 달리 매우 쉽게 CAV 학회에 발표되었고[3], 거기서 우수논문으로 선정되어 저널에도 출판될 것이다. 이렇게 쉽게 된 이유는 양홍석 박사가 곳곳에 돌아다니면 홍보를 했기 때문이라고 생각된다. CAV 에 논문을 제출하기도 전에 양홍석 박사는 주변에 최소 4곳에 가서 이 주제로 발표를 하였다. (그렇기 때문에 본 발표는 열심히 홍보하지 않는 내가 했다.) 분명히 CAV 심사위원 중에 2-3 명은 이미 양홍석 박사의 발표를 들었을 것이다. 이미 내용을 알고 심사를 하기 때문에 논문을 잘못 이해할 가능성도 없고, 논문의 세세한 내용으로 부정적인 판정을 받을 확률이 적어지는 것이다.

잡설: 원래는 영국에 가면 주변 유럽국가에서 열리는 학회에 많이 참석할 생각이었다. 한국에서 파리에서 개최되는 학회를 가는 것은 경제적으로나 시간적으로나 그리 만만하게 볼 일이 아니지만, 런던에서 파리는 서울에서 포항간다고 생각할 수준이니까. 하지만 불행히도 유럽에서 열리는 학회에 전혀 참석하지 못했다. 학회가 많이 열리지도 않았지만 열리는 학회에 논문을 내야 참석할 가치가 높아지니까 말이다. 그런데 이 논문이 미국 유타(Utah)주에서 열리는 CAV 에 발표되었다. 이런, 유타주로의 여행은 한국에서 출발하는 것과 영국에서 출발하는 것이 경제적으로나 시간적으로나 별 차이가 없었다. 그것도 귀국하기 직전이라서 귀국일정에도 영향을 미쳤다. 어쨌든 CAV는 처음 가 보았는데 좋은 학회였고 본래의 하드웨어 검증 주제보다 소프트웨어 검증 주제가 증가하고 있음을 느낄 수 있었다.

2.2 끊긴 포인터 문제

끊긴 포인터 문제(cut-pointer problem)은 모양 분석기에서 잘 알려진 해결되지 않은 문제이다. 이 문제를 제기한 사람은 Noam Rinetzky로, 프로시저를 분석할 때 발생한다. 프로시저의 인수로 포인터가 넘어가면 분석시 포인터가 가리키는 자료구조를 넘겨야 한다. 문제는 다른 지역 포인터가 그 자료구조 내의 셀을 가리키게 될 때 발생한다. 지역변수에 저장된 포인터이기 때문에 호출된 프로시저에서는 그 포인터를 알 수가 없다. 호출된 프로시저가 끊긴 포인터를 무시하고 분석하면 프로시저가 끝났을 때 끊긴 포인터가 어느 셀을 가리키는지 정보를 잃을 수 있다. 그렇기 때문에 현재의 모양 분석기는 끊긴 포인터를 호출된 프로시저에서 지역변수처럼 다루게 하고 있다. 문제는 호출이 반복되면 끊긴 포인터가 자꾸 늘어나게 되는데, 재귀호출의 경우에는 이론상 무한히 늘어날 수 있어 분석이 불가능하게 된다.

이 문제를 완화하는 방안에 대해서 여럿이 모여서 토론을 했다. 참가자는 문제의 창시자 Noam Rinetzky, 나, 양홍석 박사, 그리고 ENS[4]의 Xavier Rival 이다. 우리가 파리의 ENS에 가서 이틀간 토론을 했는데, 두 가지 방법론을 도출하였다. (1) 넘기는 자료구조를 줄이면 문제가 완화될 것이다. 현재는 포인터가 인수로 전달되면 그 포인터에서 도달 가능한 모든 자료구조를 넘긴다. 사실 도달 가능한 자료구조가 모두 사용되지 않을 수도 있지만 어느 셀들이 사용될 지 모르기 때문에 모두 넘긴다. 꼭 사용되는 자료구조만 넘기면 끊긴 포인터의 수가 줄어들 가능성이 높다. (2) 끊긴 포인터에 대한 정보를 줄이면 완화될 것이다. 현재는 끊긴 포인터가 가리키는 셀을 정확히 추적하고 있다. 그러나 많은 경우 정확한 셀을 알 필요가 없다. 끊긴 포인터가 특정 자료구조의 아무 셀이나 가리킨다고 해도 검증에는 문제가 없다는 것이다. 이러면 끊긴 포인터의 수는 줄어들지 않지만 끊긴 포인터로 인한 부담이 줄어드는 것이다.

불행히도 이 토론은 일회성으로 끝났고 찾아낸 방법론 중에 1번 방법론을 필자 독자적으로 연구하고 있다. 프로시저가 사용하는 셀을 계산할 수 있어야 하는데 접근경로에 기반한 방법으로 분석기를 고안하였으나 그 알고리즘의 복잡도나 정확성이 아직 만족할 수준에 도달하지 못했다. 양홍석 박사와 영국에서 계속 공동 연구를 하려고 했으나, 가능한 영국에서 연구하여 도움이 되는 주제를 하기 위해서 중단하고, 이 주제는 한국으로 가지고 왔다.

---

3 Program Analysis for Overlaid Data Structures. Oukseh Lee, Hongseok Yang, and Rasmus Petersen. International Conference on Computer Aided Verification, July 2011
4 Ecole Normale Supérieure, Paris, France

잡설: 영국과 프랑스는 이제 한 나라나 다름이 없다. 2011년 1월 위 토론을 위해서 ENS를 방문하였다. 유로스타(eurostar)라는 런던과 유럽대륙 사이에 해저터널을 통과하는 고속철도로 인해 아침에 출발해서 점심식사 전에 ENS 에 도착할 수 있었다. 물론 중간에 국경을 넘을 때 입국심사가 있기는 하지만, 비행기에 비해서 훨씬 간단하게 빠른 시간에 런던과 파리를 오갈 수 있었다. 한국에서 서울과 포항을 오가듯이.

2.3 동시성 논리의 자동 검증기

분리논리에 대한 검증기가 어느 정도 익어 가니 파생되는 주제 중에 하나를 진행하였다. 예전에 포인터에 대해 전혀 정확하게 분석하지 못할 때 포기했던 문제들이 모양분석기가 나오면서 다시 재도전되고 있다. 대표적인 것이 프로그램이 종료하는지 예측하는 분석기이다. 종료 여부를 판별하는 휴리스틱에 대해서는 이론들이 쌓여 있었지만 실제 코드에 적용할 수 없는 이유가 포인터였다. 이게 어느 정도 해결되니까 한동안 종료 분석 붐이 일었었다. 다른 하나는 동시성 분석이다. 포인터로 인해 시스템 코드의 동시성을 정확하게 검증할 수 없었는데, 분리논리가 나와서 최소한 손으로는 동시성을 가진 작은 시스템 코드를 검증할 수 있게 되었다. 양홍석 박사와 나의 목적은 손으로 하는 것이 아니라 분석기를 돌려 자동으로 검증하는 것이다. 동시성 논리의 하나인 금지논리(deny logic)를 개발한 연구자들과 함께 진행하였다. 양홍석 박사도 동시성 논리 중 금지논리를 정확하게 이해하고 있지 않다고 스스로 생각했기 때문에, 자동 검증기를 개발하기 위해 이해가 필요하 다고 생각했다. 그래서 금지논리 고안자인 Microsoft Research at Cambridge의 Matthew Parkinson, Cambridge University의 Mike Dodds, 그리고 금지논리를 이용해 시스템코드를 증명해 본 경험이 풍부한 Cambridge University의 Alexey Gotsman과 함께 2-3번 정도의 세미나를 했다. 모두들 자동 증명에는 큰 관심이 없어서 아쉬웠지만, 세미나를 통해 금지논리의 의미와 금지논리를 사용하여 손으로 증명하는 핵심 기술들을 쉽게 전수받을 수 있었다. 아마도 이런 것이 영국에 왔기 때문에 가능한 일이 아닐까.

금지논리가 분리논리와 자동화 입장에서 다른 점은 방향성이었다. 분리논리는 프로그램을 앞에서 뒤로 또는 반대방향으로, 어쨌든 한방향으로 훑으면서 증명을 할 수 있다. 실제로는 오류가 나지 않기 위한 전조건과 프로그램의 실행 결과를 말하는 후조건을 유추하는 방향은 다르지만, 대부분의 경우는 한방향으로 유추해도 큰 문제가 없다. 하지만 금지논리는 특정 연산이 금지된 경우와 금지되지 않은 경우 프로그램의 증명 과정이 확연히 달랐다. A라는 연산이 금지된 경우, 프로그램을 실행하면서 도중에 다른 쓰레드가 A라는 연산을 하지 못한다는 것이 보장된다. 그렇지 않은 경우 다른 쓰레드가 아무때나 A라는 연산을 할 수 있기 때문에 중간중간 그 효과를 고려해서 증명해야 한다. 그런데 자동 증명기에서는 A의 존재여부가 프로그램 중간에서 유추될 수 있다. 이 경우 앞에 증명된 것들이 무효화되기 때문에 프로그램을 왔다갔다하면서 유추해야 한다. 이 아이디어를 구현하기 위해 분리논리 자동검증기의 방향성을 우선 양방향으로 바꾸어야 했다. 그런데 놀랍게도 방향성을 바꾸니 동시성 논리에 적용하기도 전에 분리논리 자동검증기 자체의 정확도가 개선되었다.

결과적으로 현재까지는 동시성 논리가 아닌 분리논리에 대한 검증기의 방향성을 개선하는 방법론을 찾았다. Abductor[5]에서 사용한 기술이 단방향 기술인데 일부 예제에서 전조건을 못 찾아내는 경우가 있다고 한다. 새로 고안한 양방향 분리논리 검증기를 사용하면 그러한 경우를 찾을 수 있다. 이유는 단방향보다 양방향으로 했을 때 전조건이 더 정교하게 요약되어 계산되기 때문이다.

5 Compositional Shape Analysis by means of Bi-abduction. Cristiano Calcagno, Dino Dis- tefano, Peter O'Hearn, and Hongseok Yang. ACM Symposium on Principles of Programming Languages (POPL 2009), pp. 289-300, January 2009.

3 런던라이프

사실 런던 라이프를 즐기지는 못했다. 런던을 관광으로 오는 사람들조차 런던에서 유명한 뮤지컬을 어떻게서든 보고 가고, 박지성이 나오는 축구경기를 보고 가곤 하는데 (그 짧은 일정에도), 나는 뮤지컬도 축구경기도 보지 못했다. 사실상 스스로 주 5일 근무에 9시에서 5시까지 성실히 연구하고, 휴가철[6]에는 그래도 가족을 위해 여행을 한다고 영국 밖으로 나갔으니 런던의 유흥문화를 즐길 수가 없었다. 대신 다른 것을 즐겼으니까 그것으로 되었다.

런던 이야기를 하면 대부분의 사람들이 날씨가 나쁘고 음식이 맛없다고한다. 응? 날씨 좋은데, 음식도 맛있는데, 왜 나쁘다고 할까. 날씨 이야기부터 하자. 비는 자주 온다. 하지만 그 비라는 것이 우산을 쓰지 않아도 될 정도의 비만 자주 온다. 한국에서는 비오면 우산 없이는 밖에 나갈 수 없지만 런던에서는 우산 쓰는 사람을 보기 힘들다. 비오면 공기가 깔끔해지고 온도가 서늘해져서 오히려 쾌적하다. 단지 런던의 겨울은 좋지 않다. 위도가 높기 때문에 일찍 해가 진다. 오후 세시쯤이면 어둑해진다. 거기다 습한 추위가 있어서 여성 동지들은 매우 싫어한다. 칠리(chilly)하다는 말이 무슨 말인지 나도 제대로 느끼고 왔으니까. 런던을 관광으로 간다면 겨울은 추천하지 않는다.

영국 음식은 훌륭하다. 프랑스 사람들은 소스 맛으로 먹지만, 영국 사람들은 재료 맛 그대로 먹는다. 전혀 장난을 치지 않는다. 물론 영국의 저렴한 식당에서는 조미료로 장난을 치는 경우도 간혹 느꼈지만 대부분의 경우는 재료 그대로 내준다. 잠깐 미국에 갔을 때 음식 때문에 좀 고생을 했다. 이것들은 음식에 왜 그렇게 장난을 치는지. 영국에도 유입 인구가 많아서 인도, 터키 식당이 많고, 커리와 터키식 소스를 즐길 수 있다. 중국집은 유럽에서 함부로 가지 않는 것이 좋다. 내가 런던에서 자주 가던 저렴하면서 맛좋은 중국식당을 제외하고는 유럽에서 맛있는 중국집을 찾기 어렵다. 사실 빵, 치즈, 커피도 많이 기대하고 갔는데 좋은 것을 구하기가 어려웠다. 포기하고 있었는데 Borough Market이라고 우리의 재래시장 같은 곳에서 구할 수 있었다. 소영업자들이 자기들이 직접 생산한 음식 재료들을 파는데, 그 곳에서 산 신빵(sour bread), 치즈, 커피는 아직도 너무나 그립다. 우리의 재래시장도 이렇게 질 좋은 상품을 파는 쪽으로 진화해야 하지 않을까.

영국 맥주는 훌륭하다. 사실 영국이라고 하면 사람들은 위스키를 떠올린다. 그런데 의외로 맛있는 맥주가 많아서 거의 매일 혼자 또는 와이프랑 맥주를 마셨다. 참고로 필자는 한국에서는 맥주는 맛 없어서 안 마신다. 나중에 조사해 보니 영국의 주 종족인 앵글로-색슨 족은 원래 독일지방에서 온 게르만 족이다. 게르만 족은 맥주를 주로 마시기 때문에 당연히 영국 사람들도 맥주를 무척 사랑한다. 각종 지역에서 나온 맥주들이 다양한 맛을 낸다. 우리나라도 쌀부족 시대를 거치지 않았다면 가가호호 맛있는 청주를 팔았을 터인데 매우 아쉽다.

런던의 물가는 살인적이다. 물가를 쉽게 계산하는 방법이 화폐 단위이다. 1 파운드(영국화폐) = 1달러 = 1000원. 맞는 공식이 아니지만 영국에서 1파운 드는 우리나라의 1000원으로 환산하면 살 만하다. 그런데 실제로는 1파운드는 1800원 정도이니, 물가는 우리나라의 두 배라고 생각하면 된다. 지하철 한 번 타면 1.8파운드, 학교식당이 5파운드, 외부 식당에서 먹으면 기본이 10파운드이 다. 처음에 물가 적응을 못해서 제대로 먹지도 못했는데 1파운드를 1000원으로 생각하면서 편해졌다. 그런데 재미있는 것은 차(tea)와 커피는 비싸지 않다. 어디를 가도 1.5파운드 정도면 커피 또는 차를 마실 수 있다. 한화로 환산하면 그리 싼 것은 아니지만 런던 물가에 비하면 상대적으로 한국보다 반 값으로 느껴졌다. 차를 사랑하는 영국인들이기 때문에 차 값은 올릴 수가 없었나보다.

영국은 세금을 많이 내고 대신 공공서비스를 많이 받을 수 있다. 주민세만 내면 병원이 공짜다. 아이들을 데리고 갔기 때문에 병원이 제일 걱정이었는데-알겠지만 미국에서는 아파도 돈이 없으면 병원에 갈 수 없다-영국에서는 진료가 공짜기 때문에 진료가 끝나면 그냥 병원을 나온다. 단지 진료를 받기 위해서는 48시간 전에 예약을 해야 하고, 시설이나 의사의 질이 좀 떨어진다는 단점이 있지만, 아들의 팔꿈치가 한 번 빠진 적이 있는데, 진단하고, 엑스레이 찍고, 치료받는 것이 모두 무료로 제공되었다. 게다가 딸 아이는 만5세가 되는 나이에 갔기 때문에 공립학교를 무료로 다녔다. 영국은 만5세에 의무교육이 시작된다. 갔다오고 나서 검토를 해보니 학교에서 배운 내용도 매우 훌륭했고 교사들도 수준이 매우 높았다고 생각이 된다. 물론 딸 아이도 처음 1달을 제외하고는 매우 행복하게 학교를 다녔다.

4 후기

1년여의 삶을 짧은 글에 담으려 하니 두서가 없다. 특별히 어떤 메시지를 전달하려고 하는 것이 아니니 독자들은 스스로 의미를 찾기 바란다. 그리고, 사진이 한 장도 없어 죄송하게 생각한다. 사진은 많이 찍었지만 모두 아이들 사진이고, 막상 글에 사진을 넣으려고 찾아 보니 퀸매리 대학도 연구실도 심지어 나중에 1주일에 한 번씩 방문한 옥스퍼드 대학 사진도 한 장도 없다는 것을 알았다.

6 휴가철에는 일상생활에 지장을 줄 정도로 모든 것이 정지된다. 휴가철은 4월말, 8월말, 12월말인데, 특히 크리스마스 휴가철에는 모든 서비스가 정지된다. 퀸매리 대학이 크리스마스 2주간 폐쇄되고, 슈퍼마켓조차 크리스마스 당일에는 문을 닫으니까 할 말이 없다.

전산학에서 배운 인생의 교훈들

글쓴이 황승원
POSTECH 컴퓨터공학과 조교수

ROSAEC 센터에 참여한 지난 3년여 간을 돌아보니 무결점 소프트웨어라는 주제하에 많은 분야의 지식이 (비전공자를 겸손해지게 만드는 람다부터 도구제작에 이르는) 다양한 추상화 레벨에서 논의되었다. 대학원 이후로는 전공분야 밖의 연구 내용은 접할 기회가 많지 않아서, 워크샵 때마다, 전산학의 다양한 분야를 처음 접하던 호기심 천국 학부 시절로 회춘하는 기분이 들었다. 회춘한 김에, 이 지면에서는 람다보다도 추상적인 세계인 인생에 대해, 다양한 전산학 과목을 들으며 생각하게 되었던 내용을 (에세이니만큼 약간 농담을 섞어) 나누어 보려고 한다.

성능은 점근적으로 분석되어야 한다. 전산학에서는 알고리즘 A와 B의 성능을 비교할 때, 입력의 크기 n이 특정 값일때 A의 비용이 낮다고 해서 A가 더 우수하다고 결론짓지 않고, n이 무한대로 접근할 때의 성능을 비교한다. 내 인생에 주어진 시간을 입력으로 생각하고 인생 성능에 대해 생각해본다면, 특정시간 t에서 스스로나 타인의 가치를 섣불리 판단하고 있지는 않은지, t가 커져갈 때의 인생을 위해 해야 할 일들에 대해서도 생각해보게 된다. 또 한가지는, 컴퓨터 c대를 도입하는 산술적 향상은, 점근 분석에서는 상수치로 무시될 뿐이라는 것이다. 쉽고 근시안적이고 산술적인 향상 대신, Big Theta를 바꾸는 향상을 추구할 수 있을까?

대인관계는 게임 이론을 참조한다. 대인관계에서 어떤 경우에 상대와 협력해야 하며, 어떤 경우에 협력하지 않는 것이 좋을까? 대체로 협력에는 어느 정도 비용이 들지만, 상대도 그 비용을 감수하며 신의를 지킨다면, 비용보다 큰 시너지를 낼 수 있을 것이다. 반면, 협력하고자 하는데 상대는 비용을 감수하지 않고 배신한다면, 손해가 막심할 것이다. 1980년경 이러한 죄수의 딜레마 상황을 반복적으로 수행하는 대회에 세계적인 게임이론가들이 다양한 방법론을 고안하여 참여하였는데, 이 대회의 우승 전략은 매우 간단하다: 1) 배반하기 전까지 항상 협력한다. 2) 만약 배반당했다면, 다음에는 배반한다. 3) 상대가 다시 협력해온다면 빠르게 관용을 베풀어 다시 협력한다.

처음 대하는 상대를 편견을 갖고 배척하지 않고 협력한다는 것 (원칙 1), 계속되는 배반 속에서 이용당하지 않는 것 (원칙 2), 그리고 과거에 원한관계에 연연해서 미래를 망치지 않는 대인배가 되기 (원칙3)등은 한번쯤 생각해 볼 전략이고, 실제로 국제외교에 활용되는 원칙이기도 하다.

게으름은 종종 효율적인 전략이다. 피보나치 힙은, 새로운 값이 추가될 때 힙에 추가해 정리하지 않고, 별도의 힙으로 생성해서 놓아 둔다거나, 루트를 추출한 후에도 힙을 정리하지 않고 내버려두는 게으른 구조이다. 추후에 필요하지 않을지도 정리를 게으름을 통해 회피함으로써 다른 부지런한 힙보다 (amortized cost면에서) 좋은 성능을 보인다. 비록 정리정돈을 하지 않을 수 밖에 없는 나의 학자적 양심을 비전공자인 가족들에게 이해시키는 데는 실패했지만, 분주함 속에서 위안을 찾는 마음을 경계하고 가끔은 현명한 게으름을 실천할 수 있어야겠다는 생각이 든다.

욕심만으로는 큰 뜻을 이룰 수 없다. 시간 관리를 비롯한 인생의 많은 문제는 NP-complete문제로 변환될 수 있는 어려운 문제이다. 그래서인지 탐욕스러운 근사 방법론이 대세이지만, 이 방법의 잘 알려진 최후는 차선책에 안주하게 되는 것이다. 때로 당장의 이로움을 버리고 골짜기 밑바닥을 향하는 것처럼 보이는 두려운 내리막길을 걸을 수 있어야 더 높은 곳에 이를 수 있다는 사실을 기억하고 싶다.

과학자의 고뇌:
세상은 온통 어림셈이고 마음먹은대로 흐르지 않는다

글쓴이 문병로
서울대학교 컴퓨터공학부 교수

과학자는 대부분의 시간을 '안타까운' 상태에서 보낸다. 해결해야 할 문제를 안고 잠시 기뻤다가는 좌절하기를 반복한다. 문제가 최종적으로 해결되는 순간에 느끼는 환희는 크지만 그리 길지는 않다. 그 순간부터는 그 해결 과정을 남들에게 전하기 위한 정리 작업에 들어간다. 정리 작업은 아무래도 문제 해결 과정보다 흥분도는 떨어지고 어떤 때는 몹시 스트레스를 준다. 과학이 혼자서 할 수 있는 순수한 작업 같지만 자신이 해놓은 것은 남에게 전달하는 능력은 연구 능력만큼이나 중요하다. 프리젠테이션과 논문 작성은 자신을 '마케팅'하는 행위다.

젊은 시절의 과학자는 자신이 해놓은 연구의 결과는 'hot' 하고 유용하게 쓰이게 될 것이라고 믿고 싶어한다. 그렇지 않다면 과학자라는 고상한 타이틀을 획득하기까지 바친 정열과 시간, 비용을 정당화할 수 없다고 믿는다. 슬프게도 과학자의 업적이 그대로 인류에 기여하는 경우는 아주 드문 케이스다. 대신 그 과정에서 훈련한 체계적 접근 방식, 사고법, 시행착오, 불가능한 스케줄을 감당해내는 능력, 이런 것들이 나중에 다른 현실적인 문제를 해결하도록 돕는다. 장년의 과학자는 젊은 시절의 자신과 쏙 빼닮은 젊은 과학자들을 보면서 그들이 가진 이상주의에 아름다움과 허무함을 동시에 느낀다. 다행스러운 것은 이상주의를 거친 과학자가 그렇지 않은 과학자보다 이론적으로 견고한 기반을 가질 확률이 높다는 것이다.

과학이 생각만큼 엄밀하지 않다는 것도 젊은 과학자의 자부심을 건드린다. 많은 과학 논문이 노이즈 속에서 가장 개연성이 높은 결론을 취한다. 이런 것을 못 견디면 노이즈가 상대적으로 적은 분야에서 활동하는 수학자가 된다. 과학과 공학의 실체에 가까이 가보면 놀라울 정도로 많은 부분이 노이즈와의 싸움이라는 것을 알게 된다. 자연과 사회가 노이즈로 엉킨 지저분한 장이라는 것을 받아들여야 한다. 이것은 자부심이나 자존심의 문제가 아니고 세상이 근본적으로 그렇게 만들어져 있다.

대중은 과학이 치열하게 엄밀한 것으로 생각하지만 대부분의 과학과 공학은 어림셈이다. 과학의 기본을 이루는 물리학조차 시작부터가 어림셈이다. 원자 자체가 전자들의 통계적 움직임으로 안정성을 유지하고 있다. 인간도 확률적으로 대사의 균형을 유지하면서 존재한다. 반도체 디자인 기술이 발달해서 현재 선폭이 20 나노미터를 조금 밑돈다. 머리카락 굵기의 2천분의 1 정도다. 컴퓨터 작동의 가장 말단은 이런 회선에 전자가 채워졌다 빠졌다를 반복하는 것이다. 회선이 두꺼울수록 많은 전자가 관여하고, 가늘수록 작은 수의 전자가 관여하게 되어 속도가 빨라지고 전력을 적게 소모한다. 대신 전자가 움직이는 확률적 견고함에 의존하는 반도체 회로에서 에러가 날 확률이 높아진다. 이 확률은 미미하지만 워낙 셀 수 없이 많은 횟수로 작동하므로 극미의 확률도 시간이 지나면서 발생하게 된다. 컴퓨터는 점점 '운좋게' 돌아가고 있는 것이다. 생물체를 구성하는 DNA는 1% 미만의 유용한 정보(엑손, exon)와 99% 이상의 노이즈(인트론, intron)로 구성되어 있다. 정상적인 인간의 몸에서도 쉴 새 없이 암세포가 만들어진다. 대부분 세력화 되지 못하고 죽는다. 유전적인 요인과 생활 습관, 생활 환경에 의해 확률적으로 어떤 상황이 만족되면 암으로 진화한다. 지나가다 이웃집 지붕에서 돌이 떨어지면 쉽게 죽을 수도 있다. 항공 사고, 자동차 사고, ... 이런 확률적으로 미약한 사건들을 피하면서 존재를 유지한다. 인간 이야말로 대표적으로 운좋게 살아있다. 일정 규모 이상의 컴퓨터 프로그램에는 인간 사고의 통상적인 결함이 대부분 포함된다. 최고 수준의 프로그램이라 할지라도 평균 3천 줄 당 한 줄 정도 에러(bug)가 포함된다. 현재 기업에서 사용하는 복잡한 프로그램은 수백만 줄짜리도 드물지 않다. 온갖 수준의 프로그래머들이 관여해서 거대한 프로그램을 만든다. 그것이 확률적으로 완전할 수 있겠는가? 거의 모든 컴퓨터 프로그램도 역시 확률적으로 운좋게 돌아가고 있는 것이다. 클로드 샤논은 아예 노이즈 속에 숨어있는 정보의 크기를 재는 것을 핵심으로 하는 학문을 만들어 내었는데 그것이 오늘날 통신, 열역학, 생물정보학, 데이터마이닝 등의 이론적 기초를 구성하고 있는 Information Theory, 즉, 정보이론이다.

필자가 관여하고 있는 주식투자공학도 확률 게임이다. 시장은 상식적으로 움직이지 않을 때가 많다. 장기적 현상이 단기적 현상보다 대체로 확률적 견고함이 높다. 앞에서 언급한 컴퓨터, 원자, 인간, 보험사보다 훨씬 취약한 확률적 견고함 위에서 이루어진다. 이것이 노이즈를 제공하고 투기를 부른다. 이런 확률적 불안함으로 인해 생기는 노이즈, 인간의 인지적 오류와 불합리성이 투자의 기회를 제공한다.

주식 투자에서도 돈을 버는 사람들은 '운좋게' 수익을 내고 있는 것이다. 그렇지만 여기서 '운좋게'라는 것은 '우연히'라는 의미는 아니다. '운높게'라고 표현하면 될까? 제대로만 하면 운의 크기를 자신이 미리 결정할 수 있다. 이 분야의 가장 큰 매력이 이 부분에 있다. 원자가 정상적으로 존재하거나 컴퓨터가 제대로 돌아갈 만큼의 확률적 크기로 운좋게 승리하는 투자를 한다면 돈을 못 벌 사람은 아무도 없다. 주식 시장에서 다루는 확률은 50% 근처의 확률이 대부분이다. 많은 개인투자자들이 시장에서 행동하는 행태는 근본적으로 승률이 40%도 안되는 방식이다. 이 시장에서 50%의 승률은 실패를 의미한다. 세금과 거래 수수료 만큼의 손실이 발생한다. "확률 게임으로 불안해서 어떻게 돈을 거나?" 라고 생각하는 사람은 이 시장에 들어오지 않는 것이 좋다. 확률적 게임을 피한다고 펀드에 맡기는 것은 누군가의 확률적 게임에 자신의 운을 맡기는 것일 뿐이다. 노벨상 수상자들을 포함해서 천재들의 집단으로 각광을 받던 롱텀캐피털매니지먼트(LTCM)는 5% 확률로 예외 상황을 가정하고 원금의 30배 이상 레버리지를 키웠다가 그 예외 상황이 발생하면서 파산했다.

투자 전략은 이론적으로 무한히 많고 이들은 전문용어로 '해 공간(각 전략이 하나의 해가 되고 이들이 분포하는 다차원 공간)'에 분산되어 숨어있다. 이런 것들을 찾는 작업이 최적화 전문가가 하는 일이다. 주식 공부를 많이 한 개인도 이런 전략 중 일부를 찾을 수 있다. 그렇지만 아무래도 컴퓨터로 방대한 공간을 찾는 최적화 알고리즘을 이길 수는 없다. 사람이 처리할 수 있는 데이터와 계산의 크기는 컴퓨터에 비하면 보잘 것 없다. 결국 컴퓨터의 관점에서 보면 '아주 단순한' 전략들만 찾을 수 있을 뿐이다.

주식 투자는 노이즈 투성이인 시장에서 가능하면 성공 확률이 높은 어림셈을 해내는 것이 핵심이다. 방대한 계산 능력과 최적화 수단이 없는 개인은 '아주 대충' 어림셈을 하게 되고, 컴퓨터와 최적화 기법을 구사하는 집단은 '좀 덜 대충' 어림셈을 하게 될 뿐이다. 노이즈를 다루는 이런 과학의 비중은 앞으로 점점 더 커질 수밖에 없다. 투자 분야에서도 이런 방향으로 거대한 패러다임의 전환이 일어나고 있다.

과학자로 학문적 여정을 시작해서 공학자로 전환했다. 하다보니 공학자에서 사업가를 겸하게 되었지만 아직도 과학자라는 생각/희망의 끈을 차마 놓지 못한다. 인생이 계획대로 흘러가지 않는다. 자신이 가진 다양한 빌딩블록들과 우연이라는 인자가 결합해서 전개되는 것 같다. 인생의 방향을 결정하는 데에 성공하겠다는 의지보다는 어떤 빌딩블록을 갖고 있느냐 하는 것이 더 중요한 듯하다. 이러다 보면 결국은 기초의 중요성을 한번 더 강조하게 된다. 어떤 집단의 연구도 방향을 지나치게 고정하는 것보다 구성원들이 자유로이 만들어내는 빌딩블록들이 자연스럽게 결합되는 과정이 더 소중해 보인다. 이런 면에서 무결점센터는 내가 본 가장 인상적인 조직이다. 우리 연구실은 무결점센터에서 가장 '결점많은' 연구주제를 만지고 있는 것 같다.

7

블로그 : ROSAEC minutes

Blog

New post | Edit | Delete

여름 워크샵 장소

2012 / 03 / 26

작성자: 이광근

교수님들께,

어제 개구리소리를 들었습니다. 봄은 여지없습니다.

이번 여름 센터워샵을 7-8월 중에 계획하고 있습니다.
워샵 프로그램은 계속되고 변화합니다. 성과발표/번개발표/소그룹난상토론/새 분야소개/기술전수/연구장터/파티 등.

장소가 고민입니다. 괜찮은 곳 추천부탁드립니다.

산, 섬, 리조트, 분교마을 어디라도 좋습니다. (해외까지)

이광근 드림

Scrap | Comments

« prev

next »

Blog

New post | Edit | Delete

2단계 참여교수님 변화

2012 / 03 / 25

작성자: 이광근

교수님들께,

학기 순항중이시라 믿습니다.

이번 3월부터 저희 센터의 2단계 3년이 시작되었습니다.

2단계에는 참여 교수님들의 변화가 있었습니다.

우선, 새로 참여하시는 교수님들은

- 박상혁 교수님(항공대): MIT에서 무인비행체를 연구하셨고, 해당 소프트웨어 검증관련 시너지의 한 축이 되실 것입니다.
- 염헌영 교수님(서울대): OS분야와 정적분석의 흥미로운 협업을 이끌어 주십니다.
- 차성덕 교수님(고려대): SW공학에서 접근하는 SW 신뢰도 검증을 맡아 주십니다.

이번 여름 웍샵에서 상견례하겠습니다.

그리고, 우치수/최진영/김유단 교수님들이 2단계에서는 물러나셨습니다.
그동안 감사드립니다.

이광근 드림

Scrap | Comments

« prev

next »

Blog

New post | Edit | Delete

1단계 중간평가 리포트

2011 / 11 / 27

작성자: 이광근

교수님들께,

오늘 저희센터 1단계 중간평가용 보고서를 온라인 제출했습니다.

행정팀과 최종 정리를 맡아준 이원찬 연구원과 이우석군이 고생많았습니다.

정리하다 보니, 교수님들과 연구진들의 성과들이 자랑스러웠습니다.
컴퓨터 분야 유일한 "선도연구센터"로서, 저희 성과가 그 이름 값을 했다고 생각됩니다.

12/14(수) 오전에 100분간 서울 연구재단에서 발표평가가 있습니다.
몇 분 같이 참여해 주십사 부탁드리겠습니다.

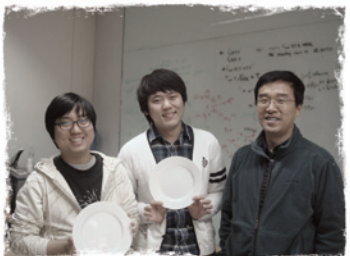
감사합니다,

이광근 드림

Scrap | Comments

« prev

next »



Blog New post | Edit | Delete

6th Workshop

2011 / 04 / 20 작성자: 이광근

교수님들께,

꽃과 바람이 많은 날들입니다.

이번 여름 센터워샵 관련입니다.

- 1단계 평가 전 마지막 웍샵입니다.
  모두 모여서 성과들은 헹가래해주시고 미흡한점은 기대어주시기를.
  교수님들끼리 모여서 평가준비 아이디어를 모으는 시간을 가지겠습니다.

- 프로그램위원장은 박성우 교수님이 맡아주시기로 했습니다. 감사합니다.
  (1단계 웍샵 전속이십니다)

- 프로그램으로는 성과발표/번개발표/포스터웅성웅성/뇌폭풍/소풍활동/회식 등이 계속됩니다.

- 새로운 프로그램중 하나로: 밤에 재주있는 학생과 교수님들의 마당음악판을 생각하고 있습니다.
  (피아노, 드럼, 기타, 베이스, 섹소폰, 해금, ...) 학생들중에서 재원을 찾아주시고 또 자원해 주십시요.

* 웍샵 프로그램으로 시도해 볼 좋은 아이디어 있으시면 박성우교수님이나 제게 말씀해 주십시요.

* 행정실에서 자세한 사항 그때 그때 공지드릴 것입니다.

여름날 뵙겠습니다,

이광근 드림

Scrap | Comments

« prev next »

Blog

New post | Edit | Delete

newsletter, visiting students

2011 / 02 / 18

작성자: 이광근

안녕하십니까.

몇가지가 있습니다.

* 오늘 센터 뉴스레터를 인쇄소에 맡겼습니다. 지난 5월에 이어 두번째입니다.
  센터홍보 차원에서 널리 발송하려고 합니다. 읽어보실 분은 20% 미만일겁니다.
  하지만 펼쳐보실 분들을 생각했습니다.

* 이번 봄/여름에 외국에서 박사과정 학생들이 연구인턴으로 센터를 방문합니다. 기대됩니다.

- Ludovic Patey(박사과정), 3/10-8/15,
  Ecole Normale Superieure, Paris
  (지도교수: Patrick Cousot)

- Cristian Gherghina(박사과정), 5/30-8/30,
  National Singapore U, Singapore
  (지도교수: Wei-Ng Chin)

- Jie Chen(박사과정), 3/9-6/9,
  Chinese National U of Defense Technology, Changsha
  (지도교수: Ji Wang)

- Saransh Srivastava(학사과정), 5/10-7/10, IIT Kanpur, India

* 그리고 1단계 평가에서 우리가 내세울 포인트에 대해서도 곧 말씀드리겠습니다.

이광근 드림

Scrap | Comments

« prev

next »

8 연구실, 유학생, 행정팀 소식

한양대학교 프로그래밍언어연구실

도경구 교수님 연구실 소개

한양대학교 프로그래밍언어연구실은 도경구 교수가 부임한 1995년 9월에 설립하여 총 35명의 석사를 배출하였으며, 현재 연구교수 1명, 박사후연구원 1명, 박사과정 4명, 석사과정 5명, 학사과정 4명이 열심히 연구에 매진하고 있다. 프로그래밍언어연구실은 별명인 PLaSSE(Programming Language applied to Software Security and Engineering)를 보면 연상할 수 있듯이, 프로그래밍언어 기술을 개발하고 이를 소프트웨어 보안 및 소프트웨어 공학에 적용하여 양질의 건강한 소프트웨어를 제작할 수 있는 토대를 만드는 연구를 주로 하고 있다.

문자열분석

소프트웨어가 실행 중에 문자열로 생성하는 프로그램 또는 반구조문서의 구문과 의미를 소프트웨어를 실행해보지 않고 어림잡아 그 특질을 파악하는 문자열 분석기술을 연구하고 있다. 실행 중에 생성되는 문자열의 구문을 정적으로 분석할 수 있는 요약파싱(abstract parsing) 기술을 개발하고 구현하였으며, 속성문법을 사용하여 의미분석까지 할 수 있게 요약파싱을 확장하였다. 이를 토대로 JSP가 실행 중에 항상 유효한 HTML 문서를 생성하는지 정적으로 분석하는 실험을 성공적으로 수행하여, 실제로 현장에서 사용하고 있는 JSP에서 다수의 오류를 발견하였다. 요약파싱 기술을 계속 발전시키고 관련 이론을 정립하고 있으며, 현재 명령삽입공격 취약점의 정적분석의 정밀도를 획기적으로 높이는데 적용할 수 있는 방안을 연구하고 있다.

웹애플리케이션 보안취약성 자동 진단

웹애플리케이션의 소스코드에 숨어있는 보안 취약성을 찾아내는 분석 방법을 연구하고 있다. 보안 취약성을 유발하는 구문 및 실행 흐름 패턴을 규칙으로 명세하고, 이를 프로그램 분석을 통하여 찾아내는 분석기를 산학협동으로 개발하여 출시한 제품에 핵심엔진으로 장착하였다. 현재 자동 진단 엔진의 오탐 및 미탐을 줄이기 위하여 분석엔진의 정밀도를 높이기 위한 연구와 동적분석과 결합하는 연구를 계속 진행하고 있다.

KAIST PLRG 연구실

류석영 교수님 연구실 소개

KAIST PLRG (PL알지!) 연구실은 2009년 12월 류석영 교수님께서 부임하시면서 시작되어, 현재 석사후연구원 1명, 박사과정 1명, 석사과정 2명, 학부과정 5명으로 구성되었습니다. 다양한 프로그래밍 언어와 프로그램 분석에 관련된 연구를 진행하고 있으며, 학부생들이 개별 연구 및 학부생 연구 프로그램으로 함께 연구를 진행하고 있습니다. 저희 연구실에서 진행하는 연구 분야를 소개하면 다음과 같습니다.

안전한 JavaScript 웹 어플리케이션 개발/사용을 위한 시스템 개발

JavaScript 프로그래밍 언어는 원래 간단한 스크립트를 작성하는 용도로 처음 개발되었지만, 사용자의 입력에 따라 달라지는 동적인 웹페이지를 손쉽게 나타낼 수 있어서, 거의 모든 웹 서비스에서 JavaScript를 사용하고 있습니다. 가장 많이 사용되는 언어 10위에 오를 정도로 널리 사용되고 있지만, 외부 코드에 의해 내부의 보안 정보를 유출시키는 등 보안에 매우 취약하다는 단점이 있습니다. 본 연구에서는 JavaScript 언어에 모듈 시스템을 추가하여 프로그램을 모듈별로 안전하게 개발할 수 있도록 하고, 언어 기반 샌드박싱 기법을 제공하여 중요한 정보가 유출되지 않는 시스템을 개발하며, 실제 광범위하게 사용되고 있는 JavaScript 어플리케이션의 동적 성질을 조사, 연구하여 개발자들에게 직접적으로 도움이 되는 JavaScript 개발 환경을 연구하고 있습니다.

Fortress 프로그래밍 언어 설계 및 개발

Fortress 프로그래밍 언어는 다양한 기능을 제공하고 있는데, 여러 가지 기능을 안전하게 지원하기 위해서 타입 검사가 복잡해집니다. Fortress 언어는 처음 설계될 때부터 제대로 검증하면서 만드는 것을 목표로 하였는데, 본 연구에서는 Coq이라는 증명보조도구를 이용해서 Fortress의 복잡한 타입 시스템이 제대로 만들어져서 타입 안전성을 보장한다는 것을 증명합니다. 먼저, Fortress 핵심 언어에 같은 이름을 갖는 여러 함수를 동적 타입으로 선택하는 것과 다중 상속을 모두 지원하도록 Coq으로 정의하여 그 타입 안전성을 증명하였습니다. 현재는 타입 안전성이 증명된 핵심 언어에 모듈과 다양한 함수 모양을 추가하여 그 의미구조를 엄밀하게 정의하고, Coq으로 증명을 진행하고 있습니다.

의료 영상 기기의 소프트웨어 개발/분석

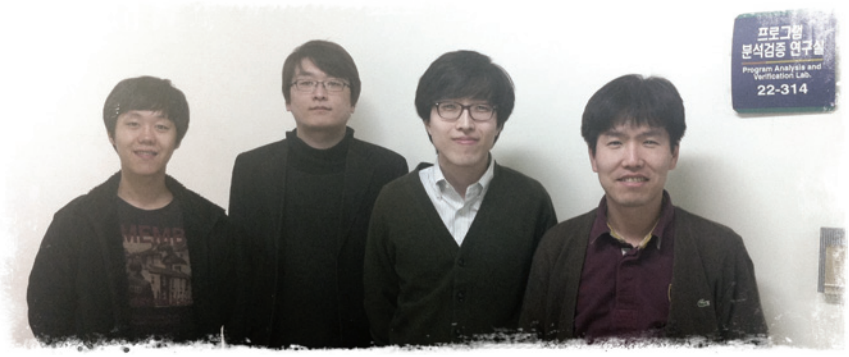
OFDI라는 영상 기기를 사용하여 혈관 내부를 이미징하는 시스템 소프트웨어를 개발 및 분석하고 있습니다. C++로 작성된 코드를 CUDA로 재작성하여 병렬화를 높이고 실제 임상에서 사용할 수 있도록 성능 및 기능을 향상시키고 있습니다. GPU 특유의 메모리 구조와 프로그래밍 스타일을 분석하며, 다양한 정보를 효율적으로 보여주는 영상 가공 소프트웨어를 작성하고 있습니다. 이를 이용하여, 의료 영상 소프트웨어의 안전성을 분석하는 데에 관심을 가지고 있습니다.

중학생을 위한 프로그래밍 언어, 바심 개발

학부생들이 주축이 되어 시작하고 주도하고 있는 연구로, 중학교 1학년생을 대상으로 하는 프로그래밍 언어인 바심을 설계하고 개발하고 있습니다. 중학교 교과과정을 고려한 교육 과정도 설계하여 교재를 개발 중이며, 어린 학생들이 따로 개발 환경을 설치할 필요가 없도록 웹기반 개발 환경을 만들었습니다.

한양대학교 프로그램 분석검증 연구실

이욱세 교수님 연구실 소개



한양대학교 프로그램 분석검증 연구실(PAV Lab; Program Analysis and Verification Lab)은 견고한 프로그래밍 지원을 위한 프로그래밍 언어 분야 핵심 기술을 연구하고 있습니다. 본 연구실은 2004년에 이욱세 교수가 창설하였으며, 현재 박사과정 1명, 석박사통합 1명, 석사과정 2명으로 구성된 작지만 젊은 연구실입니다. 2004년부터 지금까지 중국 학생과 파키스탄 학생을 포함하여 7명의 석사 졸업생을 배출하였습니다. 연구실의 주된 연구분야는 프로그램 분석 기술과 응용이며, 같은 과 도경구 교수님의 프로그래밍 언어 연구실 및 영국 옥스퍼드 대학 양홍석 교수님과 긴밀한 협력 연구를 진행하고 있습니다.

무결점 소프트웨어를 위한 오류 검출

주된 연구 주제는 무결점 소프트웨어를 위한 오류 검출입니다. 특히 그 복잡성으로 인해 단순한 프로그램 분석으로는 오류를 검출하기 어려운 시스템 코드가 그 대상입니다. 지금까지는 복잡한 자료구조를 분석하는데 초점을 두어 왔고, 찾아낸 방법론은 분리논리(separation logic)를 기반으로 하는 모양 분석기입니다. 2005년 최초의 분리논리 기반 모양 분석기를 개발한 것으로 시작해서, 영국 Queen Mary University of London에서 개발한 SpaceInvader 분석기 구현에 직접적으로 참여하여, 10,000줄 정도의 복잡한 시스템 코드의 무결성을 검증할 수 있도록 하였습니다. 현재는 실제 산업현장에 적용할 수 있는 모양 분석기 개발 및 동시성 시스템 코드에서 오류를 검출하는 기술에 초점을 두어 연구하고 있습니다.

코드보안취약점검출

코드의 오류가능성 뿐만 아니라 보안취약점을 자동으로 검출해 내는 것도 연구실의 연구 주제입니다. 같은 과 도경구 교수님의 프로그래밍 언어 연구실과 협력하여 연구를 진행하고 있습니다. 첫번째 연구결과로는 요약해석 기술을 사용하여 프로그램이 생성해내는 문자열을 계산해 내는 분석기입니다. 많은 보안 취약점이 실행 중에 잘못 만드는 문자열에 기인하는데, 생성 가능한 문자열을 분석해 냄으로써 보안 취약점을 미리 검출해 낼 수 있었습니다. 두번째 연구결과로는 여러 보안 취약점을 한번에 검증해 낼 수 있는 실용적인 통합 분석기 개발입니다. 대부분의 보안취약점은 단순히 프로그램 흐름만 분석하면 검출할 수 있습니다. 그러한 쉽게 검출 가능한 보안 취약점을 그 스펙만 적어주면 분석해 낼 수 있는 분석기를 개발하였습니다. 향후에도 프로그램 분석 기술을 이용하면 많은 보안 취약점을 미리 검출해 낼 수 있으리라 생각 됩니다.

소스코드 복제 검출

조금 다른 주제이긴 하지만 소스코드 복제 검출 또한 연구실의 주된 관심사입니다. 산업현장에서 소스코드를 불법적으로 도용하는 사례가 종종 발생 하는데 이는 소프트웨어 개발자의 노하우과 노력을 훔쳐가는 중대한 범죄입니다. 법정에서 소스코드 도용을 증명하는데 필요한 기술을 개발하는 것이 목표입니다. 그 동안의 연구결과를 활용하여 한국저작권위원회가 두 소스코드 간에 유사성을 검출하는 도구를 개발하는데 직간접적으로 도움을 주고 있습니다.

KAIST 응용 알고리즘 연구실

정교민 교수님 연구실 소개

카이스트 응용 알고리즘 연구실 (KAIST Applied Algorithm Lab.)은 2009년 정교민 교수님의 부임과 함께 설립되어, 현재까지 석사학위 4명을 배출하였습니다. 현재 박사과정 학생 3명, 석사과정 학생 2명이 정교민 교수님의 지도하에 알고리즘을 분석하고 다양한 분야로 적용하는 연구를 진행하고 있습니다.

'응용'이라는 단어에서 알 수 있듯이, 알고리즘적 접근 방법을 요하는 폭넓은 문제들을 다루고 있습니다. 특히, 우리 연구실은 실용적인 알고리즘 개발뿐만 아니라, 수학적 모델링을 통한 분석 및 증명도 함께 진행할 수 있는 역량을 갖추고 있습니다. 현재 다루고 있는 연구 문제는 크게 1) 네트워크 내 정보흐름 분석 및 진화 예측 기법과 2) 대용량 데이터에 적용하기 위한 기계학습 확장화의 두 갈래로 나뉩니다.

네트워크 분석 연구

네트워크 분석 연구에서는 정보 흐름 분석과 진화 과정 예측 기법을 개발하고 있습니다. 네트워크에서 정보는 서로 연결된 구성원들 간의 접촉으로 전달되며, 우리는 임의의 네트워크 위에서 대규모 정보 확산이 어떠한 조건(tipping point)이 만족될 때 발생할 것인지를 수학적인 분석 기법을 통해 밝혀내고 있습니다. 대규모 확산의 발생 확률 및 규모에 대해서도 예측 및 분석을 하고 있으며, 보다 일반적인 긍정적/부정적 정보가 공존하는 경우에 대한 효과적인 확산 촉진/억제 기법의 개발도 하고 있습니다. 그리고 위와 같은 모델들에 대해 확장성이 있는(scalable)알고리즘을 개발하는 것에도 초점을 맞추고 있습니다. 이외에, 어떠한 구성원들로부터 정보 확산이 시작될 때 그 영향력이 가장 극대화되는지 알아내는 영향력 최대화(influence maximization) 문제, 시간에 따른 네트워크 내 연결 관계가 어떻게 변해나갈지 예측하는 네트워크 진화(network evolution) 예측 문제 등 네트워크 과학의 중점적인 문제들 역시 우리 연구실에서 해결 중인 중요한 문제들입니다. 더 나아가, 네트워크의 구조적 연구 및 동역학적 연구를 융합하여 네트워크로 표현 가능한 다양한 체계로부터 발생하는 현상들을 설명하는 것에도 관심이 있습니다.

기계학습 확장화

기계학습(machine learning)은 경험에 따라 성능이 향상되는 체계를 연구/개발하는 인공지능의 한 분야입니다. 우리 연구실에서는 여러 기계학습 기법을 대용량 데이터에 적용하는 알고리즘의 확장화에 초점을 맞춰 연구하고 있습니다. 현재 다루는 문제는 네트워크의 집단 구조 분석과 밀접한 관련이 있는 군집화(clustering) 알고리즘입니다. 네트워크의 노드들을 군집화하려면 각 노드들을 유클리드 공간으로의 임베딩(embedding)한 정보를 주는 커널(kernel) 함수를 사용하여야 하는데, 커널 함수의 특성상 데이터 개수의 제곱에 비례하는 계산 복잡도를 갖게 됩니다. 최신 복잡계 망의 크기가 100만 단위 이상이라는 것을 고려하면 군집화 알고리즘의 수행이 사실상 불가능하기 때문에 이 복잡도를 낮추는 연구를 진행하고 있습니다. 아울러, 높은 차원의 대용량 데이터 검색을 효율적으로 할 수 있도록 지역적으로 민감한 해싱(LSH: Locality-Sensitive Hashing) 기법도 연구하고 있습니다. 대용량 네트워크 데이터에서 마이닝(data mining) 작업을 하기 위해서는 빠른 속도로 데이터를 검색하는 것이 필수적이기 때문에 LSH의 확장화를 통해 데이터 마이닝의 효율성과 정확성을 높이는 것을 목표로 하고 있습니다.

유학생 소식

MIT, Laboratory for Computer Science and Artificial Intelligence 박사과정, 김덕환

소개

안녕하세요? 저는 2005년부터 2007년까지 서울대학교 프로그래밍 연구실에서 이광근 교수님을 지도 교수님으로 모시고 석사 과정을 마친 김덕환이라고 합니다. 지금은 2009년 가을에 매사추세츠 공과대학교 전기컴퓨터공학부로 유학을 와 마틴 라이나드(Martin Rinard) 교수님께 지도를 받으며 박사 과정을 밟고 있습니다.

지난 경험을 바탕으로 제 주위 외국 학생들에게서 부럽거나 배우고 싶은 세 가지를 정리해보고자 합니다.

다양한 연구 기회

여기 학생은 학부 과정 중에 정말 진지하게 연구를 경험할 기회가 참 많습니다. 세계적인 연구소에서 인턴사원으로 근무하거나 학교 연구실에서 지원을 받으며 연구에 적극 참여할 기회가 많이 열려 있습니다. 그래서, 이미 해당 분야에 상당한 지식을 갖추고 대학원에 입학하는 학생이 많습니다. 외국 학생 중 비교적 이른 시기부터 좋은 연구 결과를 내는 학생은 그런 바탕이 있었기에 가능하다고 생각합니다. 사실 이는 개인적인 노력뿐만 아니라 제도적인 뒷받침도 중요한 부분이라 부러운 점입니다.

적극적으로 피드백 받기

연구 초창기부터 자기 아이디어를 주위 사람들에게 소개하고 피드백을 받는 일에 참 열심입니다. 연구 초창기의 아이디어를 소개하다 보면 칭찬을 받기보다는 제안자가 미처 깨닫지 못한 약점을 지적받기 일쑤입니다. 그럴 때 저라면 창피해하거나 좌절할 만도 한데, 이 친구들은 참 번번스러울 정도로 그걸 대수롭지 않게 여깁니다. 그리고는 며칠 후에 지적된 사항을 개선한 아이디어를 가지고 다시 얘기를 걸어옵니다. 이런 과정을 여러 번 겪으면서 아이디어 제안자는 아이디어 자체를 튼튼하게 만들 뿐만 아니라, 사람들이 자기 아이디어 중 어떤 부분을 이해하기 어려워하거나 오해하는지를 자연스럽게 파악합니다.

문제 재단을 노련하게

문제 자체를 재단하는 데 참 노련합니다. 문제를 풀다 보면 예상치 못했던 커다란 장애물을 발견하여 문제가 처음 생각보다 훨씬 어렵다는 것을 깨달을 때가 종종 있습니다. 그 문제를 온전하게 풀려면 정면 승부를 하는 것이 옳습니다. 하지만 합리적인 범위 내에서 새로운 제약 조건을 도입하는 등 풀고자 하는 문제 자체를 수정하는 것이 훨씬 효율적일 때도 있습니다. 바꿔 말하면, 커다란 문제 안에서 합리적인 시간 안에 내가 잘 풀 수 있는 작은 문제를 찾아낸다고 할 수 있습니다. 문제라는 것이 딱 정해져서 주어지는 것이 아닌 내가 잘 풀 수 있는 문제로 변환하는 태도와 능력은 꼭 배워야겠습니다.

맺음말

문화적인 배경이나 개인적인 성향 때문에 저나 제 주위 분들에게 상대적으로 부족하지 않나 싶은 점들을 짧게 정리해보았습니다. 다음에는 학회에서 저자들로 만나 인사나눌 수 있었으면 좋겠습니다.

행정팀 소개

1. 센터에서 하는 일　2. 기억에 남는 일　3. 포부

장승영

1. 행정 전반을 담당하고 있는 장승영입니다. 지난 1년간 연구비 관리를 하며 이제 조금 익숙해졌다고 생각했는데 2단계에 들어가며 새로이 업무분장을 하게 되어 다시 입사한 기분입니다.

2. 지지향과 홍콩에서 열린 워크샵이 가장 기억에 남습니다. 원거리에 있어서 뵙기 어려운 연구원분들까지 한자리에 모이는 자리인 만큼 소프트웨어무결점연구센터의 실체를 확인할 수 있었고, 모든 프로그램에 적극적으로 참여하시는 모습이 무척 인상적이었습니다.

3. 이번 주 '바보야 성공이 아니라 섬김이야'라는 엘리제쉐핑 평전 출간 소식을 들었습니다. 행정팀으로 일하는 것이 그동안 제가 알던 성공과는 꽤 다르지만 잘 가고 있다는 생각을 합니다. 잘 섬길 수 있는 우리가 되기 위해 즐겁게 훈련받겠습니다.

유하나

1. 센터장님 비서 업무 및 행정 업무를 담당하고 있는 유하나입니다. 제 주된 업무는 센터장님과 센터 연구원들께서 연구에만 집중하실 수 있게 행정 업무를 도와드리는 일입니다.

2. 센터에서 1년간 근무하며 가장 기억에 남는 것은 홍콩에서 개최했던 "ROSAEC-HKUST CSE Joint Workshop" 이었습니다. 준비과정은 힘들었지만, 워크샵이 끝난 후 특별한 곳에서 새로운 경험을 했다는 것이 기쁘고 뿌듯했습니다.

3. 많은 분들이 이해해주시고 도와주셔서 지금까지 올 수 있었다고 생각합니다. 앞으로는 그동안 했던 실수를 반복하지 않도록 업무에 더욱 집중하고, 센터에 꼭 필요한 사람이 되겠습니다.

전다혜

1. 연구비 관리를 담당하고 있는 전다혜입니다. 연구에 참여하시는 교수님들과 이하 연구원들이 행정적으로 어려움이 없도록 돕는 것이 제 가장 중요한 업무입니다.

2. 지난 시간 동안 센터를 위해 수고해주셨던 성윤경 사무장님께서 사직하실 때 함께 환송했던 일이 가장 기억에 남습니다. 교수님과 연구원들이 다 함께 모여 카드에 감사의 마음을 담아 직접 전해주시는 것을 보며, 사람을 귀하게 생각하고 마음을 나눈다는 것이 제게도 전해졌습니다.

3. 제1순위는 연구비 관리를 통해 최고의 행정 서비스를 제공하는 것입니다. 그리고 또 다른 한 가지 포부는 일적인 부분뿐만 아니라 삶도 함께 나눌 수 있는 좋은 벗이 되고 싶습니다.

9 참여교수 연락처

제1총괄과제

1-1세부과제	교수 **백윤흥**	02-880-1748	[redacted]	ypaek@ee.snu.ac.kr	서울대학교 전기공학부
1-2세부과제	교수 **이광근**	02-880-1857		kwang@ropas.snu.ac.kr	서울대학교 컴퓨터공학부
1-3세부과제	교수 **도경구**	031-400-5667		doh@hanyang.ac.kr	한양대학교 컴퓨터공학과
	조교수 **류석영**	042-350-3538		sryu@cs.kaist.ac.kr	KAIST 전산학과

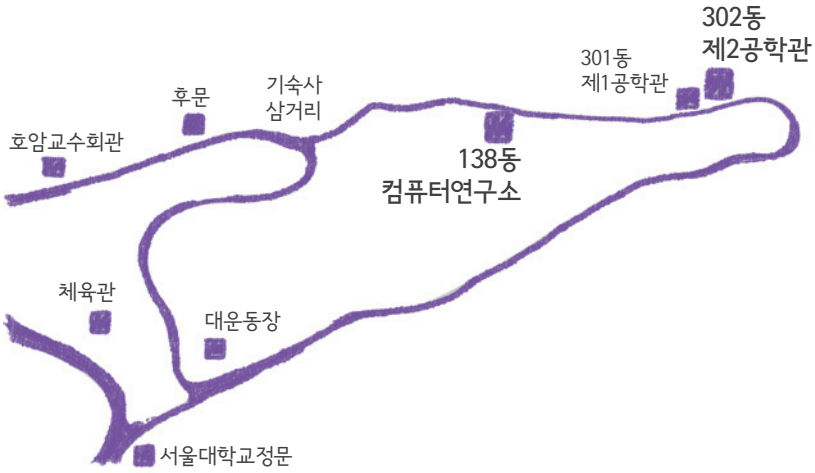
제2총괄과제

2-1세부과제	교수 **염헌영**	02-880-5583	[redacted]	yeom@snu.ac.kr	서울대학교 컴퓨터공학부
	조교수 **이욱세**	031-400-5234		oukseh@hanyang.ac.kr	한양대학교 컴퓨터공학과
2-2세부과제	교수 **문병로**	02-880-8793		moon@snu.ac.kr	서울대학교 컴퓨터공학부
	조교수 **정교민**	042-350-3544		kyomin@kaist.edu	KAIST 전산학과
2-3세부과제	교수 **박종우**	02-880-7133		fcp@snu.ac.kr	서울대학교 기계항공공학부
	조교수 **박상혁**	02-300-0171		park@kau.ac.kr	항공대학교 기계항공공학부
2-4세부과제	교수 **최광무**	042-350-3520		choe@kaist.ac.kr	KAIST 전산학과
	조교수 **김장우**	054-279-2390		jangwoo@postech.ac.kr	POSTECH 컴퓨터공학과

제3총괄과제

3-1세부과제	부교수 **박성우**	054-279-2386	[redacted]	gla@postech.ac.kr	POSTECH 컴퓨터공학과
3-2세부과제	조교수 **김문주**	042-350-3543		moonzoo@cs.kaist.ac.kr	KAIST 전산학과
	조교수 **최윤자**	053-950-7549		yuchoi76@knu.ac.kr	경북대학교 전자전기컴퓨터학부
	교수 **차성덕**	02-3290-3585		scha@korea.ac.kr	고려대학교 정보통신대학
3-3세부과제	부교수 **황승원**	054-279-2385		swhwang@postech.ac.kr	POSTECH 컴퓨터공학과
	부교수 **한욱신**	053-950-7572		wshan@knu.ac.kr	경북대학교 컴퓨터공학과

오시는길



대중교통 이용
낙성대역 (4번 출구) → 마을버스(2번) → 후문 통과 → 302동 정류장 하차
서울대입구역 (3번 출구) → 버스(5511, 5513) → 정문 통과 → 302동 정류장 하차

발행일 2012년 3월
발행인 이광근
편집인 오학주
발행처 ROSAEC center

서울시 관악구 관악로 1 서울대학교 138동 202호
T 02-880-7290
F 02-882-7234
E rosaec@snu.ac.kr
W rosaec.snu.ac.kr

디자인 www.koncept.co.kr

Research On Software Analysis for Error-free Computing

소프트웨어 무결점 연구센터 NRF ERC

151-742 서울시 관악구 관악로 1 서울대학교 138동 202호
Rm 202 Bldg 138 Seoul National University 1 Gwanak-ro Gwanak-gu Seoul 151-742 KOREA
Tel +82 2 880 7290 Email rosaec@snu.ac.kr
Fax +82 2 882 7234 Web rosaec.snu.ac.kr