2011 ROSAEC CENTER NEWSLETTER

소프트웨어무결점 연구센터 뉴스레터

No. 2 Jan, 2011

rosaec.snu.ac.kr

교육과학기술부 / 한국 연구재단 지정 선도연구센터

ROSAEC center

Research On Software Analysis for Error-free Computing

소프트웨어 무결점 연구센터 NRF ERC

목차

1 센터장 보고 01
2 센터 연구소개 02
3 대표논문과 이야기 05
4 협력연구 이야기 15
5 세미나와 워크샵 26
6 소프트웨어 클리닉 서비스 32
7 방문기 34
8 블로그 : ROSAEC minutes 49
9 연구실 소개 & 유학생 소식 55
10 참여교수 연락처 64

1 센터장 보고

안녕하십니까.

저희 소프트웨어무결점 연구센터가 선도연구센터(ERC)로 시작된 지 2년 반이 지났습니다.

그동안 저희 센터의 성과들이 자라왔습니다.

우선은 논문 성과입니다. 소프트웨어 분야(프로그래밍언어, 소프트웨어공학, 데이터베이스, 인공지능등)에서 저희가 명실상부 "세계최고"의 학회에 논문을 내고 있습니다. 모두 8000미터급 정상들입니다. 컴퓨터 분야에서 이런 학회에 논문을 내는 것은 SCI저널 논문보다 어렵고 많은 변수가 있습니다. "POPL", "ICSE", "SIGMOD", "AAAI", "ICFP" 등에 오르며 저희 센터가 한국의 소프트웨어 연구력을 세계에 과시하고 있습니다.

두번째는 산학협력 성과입니다. Sparrow라는 제품으로 상용화 된 저희 센터의 C 프로그램 오류 자동 검증기가 있습니다. 수천만원에서 일억원 이상되는 소프트웨어가 시장에서 팔리게 하는데는 엄격한 요구사항이 많습니다. 이런 현장의 문제들이 우리 고유의 연구문제가 되어 센터에 전달됩니다. "우리 주변의 문제가 세계적인 연구를 드라이브"한다는 소신을 확인하고 있습니다. 좋은 산학협력의 선순환 싸이클입니다.

관학협력도 있습니다. 행정안전부에서는 정부에서 사용할 소프트웨어들의 보안허점을 미리 검증하는 시스템을 구축하고 있습니다. 저희 센터의 기술을 기반으로 하는 검수 도구(소프트웨어 소스 분석도구)가 개발되고 있습니다. 정부에서 외산제품 대신에 저희 센터의 연구진들이 주도해서 개발한 시스템을 사용하는 것입니다.

그리고 센터기술의 보급을 위해서 "소프트웨어 클리닉" 서비스를 시작합니다. 소프트웨어의 소스를 저희 시스템으로 분석해서 오류 진단 결과를 보내드립니다. 웹 기반으로 누구에게나 무상 서비스됩니다. 걸림돌과 한계가 있지만 나섰습니다.

아직 더딘 것도 있습니다. 국내에서 로봇과 항공우주 분야에 장착되는 소프트웨어들에 특화된 검증기를 만들었으면 하는데 아직 확실한 계기를 만나지 못하고 있습니다. 준비에 바빴습니다. 기회를 찾겠습니다.

이외 여러가지가 움트고 있습니다.

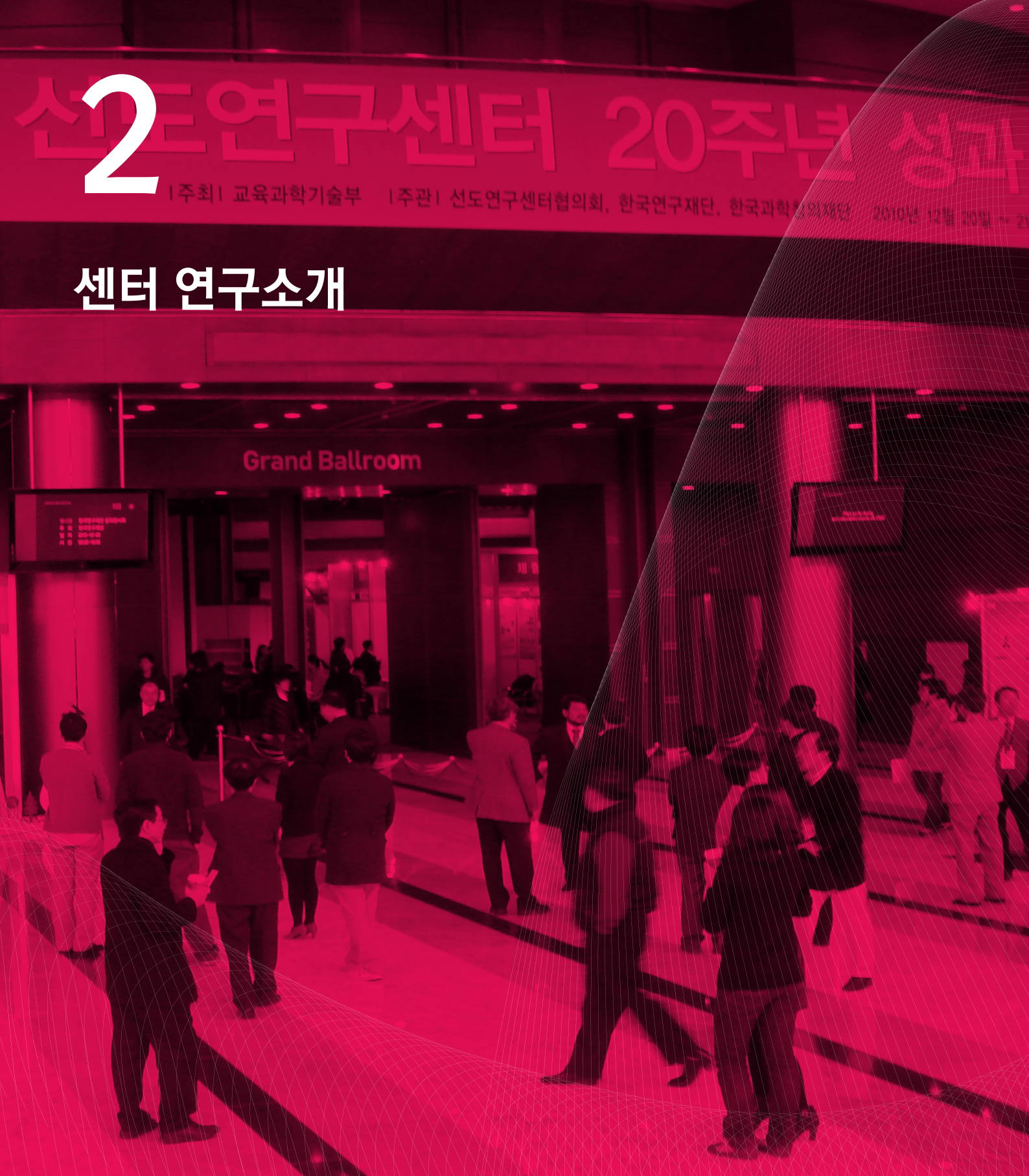
많은 관심과 격려를 부탁드립니다.

감사합니다.

센터장 이광근
서울대학교 컴퓨터공학부 교수

2

센터 연구소개



센터소개

제대로 작동할 지를 미리 검증할 수 없는 기계설계는 없다.
제대로 서있을 지를 미리 검증할 수 없는 건축설계는 없다.
인공물이 자연세계에서 문제없이 작동할 지를 미리 엄밀하게 분석하고 검증하는 기술은 잘 발달해왔다.
이러한 분석 검증 기술은 다른 엔지니어링 분야에서는 당연히 확립된 기술이다.
왜냐하면 모든 엔지니어링의 제일 근본적인 질문이 바로,
우리가 만든 것이 우리가 의도한대로 움직인다는 것을 어떻게 확인할 수 있는가? 이기 때문이다.

컴퓨터 소프트웨어에 대해서는 어떤가?
작성한 소프트웨어가 제대로 실행될지를 미리 엄밀하게 확인해 주는 기술은 있는가?
그래서 작성한 소프트웨어가 가질 수 있는 오류를 자동으로 미리 모두 찾아주거나, 없으면 없다고 확인해 주는 기술은 있는가?
그래서 소프트웨어의 오류 때문에 발생하는 개인/기업/국가/사회적 비용을 절감시켜주는 기술은 있는가?

오류 검출 및 검증 기술의 첨단에서 실용가능한 기술을 규명해내고 남들보다 먼저 산업화해내는 그룹은
확산 직전에 있는 거대한 무결점 소프트웨어 개발 도구 시장의 주도권을 잡게 될 것이다.

01 SW 분야의 근본문제

작성한 SW의 오류를 자동으로 미리 모두 찾아주거나, 없으면 없다고 확인해주는 기술들은 있는가?

02 무결점 SW의 시장가치

점점 막대해지는 SW 오류의 비용 (개인/기업/사회/국가)
미국: 60조원/년
모든 제품 경쟁력 = SW품질
가전,기계,통신,교통,국방,의료,에너지
제품 리콜, 사회 혼란, 국가 손실
SW 제작중 오류수정에 쓰는 비용 : 20조원/년
2010 SW 시장 : 310조원

03 센터의 목표

원천기술 선도 정적분석 기술 심화 + 혁신 신기술 연구

실용적 도구 개발 SW 오류 자동 검출/검증기 개발

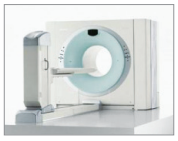
특화된 산업화 순수SW, 지능형 로봇, 무인 비행체,
금융 공학에 특화시켜 산업화

04 센터의 목표 도구

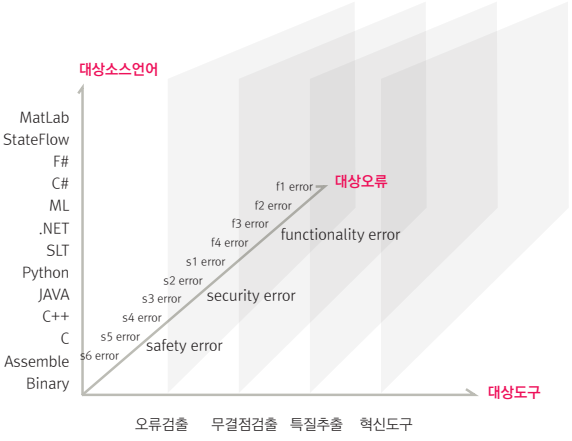
소프트웨어 MRI

소프트웨어 fMRI

소프트웨어 PET



05 목표 도구의 공간



06 세부 초점사항 : 원천기술

정적분석(static analysis)

이 기술은 엄밀히 예측한다.

이 기술은 테스팅의 단점을 보완한다.

이 기술의 실용성은 확인되었다.

산업화 완료한 SPARROW를 통해

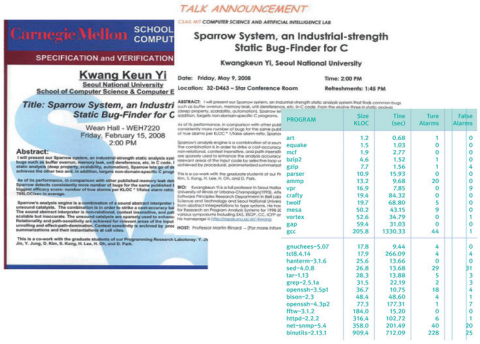
static analysis, abstract interpretation, type system, program logic, theorem proving, model checking

www.spa-arrow.com

Sparrow The Early Bird

07 원천기술 실용성 예시 : SPARROW

대상 : C, memory leak/buffer overrun,
오류 검출율 6/KLoc, 속도 10Loc/sec


TALK ANNOUNCEMENT

Carnegie Mellon SCHOOL COMPUT

SPECIFICATION and VERIFICATION

Kwang Keun Yi
Seoul National University
School of Computer Science & Computer E

Title: Sparrow System, an Industr
Static Bug-Finder for C

Wean Hall - WEH7220
Friday, February 15, 2008
2:00 PM

Abstract

Sparrow System, an Industrial-strength Static Bug-Finder for C

Kwangkeun Yi, Seoul National University

Date: Friday, May 9, 2008 Time: 2:00 PM
Location: 32-D463 – Star Conference Room Refreshments: 1:45 PM

PROGRAM	Size KLOC	Time (sec)	Ture Alarms	False Alarms
art	1.2	0.68	1	0
equake	1.5	1.03	0	0
mcf	1.9	2.77	0	0
bzip2	4.6	1.52	1	0
gzip	7.7	1.56	1	4
parser	10.9	15.93	0	0
ammp	13.2	9.68	20	0
vpr	16.9	7.85	0	9
crafty	19.4	84.32	0	0
twolf	19.7	68.80	5	0
mesa	50.2	43.15	9	0
vortex	52.6	34.79	0	1
gap	59.4	31.03	0	0
gcc	205.8	1330.33	44	1
gnuchess-5.07	17.8	9.44	4	0
tcl8.4.14	17.9	266.09	4	4
hanterm-3.1.6	25.6	13.66	0	0
sed-4.0.8	26.8	13.68	29	31
tar-1.13	28.3	13.88	5	3
grep-2.5.1a	31.5	22.19	2	3
openssh-3.5p1	36.7	10.75	18	4
bison-2.1	48.4	48.60	4	1
openssh-4.3p2	77.3	177.31	1	7
fftw-3.1.2	184.0	15.20	0	0
httpd-2.2.2	316.4	102.72	6	1
net-snmp-5.4	358.0	201.49	40	20
binutils-2.13.1	909.4	712.09	228	25

08 5대 선도 프로젝트 중심

국가 성장동력 산업과 융합

시스템 SW 오류 자동 검증 도구셋 개발

웹 및 그래픽 SW 오류 자동 검증 도구셋 개발

지능형 로봇 SW 오류 자동 검증 도구셋 개발

무인 비행체 SW 오류 자동 검증 도구셋 개발

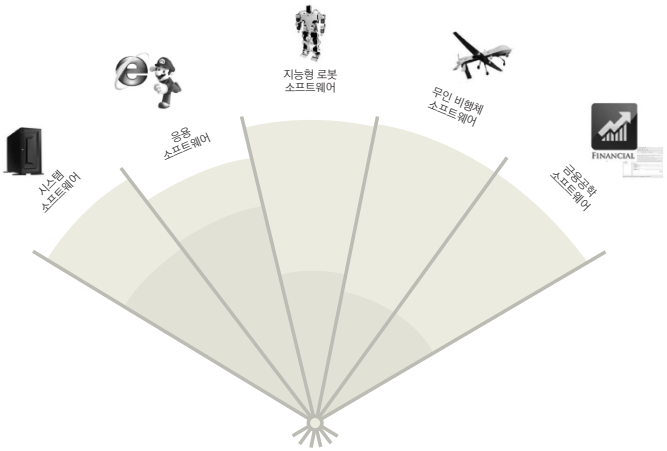
금융공학 SW 오류 자동 검증 도구셋 개발

09 총괄과제 구성 및 연계

1총괄 : 정통 첨단 기술팀 (leading analysis tech.)

2총괄 : 도메인 특화 기술팀 (domain-specific analysis tech.)

3총괄 : 이론 및 혁신 기술팀 (theory & innovative tech.)



오류 및 특질 검출기 (bug-finder, property-finder)
무결점 검증기 (verifier)

1총괄 : 정통 첨단 기술팀 (leading analysis tech.)

2총괄 : 도메인 특화 기술팀 (domain-specific analysis tech.)

3총괄 : 이론 및 혁신 기술팀 (theory & innovative tech.)

10 센터의 차별성

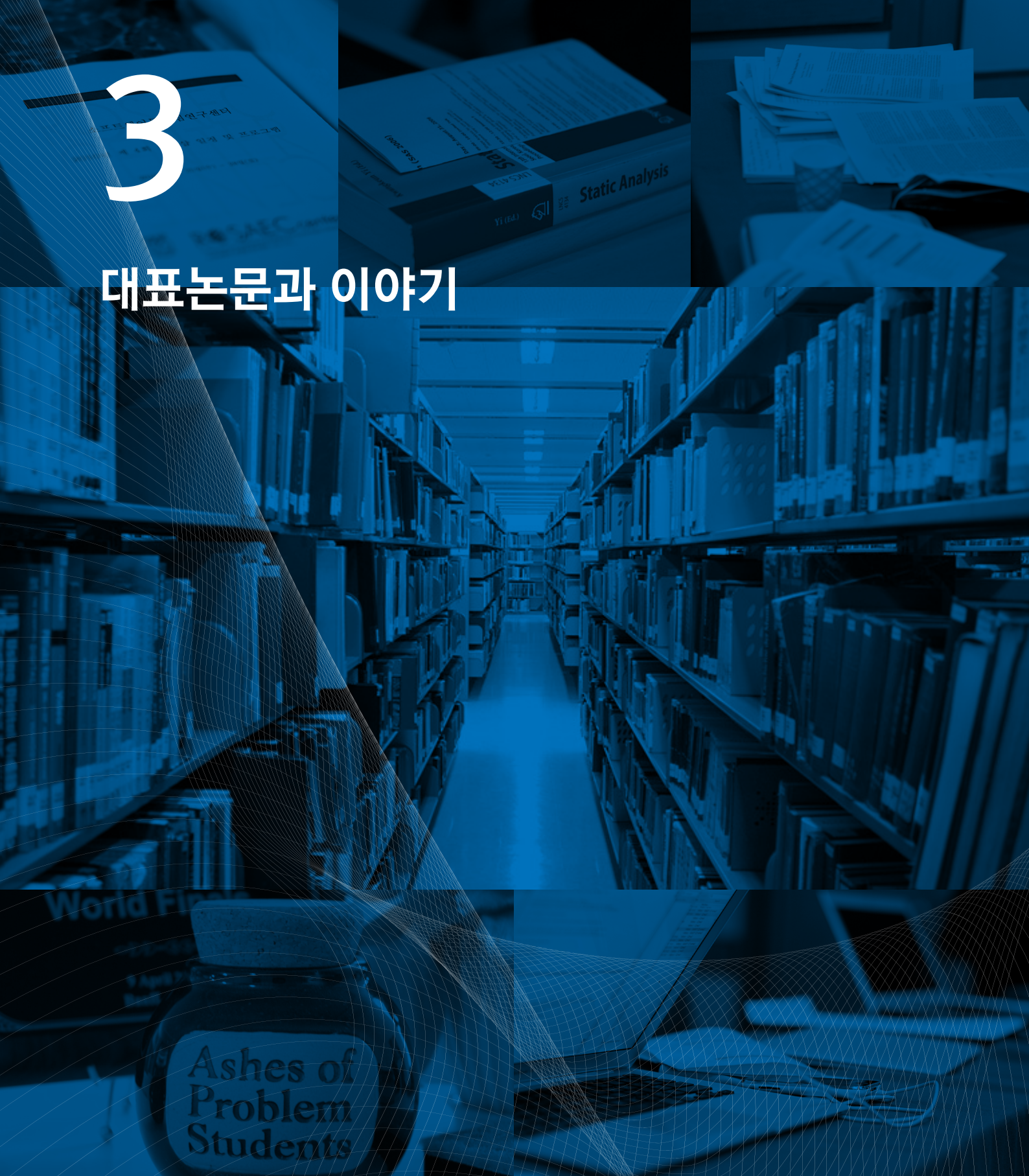
SW 원천 기술 연구센터
있는 기술의 조합, 산업화 센터가 아님

SW 실용기술 선도센터
이론 논문 센터가 아님

SW 고부가 산업창출 원동력센터
기존 산업 보조 센터가 아님

SW 융합,혁신 연구 추진센터
우물안의 센터가 아님

SW 기초 첨단 인력 양성센터
꽃을 가꾸는 센터가 아님



3

대표논문과 이야기

Static Analysis for Multi-Staged Programs via Unstaging Translation

Wontae Choi, Baris Aktemur, Kwangkeun Yi, and Makoto Tatsuta
POPL 2011: The 38th Symposium on Principles of Programing Languages

벽같은 문제였다. 프로그램이 프로그램을 만들고 실행시키는 경우(다단계 프로그래밍 multi-staged programming) 정적분석이 가능할까? 임의의 단계로 도망가며 다이나믹하게 생성되는 프로그램 텍스트들을 추적하면서 그 실행 의미를 어떻게 미리 분석할 수 있을까? 이 문제를 만나게 된 건 2006년도 POPL논문 덕택이었다. 다단계 프로그램의 타입 시스템을 고안하고 나서 든 의문이, 타입유추이상의 일반적인 정적분석은 어떻게 해야 할까? 였다.

석사과정이던 원태에게 이 문제를 던졌다. 벅찼겠지만 흥미있어 했다. 몇 달 간은 간단한 수준에서만 맴돌았다. 다이나믹하게 만들어지는 프로그램 텍스트를 예측하는 것은 되지만, 텍스트를 넘어서 그 실행의미까지를 예측하는 방법까지는 잡히지 않았다.

결정적인 날은 2009년 5월 29일이었다. UIUC에서 박사논문을 막 마친 Baris Aktemur가 자신의 학위논문을 내게 이메일했다. 우리의 2006년 POPL논문에 기초한 것이니 한 번 읽어달라고. 그의 박사논문은 다단계 프로그램을 보통의 프로그램으로 변환하는 부분이 주요 내용이었다. 무릎을 쳤다. 이게 가능했구나. Baris를 2009년 9월초 초대했다. 우리의 연구목표에 그의 변환이 핵심으로 쓰일 수 있다는 것을 논의하고 확인했다. 다단계 프로그램을 직접 분석하는 방법 대신에, 일단 보통의 프로그램으로 변환하고 기존의 분석기술로 분석한 후, 그 분석결과를 원래의 다단계 프로그램에 대한 것으로 바꿔주면 될 것이었다.

원태와 Baris가 디테일을 파보니 학위논문의 변환을 그대로 쓰기에는 헐렁한 점들이 발견되었다. 원태와 Baris가 그 헛점들을 메꾸기 위해 분투했다. 변환이 타입을 보존한다는 증명만으로는 부족했다. 원래 프로그램의 모든 실행과정이 변환후에도 그대로 보존되야 했다. 원태와 Baris는 고달픈 증명과정을 밟아갔다. 나는 원태에게

"증명의 모든 줄은 진심으로 확신하고나서 넘어가라"

는 원칙을 반복하며 독려했다. 2010년 1월 원태를 Madrid의 POPL에 데려가 Baris와 만나 연구진행을 확인하도록 했다. 모든 증명이 서서히 마무리 되어감을 느꼈다. 2010년 3월에는 순수남 Makoto Tatsuda 교수가 방문했다. 매년 봄이면 한국이 좋다고 식구들과 서울을 방문하는 친구였다. 우리의 변환장치를 보여주었다. 재미있어 했다. 돌아가서는 타입보존 증명을 들고 5월에 다시 찾아왔다. 우리의 증명보다는 약한 결과였지만 유용했다. 증명의 스타일도 좋았다.

한편 나는 분석결과를 변환이전의 프로그램에 대한 것으로 되돌리는 데 필요한 조건을 찾아보았다. 요약해석 이론의 틀속에서 쉽게 찾아졌다. 예제를 만들어 확인해 보았다. 작동했다.

이로써 모든 내용이 준비되었고 논문으로 엮는 것만 남았다. 하지만 원태와 Baris가 준비한 초고는 변환 파트 이외에는 미숙했다. 논문 제출 2-3주 전이었다. 논문쓰기에 석사생을 담금질하며 소모할 시간이 없었다. 내가 나섰고, 맹장수술 후 발목의 통풍이 한 달 이상 날 괴롭혔던 2010년 6-7월의 여름이었다. 논문을 꿰어가며 견디던 여름밤들이 지금은 추억이다. 의족을 사용하는 원태도 힘든 여름이었을 것이다. 날카로웠던 나였다.

2006년의 POPL 논문과 이번 논문으로 "다단계 프로그램"의 분석에서는 어느새 우리가 세계를 선도한다는 느낌이 들었다. 논문 리뷰의 한 구절도 그랬다:

"Thus, this paper can serve as a standard reference for further studies of static analysis of multi-staged programs."

String Analysis as an Abstract Interpretation

Se-Won Kim and Kwang-Moo Choe
VMCAI 2011: The 12th International Conference on Verification, Model Checking, and Abstract Interpretation

올해 3월, 예전의 아이디어를 다시 꺼내 쓴 논문입니다.

1. formalism의 일반화

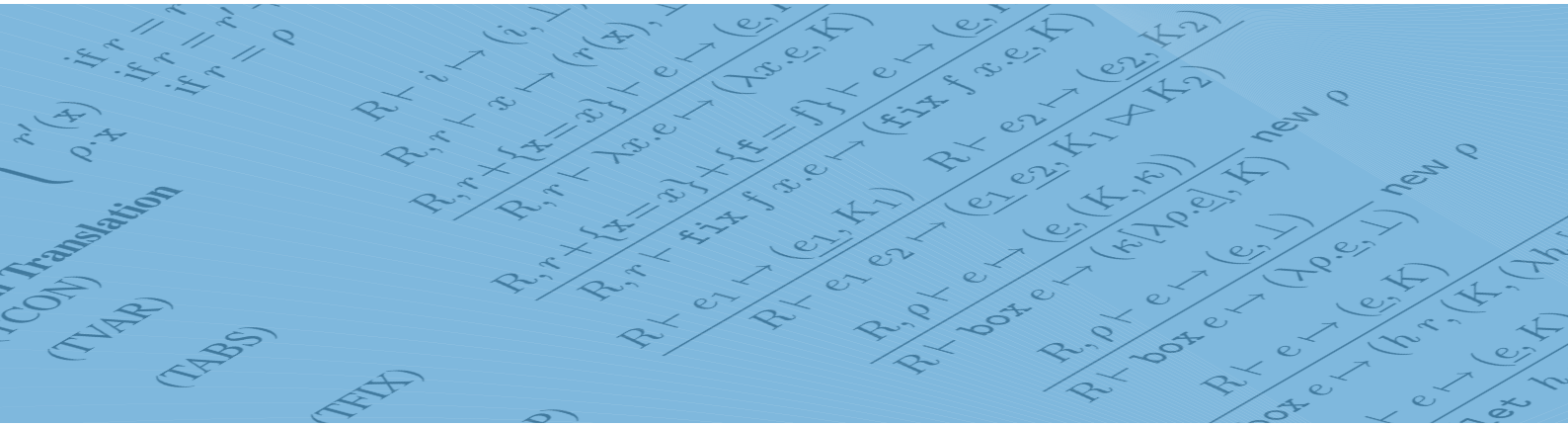
원래는 제약된 형태의 pushdown automata를 이용했었습니다. 이들의 표현력에 관한 정리를 찾을 수 없었고, 이 automata에 대한 알고리즘이 이해하기 어려웠습니다. 오토마타 개론에서 배운 형태의 pushdown automata를 이용하여 알고리즘을 다시 썼는데, 프로그램 분석을 하는 사람에게 더 이해하기 쉬운 형태가 되었습니다.

2. 타입 기반 분석 대신 요약해석을 적용

요약해석에 대해 많이 들어서 알고 있다고 생각했었는데, 새로운 도메인을 만드는 과정에서 제가 요약해석을 제대로 알지 못함을 깨달았습니다. 요약해석에 정통하기 위해 Cousot의 논문을 보며 주요 명제 하나하나를 증명하며 공부했습니다. 분석 도메인 요소의 표현력이 문맥자유언어 또는 그 이상이어서 좋은 성질의 도메인을 기대하기 어려울꺼라 생각했었습니다. 하지만, 수차례 수정을 거듭하면서 complete lattice가 얻어졌고, 가장 classical한 경우인, complete lattice간의 Galois 연결을 만들수 있었습니다.

VMCAI 논문 제출을 앞두고 약 한달간은 증명에 집중했습니다. 막히는 곳이 많았고, 물어볼 곳도 없어 답답했습니다. 이 논문 본문에는 나오지 않지만 증명에서 가장 핵심적인 보조 정리가 하나 있는데, 이 명제를 찾는 것이 가장 힘들었습니다. 증명을 위아래에서 접근하며 그럴싸한 명제를 만들고, 증명하다 실패하는 과정을 반복하였습니다. 실패를 거듭하던 며칠간은 밥도 잘 넘어가지 않았고, 실제로 몸무게도 줄었었습니다.

논문을 쓰는 동안 지도 교수님께서 두 번의 큰 수술로 부재중이셔서 조언을 얻지 못했지만 그 만큼 제게는 많은 공부가 된 논문이었습니다.



MeCC: Memory Comparison-based Clone Detector

Heejung Kim, Yungbum Jung, Sunghun Kim, and Kwangkeun Yi
ICSE 2011: The 33rd International Conference on Software Engineering (research track)

순수 국내 연구로는 최초로 소프트웨어공학 최고의 학회인 ICSE 연구 분야 (research track)에 채택된 논문입니다. 의미 분석을 통해 소프트웨어에 있는 코드 유사쌍들을 찾는 새로운 방법을 제안하는 논문입니다.

오랫동안 하고 싶어했던 연구를 좋은 사람들과 함께하고 좋은 결과도 얻어서 보람 있는 논문입니다. 2008년 겨울, 성균관대학교에서 열린 한국정보과학회 프로그래밍 연구회에 논문을 발표하러 갔었습니다. 한양대의 이욱세 교수님께서 소프트웨어 표절에 대해서 발표를 하실 때 질문을 했습니다. 표절을 찾는데 정적 분석기를 이용하는 연구는 없었냐고, 대답은 없다였습니다. 저는 함수가 일으키는 메모리 변화를 요약하여 메모리 누수를 탐지하는 분석기를 발표했었습니다. 그 함수 요약을 가지고 표절인지 여부를 판단하면 어떻겠냐고, 이욱세 교수님께 같이 연구하자고 제안했지만 거절당했습니다. 이유는 요약 정보가 너무 헐렁해서 함수 대부분이 비슷하다고 잘못 판단할 것 같다는 것이었습니다. 더 좋은 아이디어가 생기면 같이 하자고 했습니다.

2009년 삼성전자에서 김희정 학생이 연구실로 왔습니다. 회사에서 개발중인 소프트웨어에는 코드 유사쌍 (code clone)이 많이 있는데 소프트웨어 유지 보수를 위해 그런 유사쌍을 없애는 연구를 하고 싶어했습니다. 문제는 기존의 도구들은 코드의 생김새가 비슷한지만 검사를 할 수 있다는 것이었습니다. 소프트웨어 표절을 찾는데 분석기를 적용하고 싶어하던 저와 목적은 다르지만 연구 방향은 같을 수 있겠다 생각하여 그 해 10월부터 본격적으로 연구를 시작했습니다. 홍콩과기대의 김성훈 교수님께서도 소프트웨어공학 분야의 폭넓은 지식으로 도와주셨습니다. 처음에 함수의 메모리 변화 요약으로 비교를 하니 역시나 너무나 많은 함수들이 코드 유사쌍이라고 나왔습니다. 아예 요약 메모리 상태를 기억해서 비교하기로 했습니다. 비교 시간이 꽤 걸리긴 하지만 정확하게 나왔습니다. 비교 시간을 단축시키는 방법을 고안하여 구현했더니 분석 시간에 비해 상대적으로 무시할만한 시간에 비교를 할 수 있었습니다.

겨울 동안 논문을 써서 FSE 2010에 제출했습니다. 리뷰가 왔는데 제가 그동안 받아본 리뷰 중의 최고였습니다. 점수는 공개가 안 되었습니다. 다음은 2번째 리뷰의 결말 부분입니다.

> Overall, a very good paper.
> Point is favour:
> - novel approach
> - implementation done in a publicly available tool
> - data is publicly available
>
> Points against:
> - nothing, really

리뷰가 좋게 왔는데 떨어지니까 별로 리뷰가 도움이 되지 않았습니다. 실험을 더 강화하고, 우리 기술의 응용 방법을 더 추가하여 ICSE에 도전했습니다. 리뷰를 받았는데 붙을 건지 아닌지 확신을 할 수 없었습니다. 반박 (rebuttal)문을 준비해서 보내고 난 후에 희정 학생이 꿈을 꾸었는데, 제가 세 명의 남자를 때려눕혔답니다. 꿈 내용을 전해 들으신 이광근 교수님께서 ICSE에 붙는 꿈이라고 하셨습니다. 리뷰어가 세 명이었습니다. 교수님께서 예상하신 대로 ICSE에 붙었습니다. 재밌게도 김성훈 교수님께서 ICSE에 낸 논문 중에 세 편이 채택되었습니다.

– 박사과정 정영범 –

Access-analysis-based Tight Localization of Abstract Memories

Hakjoo Oh, Lucas Brutschy, and Kwangkeun Yi
VMCAI 2011: The 12th International Conference on Verification, Model Checking, and Abstract Interpretation

연구실에서 상용화한 프로그램 분석기인 Sparrow의 성능을 올리는 방법에 대한 논문입니다. 분석기의 동작은 실제 발생가능한 모든 메모리 상태를 모두 포섭하는 요약 메모리 상태를 가지고 프로그램을 요약해서 실행하는 것이라고 볼 수 있습니다. 그런데 실제 프로그램들을 분석할때는 이러한 요약 메모리의 크기가 굉장히 커지는데, 이것이 분석기를 느리게 만드는 주요 원인중 하나입니다. 필요한 것만 가지고 다니면 될 텐데, 문제는 분석해 보기 전에는 무엇이 필요한지를 미리 정확히 알아낼수가 없습니다. 따라서 무엇이 필요한지를 안전하게 예측하는 방법들을 사용하여 불필요한 부분들을 잘라내는 방법을 써야 합니다.

이 논문은 기존의 방법들보다 필요한 부분들을 더 정확히 알아내는 방법에 대한 것입니다. 사실 매우 간단한 아이디어지만 작은 프로그램을 분석할때는 잘 드러나지 않고 제법 큰 실제 프로그램들을 분석해 보아야만 드러나는 문제이기에 그 문제와 해결책을 처음으로 제시할 수 있었습니다. 지도교수님께서 늘 주변의 문제를 잘 해결하면 독창성은 따라오는 것이라고 말씀해 주셨는데 이를 직접 경험하게 해 준 논문입니다.

하지만 논문을 작성하는 것은 역시나 많은 에너지가 필요한 작업임을 다시 깨달았습니다. 디테일들을 나열하는 것은 쉬웠지만 그것을 한단계 위에서 정리하는 법, 그리고 딱 알맞은 정도로 표현하는 능력, 영어실력 등등의 부족함으로 인해 오랜시간이 걸렸고 또한 게재불가판정(reject)도 여러번 받았습니다. 그래도 돌이켜보면 이런 부족함들로 인해 많은 것을 배운 기회였고 많은 정성을 들인 논문이라 뿌듯합니다.

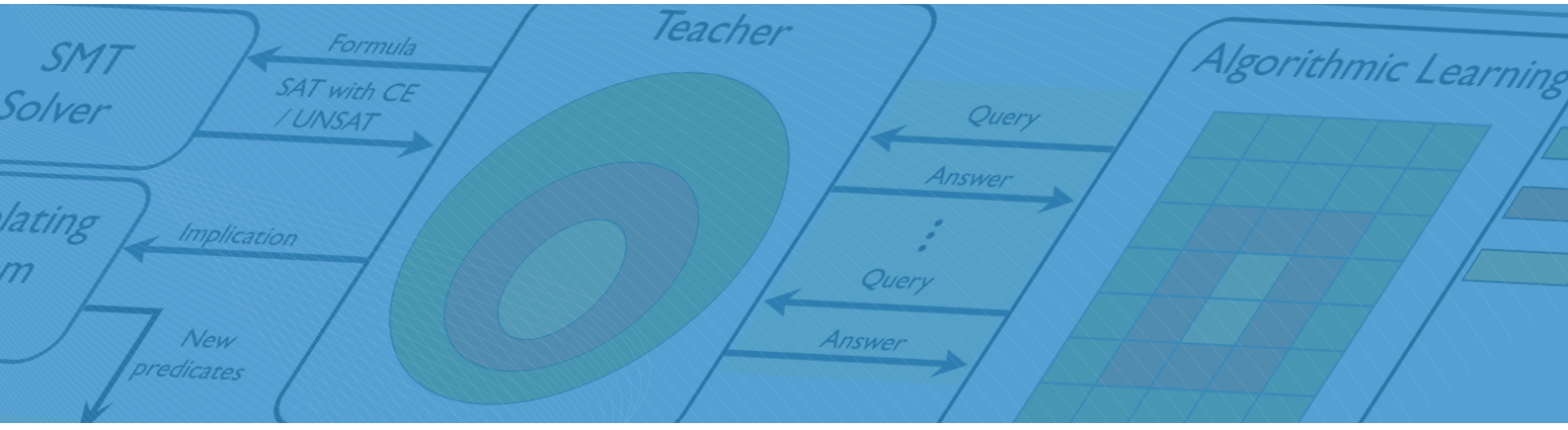
리뷰중:

"Overall, the paper is quite readable. The formulation is sufficiently but not overly formal. The theory balances well with the experimentation. And, the main idea is simple, easy to grasp and remarkably effective."

A Calculus for Hardware Description

Sungwoo Park and Hyeonseung Im
Journal of Functional Programming, 2010

몇년전 프로그래밍 언어 과목을 강의하면서 얻었던 아이디어를 발전시켜 진행한 연구의 최종 연구 결과물이다. 반도체 칩을 만들때 쓰는 프로그래밍 언어와 (반도체 칩 설계와는 전혀 무관해 보이는) 함수형 언어를 접목시키려는 노력에 대해서 공부를 하는 것도 흥미있었고, 아직까지 증명되지 않았던 근본적인 성질을 증명하는 것도 재미있었다. 반도체 산업에서 함수형 언어가 등장하고 있기 때문에 추후 관련된 연구도 진행할 수 있을 것이다.



Predicate Generation for Learning-Based Quantifier-Free Loop Invariant Inference

Yungbum Jung, Wonchan Lee, Bow-Yaw Wang, and Kwangkeun Yi
TACAS 2011: The 17th International Conference on Tools and Algorithms For The Construction and Analysis of Systems

여러 사람들과 어울려 연구하는 기쁨을 알게해준 내 첫 번째 학회 논문이다. 지난 7월 왕교수님이 다시 ROPAS를 방문하셨다. 학습 알고리즘을 사용한 반복문 불변식 유추 기법을 보강하기 위한 아이디어를 들고 오셨다. 구체화 하기 위해 처음 이 일을 시작했던 정영범, 공순호 연구원이 뛰어들었다. 그러던 중 유학으로 바빠진 공순호 연구원을 대신해 내가 참여하게 되었다. 선배님들의 훌륭한 팀웍을 익히 알고 있었던 터라 누가 되지 않도록 최선을 다했다. 팀 분위기에 녹아 들면서 연구가 점차 즐거워 졌고 좋은 결실을 맺을 수 있었다. 논문은 TACAS에 채택되었는데 유럽의 CAV라고 불리는 시스템 검증 분야의 최고 학회 중 하나다.

논문의 핵심은 반복문 불변식에 필요한 부품식(predicate)을 자동으로 유추하는 방법이다. 팀에서 불변식을 찾는 새로운 기법에 대한 논문을 지난 VMCAI 2010에 개제한 바 있다. 주어진 부품식의 조합을 탐색해서 그 가운데 불변식이 될 수 있는 것을 찾는 방법이다. 부품식 탐색에 학습 알고리즘을 사용한 참신한 발상으로 호평을 받았다. 논문에서 한 가지 다루어지지 않은 것은 부품식을 생성하는 방법이다. 필요한 부품식이 모두 프로그램 텍스트에 나타는 것이 아니기 때문에 특별한 방법이 필요했다. 아이디어는 서로 함축(implication) 관계에 있는 두 명제 A, B로부터 새로운 명제 I를 알아내는 내삽법(interpolation)을 이용하는 것이다. 부품식의 조합이 불변식이 되기 위한 여러 조건을 내삽하여 새로운 명제를 알아내고 그 명제에 포함된 부품식을 다음 탐색에 활용하도록 했다.

8월 한 달을 준비했던 VMCAI 2011에서는 고배를 마셔야 했다. VMCAI 2010의 후속 연구의 색깔이 짙어 이번에도 채택 되었으면 했다. 허나 준비가 부족했던 탓인지 안타깝게도 떨어지고 말았다. 리뷰는 주로 지난 VMCAI 2010에 제안된 기법 자체를 의문시 하는 내용이었다. 왕교수님은 리뷰가 지난 논문에 대한 것이었다면 받아들이겠지만 이번 것과는 상관없어 보인다시며 안타까워 하셨다.

심기 일전하여 도전한 TACAS 2011에서는 기분 좋은 소식이 기다리고 있었다. 논문을 수정하면서 BLAST와 비교하는 실험 결과를 포함시켰던 것이 유효했던 것 같다. TACAS는 모델 검증 논문이 많이 채택되는 학회라 사람들이 BLAST와의 비교를 관심있어 할 것이라고 예상했기 때문이다. 같은 프로그램에 대해 우리의 방법이 BLAST보다 더 적은 수의 부품식으로 불변식을 찾는 것을 보고 확신이 들었다. 초조해하며 기다리던 지난 12월, 논문이 채택되었다는 메일을 받았다. 거절(reject)하는 리뷰 없이 세 리뷰 모두 호의적이어서 더욱 기뻤다.

센터에 합류하고 처음으로 채택된 학회 논문이라 오래도록 기억에 남을 것 같다. 새벽까지 결과를 기다리며 메일 프로그램의 새로 고침 버튼을 수도 없이 누르던 기억이 아직도 생생하다. 채택 메일을 받고 건물이 떠나가도록 소리 질렀던 모습이 글을 쓰면서 떠올라 얼굴이 화끈거린다. 하지만, 이런 기쁜 일이 자주 있었으면 좋겠다.

– 연구원 이원찬 –

Directed Test Suite Augmentation: Techniques and Tradeoffs

Zhihong Xu, Yunho Kim, Moonzoo Kim, Gregg Rothermel, and Myra B. Cohen
FSE 2010: The 18th International Symposium on the Foundations of Software Engineering

작은 이야기: 대가의 면모

저널을 읽거나 학회를 참석하다 보면 자주 보이는 이름들이 있습니다. 아직 학위 과정 중에 있지만 놀라운 연구 실력을 보여주는 기대주들, 막 박사를 받고 독립 연구자로서의 능력을 유감없이 발휘하는 젊은 스타들, 그리고 90년대 혹은 그 이전부터 좋은 논문들을 많이 써 오신 대가들. 같은 연구자로서 이런 사람들은 어떻게 연구를 하길래 이렇게 논문을 재밌게 잘 쓸 수 있나 궁금할 때가 많았습니다. 그런 중에 University of Nebraska-Lincoln에 계시는 Gregg Rothermel 교수님과 함께 진행한 공동 연구는 좋은 경험이 되었습니다.

Gregg 교수님과는 작년 가을부터 올 3월까지 regression testing의 test-suite augmentation 의 성능과 효과에 영향을 미칠 수 있는 요소들을 분석하는 연구를 연구를 진행하였고 그 결과를 ACM FSE 2010 에 발표하였습니다. 특히, regression testing 과 통계적 분석, genetic algorithm 을 사용한 테스트 생성에 강점을 두고 있는 Gregg 교수님 연구실과 concolic testing과 formal analysis 에 강점을 두고 있는 우리 연구실 사이의 상승 효과가 돋보이는 연구였습니다. 특히 같은 내용이라도 문제 정의를 어떻게 하고 어떻게 글을 풀어 나가냐에 따라 논문의 내용이 크게 달라질 수 있다는 것을 Gregg 교수님과 같이 논문을 작성해 나가면서 느낄 수 있었습니다. 또한 우리의 강점을 발휘하여 체계적으로 실험 설계를 하고 연구 결과를 분석하여 헛점을 줄여 나갔습니다. 그 결과 FSE 리뷰어들로부터 'This is a very well done study...' 'The paper covers an interesting problem related to test suite augmentation in regression testing...' 와 같이 재밌는 문제를 잘 찾아 분석을 잘 한 연구라고 평가받을 수 있었습니다.

Gregg 교수님은 올 여름부터 약 1년간 연구연가로 KAIST에 머물고 계십니다. FSE 에 발표한 논문을 확장해서 더 일반화 시키는 연구와, 서로 다른 성질을 갖는 두 테스트 생성 기법이 어떻게 서로를 도와 나가 상승 효과를 극대화 시킬 수 있는지 알아보는 연구를 같이 진행하고 있습니다. Gregg 교수님과 연구하는 것은 항상 즐겁습니다. 대가의 통찰력을 느낄 수 있고, 놀라운 논문 작성 능력을 배울 수 있고, 또 같이 연구하는 사람들을 잘 이끌어가고 기분 좋게 해 주는 모습을 볼 수 있습니다. 지금 하고 있는 연구들도 좋은 결실을 맺고, 앞으로 이런 연구자가 되어야겠다고 다짐해봅니다.

iGraph: A Framework for Comparisons of Disk-based Graph Indexing Techniques

Wook-Shin Han, Jinsoo Lee, Minh-Duc Pham, Jeffrey Xu Yu, and Romans Kasperovics
VLDB 2010: The 36th International Conference on Very Large Data Bases

그래프 데이타베이스는 최근에 db community에서 각광받고 있는 토픽중에 하나이다. 이 중 그래프 인덱싱은 주어진 입력 그래프를 포함하는 데이타 그래프를 빨리 찾는데 사용되는데, 경북대학교 데이타베이스 연구실에서는 iGraph라는 새로운 그래프 인덱싱 프레임워크를 제안하였고 이는 분야 Top 컨퍼런스인 VLDB 2010에 발표되었다.

그리고, 한국에서 발표한 논문으로는 최초로 VLDB 2010의 best로 선정되어 VLDB Journal에 초청받았다. iGraph에서는 대표적인 모든 그래프 인덱싱 방법을 하나의 프레임워크에 구현하고, 이 방법들의 성능에 대해서 정성적으로 분석한다. 이러한 프레임워크는 향후 그래프 데이타베이스 분야의 후속 논문들에서 많이 사용될 것으로 기대된다.

Scala for Generic Programmers

Comparing Haskell and Scala Support for Generic Programming

Bruno C. d. S. Oliveira and Jeremy Gibbons
Journal of Functional Programming, 2010

This paper is about doing datatype-generic programming in Scala and it was published last year at the Journal of Functional Programming.

Together with Jeremy Gibbons (my former PhD advisor), I have showed that Scala (like Haskell) already provides all the essential features required for the development of DGP libraries.

Moreover, we have demonstrated that Scala has important advantages over Haskell when it comes to implementing DGP libraries. In particular, Scala supports a form of open datatypes, which can be used to support extensible generic functions in a more natural way. We believe that this work will provide important foundations for future developments of DGP libraries in OO languages and, hopefully, help the mainstream adoption of DGP techniques.

Malware Detection based on Dependency Graph using Hybrid Genetic Algorithm

Keehyung Kim and Byung-Ro Moon
GECCO 2010: The Genetic and Evolutionary Computation Conference

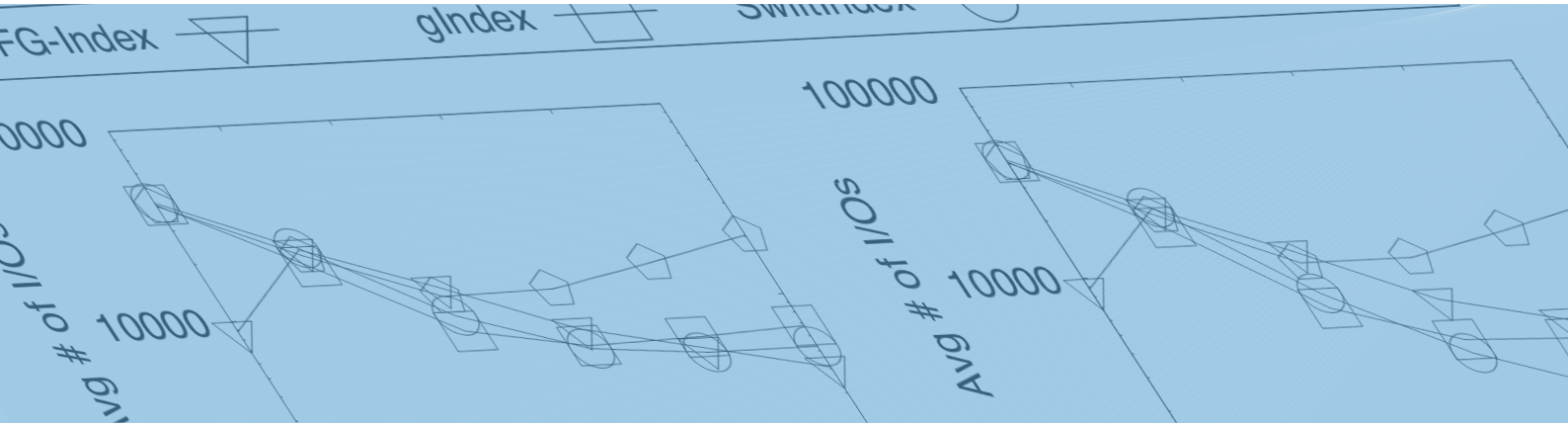
본 연구실에서는 2009년 7월 GECCO 컨퍼런스에서 발표된 바이러스 자동 생성 유전알고리즘 논문을 접하고, 변형 바이러스에 대한 대응책이 시급히 필요하다는 판단에 작년 8월부터 본 연구에 착수하였다. 약 5~6개월 정도의 연구결과를 집약한 본 논문이 2010년 7월 GECCO 컨퍼런스에서 발표되었다.

기존의 바이러스 감지 알고리즘들은 피해가 보고된 후 바이러스를 수배하여 이에 대한 백신을 만드는 방식이다. 본 연구에서는 알려지지 않은 다형변이로 만들어지는 바이러스를 감지하기 위해, 바이러스 검출 문제를 Maximum Subgraph Isomorphism 문제로 표현하고, 혼합형 유전 알고리즘을 사용하여 경험하지 않은 바이러스에 대해서 제로데이에 감지할 수 있도록 하는 것이다.

직접 제작한 바이러스를 포함한 총 18개의 다형변이 바이러스 데이터에서 기존 백신 소프트웨어의 검출률과 연구실에서 제안한 방법의 검출률의 비교를 수행했다. 그 결과 예상대로 기존 백신 소프트웨어들은 새로운 변형 바이러스의 검출에 취약한 모습을 보여주었고, 어느 백신도 본 연구실에서 제안한 방법의 검출률에 미치는 결과를 보여주지 못했다.

비록 아직까지는 작은 크기의 초보 수준의 다형변이 바이러스를 이용한 감지를 성공적으로 수행한 것이지만, 이를 바탕으로 강력한 감지 알고리즘을 만들고, 실험을 위한 바이러스를 자동으로 생성하는 알고리즘을 개발하며, 유전 알고리즘의 속도 문제를 해결하기 위한 다양한 접근법을 개발 및 시도하고 있다.

언제가 될 지 확언할 수는 없지만 현재의 바이러스들과는 전혀 성격이 다른 다형변이 바이러스들이 대량으로 출현하는 시기가 올 것이다. 만일 바이러스 안에 다형변이를 생성하는 기능이 들어간다면 걷잡을 수 없을 정도의 속도로 출현할 가능성도 있다. 본 연구가 다형변이 바이러스들을 감지하는데 큰 기여를 하기를 기대해본다.



Kripke Models for Classical Logic

Danko Ilik, Gyesik Lee, and Hugo Herbelin
Annals of Pure and Applied Logic, 2010

지금까지 몇 번 공동연구를 진행했었지만 이 논문처럼 재밌게 한 적이 드물다. 프랑스 인리아에서 포닥을 하면서 알게된 두 친구와 함께 진행한 연구인데 세 명의 저자가 각자의 지식을 적절하게 조합해서 이뤄낸 성과를 논문이 담고 있을 뿐만 아니라, 함께 연구를 진행하면서 서로의 의견을 진지하면서도 비판적으로 주고받을 수 있는 분위기를 끝까지 유지하면서 완성시킨 논문이기도 하다.

논문의 내용은 직관주의 논리 또는 양상논리 modal logic에만 적용된다고 알려진 Kripke semantics를 고전논리에도 적용할 수 있음을 보이는 내용을 관련 분야 내에서 최초로 담고 있다. 논문의 후속타가 가능하다고 보는데 지난 시간동안 다른 일에 집중하느라 진행시킬 수가 없어서 많은 아쉬움을 갖고 있으며 조속한 시기에 새로운 시도를 해서 다시 한 번 즐거운 공동연구를 다시 한 번 경험할 수 있기를 바랄 뿐이다.

QSkycube: Efficient Skycube Computation Using Point-Based Space Partitioning

Jongwuk Lee and Seung-won Hwang
VLDB 2011: The 37th International Conference on Very Large Data Bases

Skyline 계산에 dominance 개념이 주로 사용되었는데 incomparability를 활용하면 연산을 보다 최적화 할수 있음을 2009년에 발견하고 투고하였다. 슬프게도 같은 내용의 논문이 SIGMOD 2009년에 실려 소위 “scoop”을 당했다. Skyline은 마치 퍼즐처럼 누구나 듣고 쉽게 이해할 수 있는 문제다보니 도전자가 많아 scoop 당하는 일이 벌써 3번째이다. 하지만, 쌍생아 같아 보이는 두 논문도 언제나 다르고 내 자식만의 강점을 파고들다보면 오히려 더 튼튼한 논문이 되는 것 같다. 결국 scoop당한 자식은 EDBT 2010과 VLDB 2011에 각각 실리게 되었다. 리뷰는 아래와 같으며 이 두 논문에 대해서는 milestone 발표를 통해 이종욱학생이 소개할 것이다.

Overall, I positively evaluate the current work.
1) The problem statement is interesting, and 2) the proposed solution presents substantial improvements over the existing methods. 3) The paper is well-structured and well-written, and 4) the technical details easy to follow. 5) The experimental evaluation is adequate.

Type Classes as Objects and Implicits

Bruno C. d. S. Oliveira, Adriaan Moors, and Martin Odersky
OOPSLA 2010: The Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages and Applications

This paper was published at last years Object-Oriented Programming, Systems, Languages and Applications (OOPSLA) conference and it describes how the concept of type classes (originally from Haskell) inspires a useful programming technique in Object-Oriented (OO) Languages with generics (such as Java or C#).

Type classes were originally developed in Haskell as a disciplined alternative to ad-hoc polymorphism. Type classes have been shown to provide a type-safe solution to important challenges in software engineering and programming languages such as, for example, retroactive extension of programs. They are also recognized as a good mechanism for concept-based generic programming and, more recently, have evolved into a mechanism for type-level computation.

Together with Adriaan Moors and Martin Odersky, I have shown a lightweight approach to type classes in object-oriented (OO) languages with generics using the Concept pattern and implicits (a type-directed implicit parameter passing mechanism). Furthermore, I have also shown that Scala, a multi paradigm language that blends functional and object-oriented programming and supports implicits, supports many features that make the language ideally suited for expressing various types of generic programming.

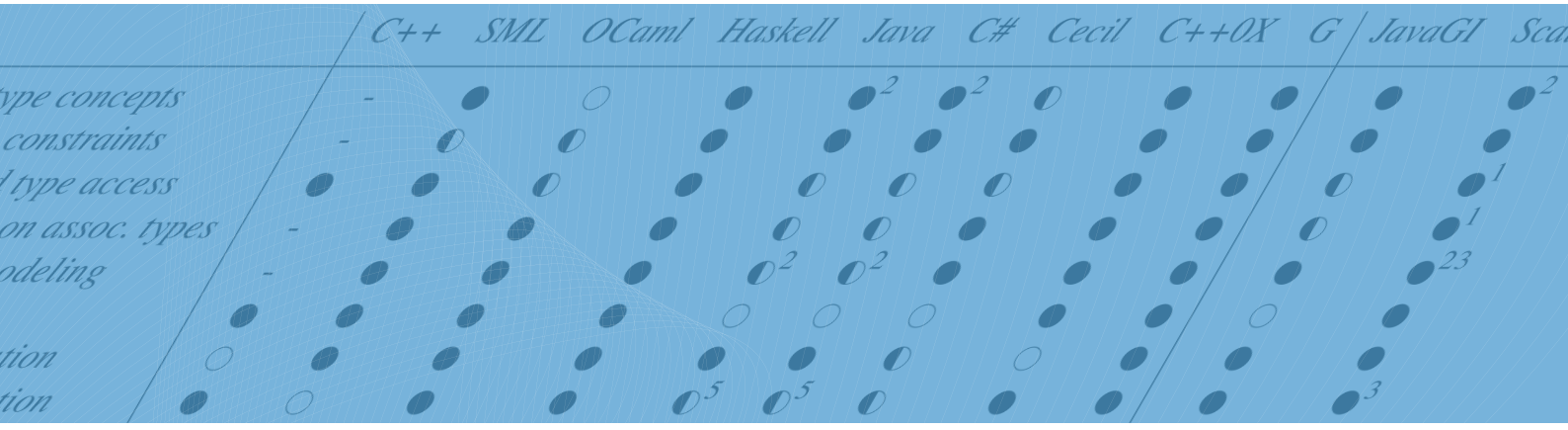
I was motivated to write this article because I think the concept of type classes is not well-understood by a lot of people. On the one hand OO programmers are puzzled by type classes declarations in Haskell and they wonder how it relates to OO programming. On the other hand Haskell programmers tend to have a misconception that type-classes are fundamentally different from classes in OO languages and that it is not easy to encode similar behaviour in OO. I think this article clarifies those issues and it also shows how implicits can effectivelly support type class programming in OO languages.

An Action Semantics Based on Two Combinators

Kyung-Goo Doh and David A. Schmidt
Semantics and Algebraic Specification, Lecture Notes in Computer Science, 2009

UCLA의 Jens Palsberg 교수로부터 이메일이 왔다. Peter D. Mosses 교수의 60세 생일이 되는 해를 기념하여 평생 연구업적을 기릴 헌정논문집(festschrift)을 기획하고 있으니 적극적인 참여를 바란다고 했다.

나도 오래동안 쌓아온 인연이 있기에 뭔가 공헌하고 싶은 마음이 생겼다. 그래서 쓰게된 논문이다. 논문을 만들면서 "Mosses abstraction"이란 신조어도 만들었다. 정말 괜찮은 환갑 선물이다. 영향을 받은 한 학자로서 보답을 했다는 뿌듯함도 느낄 수 있었다.



FortressCheck: Automatic Testing for Generic Properties

Seonghoon Kang and Sukyoung Ryu
SAC 2011: The 26th Symposium on Applied Computing

Haskell 언어의 랜덤 테스팅 라이브러리인 QuickCheck에 대해서는 익히 들어 알고 있었는데, 올 봄에 일본에서 열린 IFIP Working Group 2.8 미팅에서 John Hughes 교수님의 강연을 듣고 Fortress에 적용해보면 재미있겠다는 생각이 들었습니다. 학교로 돌아와 이제 막 연구실에 배정받은 강성훈 학생에게 Fortress와 QuickCheck에 대해서 이야기하고 몇 가지 아이디어들을 나누었습니다. 강성훈 학생은 Haskell, QuickCheck, Fortress에 대해서 하나씩 배워가면서, Haskell과 다른 Fortress 언어의 특징인 두 가지 polymorphism (subtype polymorphism과 parametric polymorphism)을 테스트하는 방법에 대해서 방향을 잡아가기 시작했습니다. 여름 동안 제가 보스턴에서 지내면서 원격으로 논문 작업을 같이 하고 ERC 여름 워샵에서 막바지 작업을 마치고 제출한 논문입니다.

EffectiveAdvice: Disciplined Advice with Explicit Effects

Bruno C. d. S. Oliveira, Tom Schrijvers, and William R. Cook
AOSD 2010: The 9th International Conference on Aspect-Oriented Software Development

This paper, published at last year proceedings of the Aspect Oriented Software Development (AOSD) Conference, is about a difficult problem in reasoning about programs: how to reason modularly about inheritance in the presence of effects?

In the paper we focus on advice, which is a form of inheritance widely-used in aspect-oriented programming (AOP) languages and which allows one program component to augment or modify the behavior of other components. When advice and other components are composed together they become tightly coupled, sharing both control and data flows. However modular reasoning about a component becomes very difficult; and two tightly coupled components may interfere with each other's control and data flows.

Together with Tom Schrijvers and William Cook, I proposed EffectiveAdvice: a disciplined model of advice that supports both modular reasoning and reasoning about interference. By modelling advice in a monadic purely functional language, we have shown that traditional reasoning techniques such as equational reasoning and parametricity can be used to prove properties about interference in a modular way.

Here's what one of the reviewers said about this paper:

> The problem tackled in that paper is probably the most important and difficult one AOP has to face. Even if the language considered is not full AOP (e.g. no pointcuts), it remains very expressive and allow modular reasoning. The approach and the use of interference combinators are very interesting and the results look impressive.

Sharp Thresholds for Hypergraph Regressive Ramsey Numbers

Lorenzo Carlucci, Gyesik Lee, and Andreas Weiermann
Journal of Combinatorial Theory, Series A, 2011

이론쪽 저널 논문 출판이 좀 오래걸린다. 원고 제출 후 6개월 내에 결정이 나면 초스피드이고 보통 1년이내에 되면 빠른 편이다라고 하는데 이 논문은 그러한 과정을 3번 거쳤다. 그러다보니 만 3년이 되어서야 출판 결정이 되었다. 논문의 주 내용은 수리논리의 증명론과 조합론과의 연관성을 밝혀 낸 내 박사논문의 일부 결과를 일반화 시킨 연구다. 박사논문 지도교수와 이태리 출신 친구와 함께 공동으로 진행한 연구인데 결과가 아주 좋다고 판단을 해서 수학쪽 탑10 안에 들어가는 저널에 제출했다가 1년 반만에 흥미롭긴 한데 분야가 논리라서 좀 특수하다라는 평아닌 평으로 거절당했다. 논문을 제출했다가 거절을 당할 수도 있지만 무려 1년 반을 기다리게 하고선 단 몇 줄로 '흥미롭긴 한데 좀 특수하다'라는 말로 논문평가를 내리는 건 무슨 심보일까 궁금하기만 하다.

두 번째로 선택한 저널은 랭킹이 한 단계 더 높은 데를 선택했다. 랭킹이 하나 더 높긴 했지만 우리 논문과 관련된 논문이 이미 출판된 역사가 있기에 시도한 것이지 오기에 그런 건 아니었다. 내 입장에선 논문 출판이 하나라도 빨리 되기를 바랬기에 처음엔 반대했지만 다른 두 사람이 강력히 원했기에 '그래, 이미 시간은 한참 지나갔다. 그래도 되면 대박아니겠어' 라며 동의 했다. 근데 1년여의 시간이 흘러 첫 시도와 비슷한 반응으로 거절 당했다. 이젠 좀 신경질이 났다. 도대체 몇 년을 이 논문 하나 때문에 시간을 보내야 하는지. 그래서 작년 초에 논문의 내용에 수정을 가하여 우리 논문에 가장 많은 관심을 보여줄 combinatorics 쪽 저널에 제출을 했다. 탑10은 아니지만 그래도 위에서 30 등 안에 드는 저널이다보니 경험상 이곳도 최소 1년은 기다리는 것으로 알고 있었기에 논문 제출 사실을 아예 잊고 지내기로 했다. 하지만 무슨 일인지 6개월 만에 긍정적인 결정이 났다. 그래도 좋은 저널에 출판이 되는 것으로 마무리 되어서 정말 다행이지 그렇지 않다라면 너무나도 아쉬운 논문이 될뻔 했다. 오는 2월 출판 예정이다.

SocialSearch: Enhancing Entity Search with Social Network Matching

Gaewon You, Seung-won Hwang, Zaiqing Nie, and Ji-rong Wen
EDBT 2011: The 14th International Conference on Extending Database Technology

DB연구자로 센터에 기여하고자 하는 방향은, 아주 많은 데이터가 주어진다면 기존의 모델기반의 방법론을 보완할 수 있는 새로운 직관을 제시할 수 있다는 것이다. 이 논문의 novelty는 웹 문서의 정보를 하나의 그래프, 소셜네트워크를 다른 하나의 그래프로 보고 매핑하면 웹 문서에 등장하는 인명과 소셜네트워크 ID를 매핑할 수 있다는 아이디어에 있다. 같은 방법으로 영어 문서의 정보를 하나의 그래프, 중국어 문서의 정보를 또 하나의 그래프로 보고 매핑하면 번역도 가능하다는 논문을 EMNLP 2010에 발표한바 있다. 이러한 데이터기반 방법론에 대해 워크샵에서도 소개할 예정이며, 2 Accepts와 1 Strong Accept로 게재 확정되었다.

This is a very interesting read, based on solid research, applied on real-world data. Matching graphs of social networks to identify individuals is not especially novel, but this is, as far as I know, the first time this is applied in conjunction with co-occurrence graphs extracted from the Web. Experiments are convincing. For all these reasons, I strongly support the acceptance of this paper.

Nonlinear Estimation for Spacecraft Attitude using Decentralized Unscented Information Filter

Jonghee Bae and Youdan Kim
ICCAS 2010: The International Conference on Control, Automation and Systems

"S/W는 오류검증기, H/W는 고장진단 알고리즘"

소프트웨어 오류 검증기는 말 그대로 S/W 개발자가 만든 코드의 오류를 찾아낸다. 그렇다면, 하드웨어에 오류(혹은 고장)가 있다면 어떻게 해야 할까? 현재, 이 이야기의 저자가 몸담고 있는 항공분야에서는 고장이 인명적, 비용적 측면에서 큰 손실을 가져올 수 있다.
물론 소프트웨어 검증기를 통해 S/W의 오류를 검증하게 되면 고장의 확률을 낮출수 있다. 하지만 하드웨어에 고장이 발생한다면, 이를 진단하고 분리하는 알고리즘이 추가로 필요하다.

이 논문은 틸트로터 스마트 무인항공기가 대상이며 항공기의 자세를 측정하는 센서에 고장이 발생할 경우, 센서고장을 검출하는 알고리즘에 대해 다룬다. 소프트웨어 검증기에 하드웨어 고장진단 알고리즘이 추가된다면, 정말로 'smart'한 틸트로터 스마트 무인항공기가 될 수 있을 것이다.

Structure Grid for Directional Stippling

Minjung Son, Yunjin Lee, Henry Kang, and Seungyong Lee
Journal of Graphical Models, 2010

컴퓨터 그래픽스 분야를 생각할 때 가장 먼저 생각나는 것은 역시 영화에서 자주 사용되는, 현실과 구별이 어려울 정도로 멋진 CG화면들이다. 실제로 컴퓨터 그래픽스 분야에서 많은 사람들이 집중적으로 연구해온 것은 보다 사실적인 영상을 만드는 방법이었다. 하지만 과연 사람들이 만들고 싶은 영상이 꼭 사실적인 영상들 뿐일까? 정말 그렇다면 사진과 인쇄술이 발달한 이 시대에도 예술가라는 직업이 존재하고, 사진 대신 풍경화나 초상화를 그려 걸어둘 이유가 없을 것이다. 이렇듯 사람들은 꼭 사실적인 사진이 아니더라도, 개성 있고 다양한 스타일의 그림들을 좋아한다. 하지만 예술가의 작품들은 만드는데 시간이 오래 걸리고 그 희소성으로 인해 누구나 쉽게 갖기가 어렵다.

다양한 스타일 중 신문이나 잡지에서 초상화를 위해 사용되는 헤드컷(headcut)의 경우, 점과 선만으로 이루어진 단순한 스타일임에도 불구하고도, 이들이 방향성 있게 나열되어 대상의 특징을 잘 표현해주는 인상적인 스타일이다. 그리고 그 특성으로 인해 인쇄술이 발달하여 굳이 사용할 필요가 없게 된 지금에도 계속해서 사용되고 있다.

하지만 이러한 헤드컷을 만들기 위해서는 숙련된 헤드컷 아티스트라 하더라도 3~4일이 소요되고, 따라서 신문이나 잡지에 나올 정도로 유명한 사람이 아닌 이상, 보통 자신의 모습을 이러한 헤드컷 스타일로 만들기는 거의 불가능하다.

우리가 개발한 기술은 이러한 헤드컷 영상을 사진 한 장만으로 누구나 쉽고 빠르게 만들어낼 수 있도록 한다. 이를 위하여 먼저 예술가가 그리는 헤드컷의 특징을 분석하고, 분석 결과에 따라 대상의 중요한 흐름을 정의하는 방법, 흐름에 따라 점이나 선의 위치를 정하는 방법, 그리고 실제 점이나 선으로 톤을 표현하는 방법을 정의하였다. 그리고 여기에 최신 GPU를 이용함으로써, 단 1~2초 이내에 멋진 헤드컷을 만들 수 있도록 하였다. 이 기술을 이용하여 유명한 사람이 아니라도 누구나 자신만의 특별한, 헤드컷 형태의 초상화를 가질 수 있게 되기를 기대한다.

4

협력연구 이야기

프로그래밍 언어와 소프트웨어 공학의 행복한 만남

방문객 김성훈
홍콩과기대 컴퓨터공학부 교수
방문기간 2008-06 ~ 11

ROSAEC 의 이광근 센터장님을 처음 뵙게 된 것은 필자가 MIT에서 포닥으로 일하던 시절이었다. 이교수님께서는 MIT를 3개월간 방문 중이셨는데, 너무나도 편안하게 대해 주시던 교수님이 좋아 교수님 연구실에서 포닥을 하겠다고 따라 나섰다. 교수님은 프로그래밍 언어 (Programming Languages, PL) 쪽을 연구하셨고 필자는 소프트웨어 공학(Software Engineering, SE) 전공이였는데도, 교수님께서 흔쾌히 6개월간 필자의 포닥 advisor가 되어 주셨다.

그러면서 ROSAEC이 시작되었고, 소속 교수님들, 학생들과 같이 공동으로 연구할 기회를 가질수 있었다. 필자가 소속된 이광근 교수님 연구실 학생들은 PL에 관련한 연구를 하고 있어서 필자에게는 새로운 것을 많이 배울수 있었다. 같이 생활을 하면서 SE에 있는 문제들을 PL의 기법을 사용하여 풀 수 없을까 학생들과 많이 고민 하는 시간을 가졌다.

그러던 중 이교수님 연구실 석사 과정에 있던 정가을 학생과 같이 버그리포트를 자동으로 적당한 개발자에게 할당하는 연구를 진행하였다. 버그가 개발자에게 할당이 된 다음에도 간혹 이 버그들이 다른 개발자들에게 재할당되는 관계에 착안, 개발자 간의 버그 재할당 소셜넷트웍 (bug-tossing graph)을 찾아 내고, 이 넷트웍을 이용하여 기존의 자동 할당방법보다 성능이 높은 알고리즘을 개발했다. 프로그램 관계 그래프나 호출 그래프등, PL분야에서는 그래프가 많이 사용되어고, 그런 그래프 모델을 버그 할당에 적용한 것이 연구의 핵심이였다.

이 연구 결과는 2009년 SE 최고의 학회중 하나인 ACM SIGSOFT International Symposium on the Foundations of Software Engineering (FSE) 에 발표 되었다. 이 때 사용되었던 tossing graph는 이미 다른 연구에도 많이 활용되어 ICSM 2010, ICSE 2010, CSCW 2011등에 tossing graph를 응용한 추가 논문들이 다른 연구자들에 의해 발표 되는 등 세계 SE 관련 커뮤니티에서 알려진 알고리즘이 되었다.

그 다음 해에 이광근 교수님 지도하에 김희정 석사과정 학생과 정영범 박사과정 학생이 Sparrow의 static analysis를 이용한 코드 클론 검출방법을 연구하였는데, 필자는 이 연구에 공동으로 참여하는 행운을 얻었다. 클론 검출은 SE에서 많은 연구가 이루어 졌는데, 이 연구가 PL기반의 static analysis 를 만난 것이다. 기존의 SE에서 주로 하던 클론 검출방법은 주로 코드의 겉모습의 유사성에만 기반한 것인데 반해, 우리의 연구는 프로그램 실행의미에 기반한 정적분석기를 가지고 코드 클론을 검출하는 방식이었다. 그 결과로 코드의 겉모습과는 관계 없이 같은 일을 하는 모든 클론을 찾아 낼 수 있게 해주는 획기적인 클론검출 방법이 탄생한 것이다. 이 연구 결과는 까다롭기로 소문난 SE 의 최고 학회인 International Conference on Software Engineering (ICSE) 2011에 발표될 예정이다. 이는 한국 첫 ICSE논문으로 기록될 것이다.

이 두 논문의 성공적인 사례는 PL과 SE와의 좋은 만남의 결과를 보여 준다. 더우기 이 두 논문의 실험과 작성을 주도한 1저자가 모두 석사학생이라는 점은 한국 대학원생들의 우수함과 그들의 가능성을 보여 주었다.

앞으로도 더욱 많은 PL과 SE의 행복한 만남들, 그리고 그런 만남을 통한 좋은 이야기들을 기대한다.

A Wonderful Summer with Monotonicity

Visitor Ben Lickly
Ph.D Student, EECS Department, UC Berkeley
Period 2010-05-13 ~ 08-09

As someone who is both a graduate student working on static analysis research and a Korean language learner, it was my good fortune to meet Professor Yi by chance at the MODELS conference in October 2009. He was very friendly and open to the idea of collaboration, and I felt even more encouraged after reading some of his papers. In particular, I found his paper on static analysis of the monotonicity of expressions ("Static Monotonicity Analysis for Lambda-definable Functions over Lattices", Murawski and Yi, 2002) very interesting, since I could easily see the applicability of the work to my research. One thing led to another, and I've ended up spending this summer at the ROSAEC lab as a visiting researcher.

At first I worried that the application that I had in mind would be too straightforward. Soon I worried that a single summer wouldn't be nearly enough time to complete all the revisions and extensions I had in mind. In the end, I've found the right scope of extensions to the monotonicity analysis that is useful for proving monotonicity of interesting expressions, but still completable in one summer. I'm proud of the final product that I've settled on and all the work I've achieved, and I'm glad that Prof. Yi has given me the opportunity to achieve it. He has accepted me into his lab and I really feel like one of his students here. Looking back, the three months I've spent here seem so short, and I wish I could stay longer. I know that if I could stay longer, it would be time well spent, and I hope that this isn't the last I see of Prof Yi, or any of the ROSAEC members who have made my summer here so wonderful.

프로그램 분석 프로그램

조규곤
파수닷컴 대표이사

프로그램을 분석하는 프로그램을 만들면 시장에서 기회가 있을까? 지난 2006년 서울대 ROPAS Lab과 프로그램 분석 기술의 상업화 고민에 빠져있었다. 과연 프로그램 분석 기술이 상용화할 수 있을 만큼 성숙된 기술인가? 프로그램을 분석하는 프로그램을 만든다는 것은, 이론적으로는 무한대에 가까운 분석 대상 공간을 여러 방법으로 줄인 다음, 근접한 해답을 찾는 것이라고 볼 수 있다. 계산복잡도 이론에서 본다면, 물론 분석 대상 프로그램에 따라 다르겠지만, 어느 규모 이상의 프로그램의 분석은 그 계산복잡도를 다항식으로 표현 할 수 없는 문제라고 보아야 한다. 학계에서야 NP 문제, NP-hard, NP-complete 문제들이 좋은 주제이지만, 산업계에서 보면 그런 문제를 해결하려고 했던 것치고 사업적으로 성공했던 경우가 거의 없다. 80년대에 학계뿐 만 아니라 산업계에서도 큰 붐을 이루었던 인공지능 기술의 상용화 실패가 그 대표적인 예가 될 수 있을 것이다. 불확실성은 배제된 정확한 규정과 그것의 효과적인 구현으로 인간의 단순 반복적인 데이터 처리를 대신해 주는 소프트웨어들이 사업적으로 성공해 왔었다.

그럼에도 불구하고 프로그램 분석 프로그램에 도전해 보기로 했다. ROPAS Lab 구성원의 열의와 그 간 변화된 컴퓨팅 환경을 생각해 보면 시도해 볼 만하다고 판단했다. 프로그램은 점점 많아지고 복잡해지고, 컴퓨팅 파워는 더욱 강력해지고 있다. 관련된 기술들도 꾸준히 발전해서, 초기 상태이기는 하지만 선발대들이 시장에 성공적인 진입을 하고 있다. 이런 기술들은 개발에서는 시행착오가 많을 수 밖에 없는 기술이기 때문에 먼저 시작했다고 해서 꼭 유리하지만은 않다. 앞서가고 있는 경쟁자가 있지만, 그 차이는 따라 잡을 수 있으리라 믿었다. 얼마나 실 수요자와 밀접하게 협력을 할 수 있느냐가 중요하리라고 생각했다. 비록 제공하는 기능이 제한적일지라도 가격 이상의 가치를 제공 할 수만 있으면 오히려 사업하기 좋을 수 도 있겠다는 생각이 들어 결심할 수 있었다. 4년이 지난 지금, 그 동안 쉽지만은 않았으나 시장에서 우리 포지션을 찾아 가고 있다. ROPAS Lab과의 협력 관계도 산학 협력의 좋은 모델이 되고 있다.

그런데 지난달 미국 Orlando에서 열린 Gartner Symposium에 다녀오면서 우리가 하고 있는 프로그램 분석 기술 및 시장에 대하여 새로운 생각을 할 수 있었다. 매년 Gartner에서는 향후 5년 동안 IT기술 및 산업의 변화를 예측하고 가이드라인을 제시하는 Symposium을 열고 있는데 올해는 행사 20주년을 맞이하여 향후 20년을 주도할 기술 트렌드 4가지를 제시하여 주목을 끌었다. 그 중 셋은 “Cloud Computing”과 “Social Computing”, “Context-aware Computing”으로 어느 정도 예상할 수 있는 것들이었는데 마지막 한가지는 예상 외로 “Pattern Based Strategy”라는 트렌드를 꼽았다. 앞으로는 더욱 더 많은 정보가 쌓이게 될 것이고 이 많은 정보로부터 어떻게 가치 있는 pattern을 찾아 낼 것인가가 기업의 경쟁력을 결정할 것이라는 것이었다. 물론 관련 기술이 발전하리라는 예상을 내 놓았다. Statistical analytics, clustering, segmentation, unsupervised learning, pattern matching, pattern recognition, rule induction, neural networks등 통계학이나 인공지능 관련 여러 기술들이 발전 활용 될 것이라고 한다. 대부분 한 동안 시장에서 현실성이 떨어지는 기술로 분류되던 기술들이다. 많은 사람들이 쉽게 수긍하지 못하고 있었다. 이런 예상을 하게 된 배경에는 우선 엄청나게 많은 정보로부터 가치를 창출해 내고자 하는 시장의 요구가 있기 때문이며 Cloud Computing 덕택에

언제나 거의 제한 없이 컴퓨팅 파워를 쓸 수 있고, Context-aware Computing, Social Computing에 의한 좀더 의미 있는 정보를 활용 할 수 있기 때문이라고 설명하고 있다. 환경의 변화가 그간 산업계에서는 금기 시 하던 어려운 문제를 푸는 소프트웨어의 상용화 노력을 촉구하고 있는 것이다. 프로그램 분석 기술도 이런 부류에 속하는 기술이고 그 가능성을 우리는 보아왔다. Gartner의 주장을 좀 더 다른 느낌으로 받아 들일 수 있는 이유였다.

분명 지난 2-30년 간 소프트웨어 시장의 주류는 OS, database, network, business applications등으로 대별할 수 있는 소프트웨어들이었다. 이런 소프트웨어들은 크기는 크지만 계산복잡도라는 면에서 보면 쉬운 문제를 푸는 소프트웨어들이었고, 앞으로는 계산복잡도가 어려운 문제를 풀려고 하는 소프트웨어가 시장에서 각광을 받을 것이라는 것, 그런 개발을 시도하는 회사들에게 기회가 많을 것이라는 것을 쉽게 예상해 볼 수 있다. 여기서 한걸음 더 나가면 어려운 문제를 푸는 소프트웨어를 개발하기 위한 개발 방법론은 기존의 개발 방법론과는 좀 달라질 필요가 있다. 정확한 설계와 그에 따른 테스트에 기반한 개발을 수행하기가 쉽지 않기 때문이다. 좀더 많은 창의적인 아이디어를, 현장으로부터의 피드백을 제품에 빨리 수용할 수 있을 것인가가 더욱 중요할 것으로 보인다. 소프트웨어 개발 패러다임의 변화도 필요해 보이는 부분이며 많은 소프트웨어 회사의 부침도 예상된다. 우리회사가 프로그램 분석 프로그램, Sparrow를 만들어 가는 과정에서 겪는 시행착오를 보면서, 그러면서 만들어오고 제품의 가치를 보면서, 이런 기술을 개발해야 하는 당위성뿐만 아니라 성공 가능성도 같이 예감하게 된다. 프로그램 분석 프로그램, 어려운 일이지만 충분히 도전해 볼 만한 가치가 있는 일이다.

Fasoo.com PA Story

박영미
파수닷컴 PA 사업부 팀장

PA DIVISION

파수닷컴의 PA(Program Analysis)사업본부는 2006년 말 서울대 프로그래밍 연구실의 연구 결과를 상용화 하는 것으로 시작하였습니다. 기업문서보안이 주된 사업내용이었던 파수닷컴으로서는 쉽지 않은 의사결정이었고, 그만큼 많은 기대와 염려 속에 출발하였습니다. 하지만 매년 100%가 넘는 성장률을 기록하며, 작년에는 매출 13억을 이루었습니다. 2011년에는 매출 20억을 목표로, 우리 기업들의 SW 품질 향상에 기여하며 그렇게 PA 사업본부는 성장하고 있습니다.

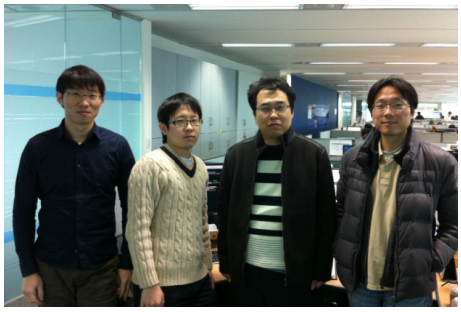
PA WORK

SPARROW for C/C++는 그 동안 많은 변화를 거쳐 지난 8월에는 version 3.0이 출시 되었습니다. 3.0 버전은 C++를 확장 지원하며, ▲오탐율 10% 미만 ▲분석 속도 5배 증가, ▲리포팅 기능 강화, ▲50% 이상 메모리 사용량 감소 등, 기존 버전 대비 사용성과 효율성이 대폭 향상 되었습니다. 또한 지난 2010년 9월에는 'SPARROW 3.0'을 서울대에 기증하였는데, 이는 학교에서 시작한 기술을 기업이 성공적으로 상용화하고, 이를 다시 학교에 기증한 모범적인 사례로 주목을 받고 있습니다. 2011년에는 그 동안 미루어 왔던 concurrency checker등을 포함하여 보다 다양한 checker들이 추가 될 예정이며, 개발자 PC에서 독립적으로 실행 가능한 standalone version도 준비 중에 있습니다. 이를 통하여 모든 개발자들이 개발 초기부터 프로그램 오류를 수정할 수 있도록 하겠다는 의미의 "SPARROW EVERYWHERE"라는 우리만의 motto를 실현해 갈 것 입니다.

지금 SPARROW v3.5 release를 위해 대준전임, 경철전임, 경원씨, 원석씨, 희제씨가 "화이팅" 중 입니다.

SPARROW for Java도 2011년 1Q 출시를 목표로 준비 중 입니다. 그 동안 JavaFinch라는 code명으로 개발 되어온 SPARROW for Java는 C/C++ version과는 달리 syntactic / semantic engine을 모두 갖추고 있으며, 기존의 run-time error 뿐만 아니라 다양한 보안 취약점의 정적 분석 기능도 제공 예정 입니다. 분석기를 실제 기업에서 유용하게 사용할 수 있는 도구로 만들기 위해서는 이론에 여러 가지 조미료들을 첨가하여 오탐률을 낮추고 알람에 대한 설명을 이해하

기 쉽도록 해주는 등의 작업들이 중요합니다. 함수간에 전달되는 값들의 분석 부담을 최소화하면서 분석하는 것도 고려해야 하지요. JavaFinch는 요즈음 많은 각광을 받고 있는 mobile application의 안정성 보증이라는 측면에서 appstore 운영자 들에게 많은 관심을 받고 있습니다. 아마 올해 말 쯤에는 T-store나 Samsung App에서 제공하는 utility program에 JavaFinch 기반의 도구가 들어 있지 않을까 합니다.



Java Finch는 Java Sparrow라고도 불리우는 자바섬에 사는 새의 이름입니다. 사람을 좋아해서 사람 손안에 앉아 편안히 노는 이 새처럼 Java Finch가 Java programmer들을 편안하게 해 주길 바라는 마음으로 2011년 3월 제품 출시를 앞두고 진팀장과 희종전임, 규항전임, 강선임, 이선임이 매일 매일 "몰입" 하고 있답니다.

BADA와 협업

파수닷컴의 PA사업본부는 지난 1년 동안 삼성의 자체 OS 기반 스마트 폰으로 알려진 삼성의BADA 개발진과도 많은 일을 같이 해 왔습니다. 안전한 앱 개발을 위해 개발자들이 자신이 권한이 있는 BADA API만을 사용하는지 검사 하는 등 우리의 프로그램 분석 능력을 활용하는 일이었지요. 올해에는 SPARROW for C/C++를 BADA IDE에 탑재하는 계약이 이루어져, 상반기 중에 완료 될 계획이랍니다.

PA PEOPLE

Mountain biker PA 사업본부에서 누구를 소개하면 좋을까 생각을 하니, 떠오르는 사람이 너무 많습니다. Hand drop coffee의 달인으로 본부내에서 SPARROW Café 성업 중인 규항, 경원, 얼마 전 발리로 신혼여행을 다녀온 희종전임, 2년째 스윙댄스에 빠져있는 희제, 직업도 개발 취미도 개발인 경철전임... 요즘 한창 당구 재미에 신이 난 당구50 이선임... 그래도, 사업부 팀장으로 합류한 지 얼마 되지 않은 시간 동안, 많은 도움을 주고, 이야기를 나누고 고민을 나누고 있는 개발부 팀장인 진민식 책임을 잠시 소개하려고 합니다.

진민식 팀장은 곧 15명의 개발팀을 이끌어 가야 하는 막중한 책임을 가지고 4년째 PA사업본부의 대들보 역할을 하고 있습니다. 특유의 인자함이 업무에서도 묻어 나와 팀을 부드럽게 끌어가고 있어, 진팀장의 팀원이 되는 것도 나쁘지 않을 것 같습니다. 요즘 유지보수 없는 직업에 대한 강한 동경을 유지보수 필요 없는 완벽한 SW 만드는 고민으로 쏟고 있는 것 같아 조금 더 든든하기도 합니다. 3년째 산악자전거로 온 산을 누비고 있는 PA사업본부 유일의 뱃살 없는 사람이기도 하지요. 출퇴근을 자전거로 시작한 것이 계기가 되었다고 하는데, 올라갈 때는 등산의 재미가 그리고 내려올 때는 스키 탈 때의 재미가 있다는 표현이 좀 멋진 것 같기도 합니다. 파수와 참 닮은 사람이라는 생각이 듭니다.

Fasoo Year-end party

Fasoo year-end party는 매년 12월 29일에 사내에서 팀을 정해 음식을 만들어 서로 나눠먹는 송년 파티입니다. 사원만 참여하는 것이 아닌, 사장님부터 상무님, 부장님 모든 임직원이 참여하는 행사로 다른 회사와는 차별화된 우리 파수만의 기업문화라고 할 수 있습니다.

이번 컨셉인 "한식의 슈퍼스타"는 세계적인 소프트웨어 기업으로 거듭나기 위해 가장 중요한 과정인 현지화(Localization)의 중요성을 강조하기 위한 뜻이 담겨 있습니다. 작년에 best chef상에 빛나는 경원 chef의 진두지휘에 따라 이번에 PA사업본부가 준비한 메뉴는 "된장 소스를 곁들인 삼겹살 퓨전 샐러드"와 "무쌈" 입니다. chef의 비법 소스로 160명 파수인의 입맛을 사로 잡았지요.

그리고 Fasoo year-end party는 또한 파수 가족들의 기증품으로 벼룩시장을 열어 아름다운 재단에 기부를 하고 있어, 연말을 매우 뜻 깊게 보낼 수 있도록 하는 행사이기도 한데, 이번에도 100개가 넘는 물품이 모아져 수익금과 함께 전달되었습니다.

PA VISION

PA 사업본부는 갈수록 복잡해 지는 프로그램의 다양한 성질을 잘 분석하여 제공함으로써, 개발자들이 좋은 품질의 프로그램을 쉽고 편하게 개발 할 수 있도록 하고자 합니다. SPARROW emblem이 세계 모든 개발 회사에 찍히는 그 날까지 열심히 달려 보고자 합니다.

SPARROW SCANNED

SPARROW Flies in 2011~!!

Invariant Generation through Learning

Visitor Bow-Yaw Wang
Associate Research Fellow
Institute of Information Science, Academia Sinica
Period 2010-06-12 ~ 08-19

Computer programs are infamous of their unpredictability. An unintended typo or a missing case may lead to disasters. One of the best known software failures is the explosion of Ariane 5 on her maiden flight. On June 4 1996, the Ariane 5 launcher exploded in the air 40 seconds after initiation of flight sequence due to an overflow exception. Ariane 5 used the same software in its flight control system as in its predecessor Ariane 4. Ariane 5 however had a velocity 5 times higher than Ariane 4. The high velocity was not expected by the old software and caused overflow in the new launcher.

To ensure program correctness and prevent software failures, formal methods have several techniques to offer. To illustrate how formal methods help program development, consider the annotated while loop in the following code:

```
[ i == 0 ]
while (i < 10) {
  b = rand() % 2;
  if (b) i++;
}
[i == 10 && b]
```

The variable `b` can have 0 or 1 in the beginning of the loop. If it is 1, the variable `i` is incremented by 1. The first square bracket `[ i == 0 ]` states the assumption of the loop: `i` is 0 before entering the loop. It is called a *precondition* of the loop. The second square bracket `[ i == 10 && b ]` states the results of the loop: `i` is 10 and `b` is not 0 after executing the loop. Similarly, it is called a *postcondition* of the loop. In formal methods, programmers are required to prove that the postcondition (`i == 10 && b`) must hold after executing loop if the precondition (`i == 0`) holds before entering the loop.

One way to prove the correctness of the while loop is to find an invariant. An *invariant* is a property that is preserved in each iteration. Moreover, it also proves that the postcondition must hold after executing the loop. In our example, `(i < 10) || (i == 10 && b)` is an invariant. Clearly, it is true at the beginning of the loop; it is also true at the end of the loop in each iteration. Moreover, it proves the postcondition because `i < 10` does not hold after executing the loop. Since an invariant exists, the loop is correct.

Although invariants ensure the correctness of while loops, finding them is not easy. It often needs

insights to programs. When pre- and post-conditions are more complicated, even programmers may not find invariants easily. If tools can find invariants automatically, programmers only need to specify pre- and post-conditions without actually proving them. Invariant generation is hence an important problem in program correctness.

Observe how programmers infer invariants. Initially, they have their guesses of invariants. If their guesses are too strong, programmers weaken them; if they are too weak, programmers strengthen their guesses. The process is very similar to learning. Programmers present their guesses. They use a tool (or metaphorically, a teacher) to justify their guesses. If the tool finds counterexamples, programmers will refine their guesses and repeat.

During my visit to ROSAEC Center this summer, Professor Kwangkeun Yi, Yungbum Jung, Soonho Kong, and I develop a new technique for invariant generation. Based on the previous observation, we apply machine learning to the invariant generation problem. We use a learning algorithm to play the role of a programmer. Moreover, we devise a teacher which answers queries from the learning algorithm. By interacting with our teacher, the learning algorithm is able to generate invariants for some Linux device drivers and SPEC2000 benchmark programs with complicated pre- and post-conditions.

Our approach is very simple and flexible. The hardest part (that is, inferring invariants) is done by the learning algorithm. We only need to define a teacher which answers queries. Of course, more sophisticated teachers can be used in our approach. In our experiments, however, we find that it may not be necessary. Our teacher always gives a random answer if it cannot resolve a query conclusively. In one of the Linux device driver programs, we find that the learning algorithm successfully generates an invariant regardless of random answers. There are sufficiently many invariants that a few random answers could not prevent the learning algorithm from finding one.

Machine learning has been applied to many areas. Our work suggests that it may be useful to program analysis as well. In the future, we plan to explore its applications in proving program correctness. We also invite our colleagues to work on this exciting new idea. For further information, please contact me or Professor Yi at ROSAEC Center, Seoul National University.

산학연 협력

한국저작권위원회 코드잼 서비스 개발

한국저작권위원회(KCC)와 소프트웨어 무결점 연구센터가 공동으로 연구개발한 워터마킹 및 코드 난독화 기술에 기반한 코드잼 서비스가 2010년을 빛낸 IT혁신 상품으로 선정되었다. 코드잼은 소프트웨어 소스의 의미구조(semantics)에 원저자 고유의 이미지가 만들어지도록하는 소스를 뿌려놓고 정적분석기술을 이용해서 그러한 소스를 찾아내어 원저자를 규명해내는 새로운 기술이다.

파수닷컴, 서울대에 SPARROW 기증

2010년 9월 15일, 파수닷컴(대표 조규곤)이 서울대학교 컴퓨터공학부(학부장 신영길)에 1억4000만원 상당의 정적분석 시스템 '스패로우(Sparrow)'를 기증하였다. Sparrow시스템은 서울대학교 프로그래밍 연구실의 연구결과를 산용화한 것으로, 시장의 요구사항에 맞게 새롭게 업그레이드된 후 다시 학교에 기증된 사례가 되었다.

ROSAEC, KAOC, and CEEDS Joint Workshop

2010년 4월 12일, 한국인공장기센터(KAOC)의 인공심장 소프트웨어에 대한 고신뢰 융합소프트웨어 공학센터(CEEDS) 및 소프트웨어 무결점 연구센터(ROSAEC)에서의 테스트 결과 및 분석 결과에 대하여 토론하는 워크샵을 개최하고, 3개 기관이 연구 협력에 대한 양해각서(MOU)를 체결하였다.

5

세미나와 워크샵

세미나

2010-12-22 **TaintDroid: An Information-Flow Tracking System for Realtime Privacy Monitoring on Smartphones**
Byung-Gon Chun, Intel Labs Berkeley

2010-11-19 **Robot Skill Synthesis through Human Sensorimotor Learning**
Jan Babic, Jozef Stefan Institue, Slovenia

2010-11-11 **String Analysis as an Abstract Interpretation**
Se-Won Kim, KAIST

2010-11-04 **Abstract parsing with string updateds using LR-parsing technology**
David Schmidt, Kansas State University

2010-11-04 **Namespace-based Language Modelling**
David Schmidt, Kansas State University

2010-09-03 **A Safety-Assured Development Approach for Real-Time Software**
Eunkyoung Jee, University of Pennsylvania

2010-04-15 **Type Checking with Open type Functions**
Tom Schrijvers, Katholieke Universiteit Leuven, Belgium

2010-04-15 **Strictness Meets Data Flow**
Tom Schrijvers, Katholieke Universiteit Leuven, Belgium

2010-04-12 **ROSAEC, KAOC, and CEEDS Joint Workshop**
Sungdeok Cha, Sehun Jeong, Kwangkeun Yi, and Sungkeun Cho, CEEDS and ROSAEC Center

The 5th ROSAEC Center Workshop

기간: 2011-01-06 ~ 01-09
장수: 경상남도 통영 마리나리조트

Workshop 개회 및 센터 활동 보고
이광근, 서울대학교
좋은 발표란 무엇인가?
정영범, 서울대학교
Frama-C 프로그램 검증 시스템 소개
박종현, POSTECH
STAR : Stack-Trace Based Automatic Crash Reproduction
김성훈, HKUST
빠른 3*3 행렬 곱셈법을 자동으로 찾아내기
김진, 서울대학교
Simulation Walls for System Architects
김장우, POSTECH
PLRG @ KAIST
류석영, KAIST
스마트시대, 정보화 혁명 그리고 우리의 기회
최진영, 고려대학교
Analysing Information Spreading in Complex Networks
정교민, KAIST
스카이라인 빠르게 계산하기
이종욱, POSTECH
Implementing String Analysis for PHP Program
최재준, KAIST
ACSR과 Statechart를 이용한 소프트웨어 명세
황대연, 고려대학교
필요한 정보만 가지고 필요한 지점만 분석하기
오학주, 서울대학교
이름이 같은 함수들의 안전성을 Coq으로 증명하기
김지응, KAIST
코드 클론에서 잠재적 라이선스 위반에 관한 연구
이상훈, POSTECH
편리하고 확장 가능한 분석기 생성 도구 만들기
이승중, 서울대학교
PLaSSE Research and Education
도경구, 한양대학교
Boolean BI 논리를 위한 컷-제거 귀추 계산법
박성우, POSTECH
Implicit Programming
Bruno C.d.S. Oliveira, 서울대학교
For More Usable This Type
나현익, KAIST
Program Analysis for Overlaid Data Structures
정승철, 한양대학교
ROSAEC Center Software Clinic Service : “주변의 병든 소프트웨어를 치료해주세요”
신재호, 서울대학교

다단계 프로그램에서 불필요한 변수 제거하기
조성근, 서울대학교
QNX RTOS 기반 비행제어SW 및 고장사례 분석
한용수, 서울대학교
언어를 이용하여 정보 흐름 파악하기
박창희, KAIST
두 마리 토끼 잡기 – 정적 타이핑과 동적 타이핑의 만남
강성훈, KAIST
소셜 네트워에서 영향력 최대화에 대한 연구
허우람, KAIST
COBET : 패턴 기반 동시성 버그 검출기 프레임워크
홍신, KAIST
실제 개발 환경에 적용해 본 Concolic Testing
김윤호, KAIST
Graph Indexing Techniques
한욱신, 경북대학교
인간을 위한 School of Computing
최광무, KAIST
When Does the Tipping Point Occur? : Analysis of Information Cascade in Social Networks
이슬기, KAIST
Predicate Generation for Learning-Based Quantifier-Free Loop Invariant Inference
이원찬, 정영범, 서울대학교
필요한 정보만 가지고 필요한 지점만 분석하기
고윤석, 허기홍, 서울대학교
단계제거 변환을 통한 다단계 프로그램의 정적 분석
최원태, 서울대학교
요약해석에 기반한 경보 동일 군집 구하기
이우석, 서울대학교
Plasma와 Hadoop MapReduce 환경 비교 분석
임정표, 김동원, POSTECH
Boolean BI 논리를 위한 컷-제거 귀추 계산법
박종현, POSTECH
COBET : Effective Concurrency Bug Detector Framework for OS Kernel
홍신, KAIST
기계 학습 기법으로 스팸 메일 걸러내기
곽남주, KAIST
경보 메시지간의 종속 관계를 안전하게 계산하는 방법
김유일, 서울대학교
명시되지 않은 인자를 지원하는 계산 언어의 설계
이원찬, 서울대학교
프로그램 개발 도우미로서의 기호실행에 대한 재조명
이주용, 고려대학교
보다 친숙한 Coq 증명을 위한 방안 연구 II
이계식, 서울대학교
무작위 과정을 분석하는 미분 방정식 방법
김보영, KAIST
단계제거 변환을 통한 다단계 프로그램의 정적분석
최원태, 서울대학교
재귀 모듈을 위한 구문 기반 타입 시스템
임현승, POSTECH

The 4th ROSAEC Center Workshop

기간: 2010-08-25 ~ 08-28
장수: 강원도 설악 대명리조트

Workshop 개회 및 센터 활동 보고
이광근, 서울대학교
웹 응용프로그램 보안 취약성 분석기 구현
안준선, 항공대학교
Malware Detection based on Dependency Graph using Hybrid Genetic Algorithm
김진현, 서울대학교
허위 경보 제로 정적 오류 분석기 사례1 : 무인비행체 제어 소프트웨어에 특화시키기
장수원, 서울대학교
Random Sampling Algorithms with Applications
정교민, KAIST
PLRG @ KAIST
류석영, KAIST
Type Classes as Objects and Implicits
Bruno C.d.S.Oliveira, 서울대학교
A Theory of Non-associative Classical BI
박종현, POSTECH
거짓 경보의 소문을 안전하게 퍼뜨리는 방법
김유일, 서울대학교
Binary Methods Support using Self Type Idiom in Fortress
나현익, KAIST
다단계 언어로 작성된 프로그램에 Concolic Testing 적용하기
김태수, 서울대학교
QNX RTOS기반 PC/104 비행제어컴퓨터 개발 및 기존프로그램 오류검출
김민구, 서울대학교
Intelligent Code Example Search Engine : Automatic Code Example Recommendation
김진한, POSTECH
소프트웨어 안전성 분석
최윤자, 경북대학교
Dynamically Scalable Concolic Testing through the SCORE Framework
김문주, KAIST
데이터타입 제너릭 프로그래밍기법을 Coq 증명에 활용한 라이브러리
이계식, Bruno C.d.S.Oliveira, 서울대학교
Decentralized Algorithm for Ranking Learning
김보영, KAIST
FortressCheck : Automatic Testing for Implicit Parallelism and Generic Properties
강성훈, KAIST
Adding Pattern Matching to Existing Object-Oriented Languages
박창희, KAIST
Concolic Testing을 사용한 Smartphone SW 검증 : Busybox 사례 연구
김윤호, KAIST
깃(Git)으로 즐기는 생활 속 유연한 버전 관리
신재호, 서울대학교
Implementing String Analysis in Soot Framework
최재준, KAIST
Coq Mechanization of Basic Core Fortress for Type Soundness
김지응, KAIST
Compiler triggered C Level Error Check
정지문, 서울대학교

GMeta : 데이터타입 제너릭 프로그래밍 기법을 Coq 증명에 활용한 라이브러리
조성근, 서울대학교

허위경보 제로 정적오류 분석기 사례1 : 무인비행체 제어 소프트웨어에 특화시키기
장수원, 서울대학교

필요한 부위만 가지고 필요한 시점에만 분석하기
고윤석, 오학주, 서울대학교

Energy Minimization under Constraints on Label Counts
임용섭, KAIST

소스코드 보안취약점 자동 탐지를 위한 취약점 패턴 명세 언어
김현하, 한양대학교

Cross-Site Scripting(XSS) 취약점 탐지를 위한 분석
최기환, 한양대학교

Introduction to Cross-Site Request Forgery and Defense Method
김성진, 한양대학교

A Syntactic Type System for Recursive Modules
임현승, POSTECH

Type Inference for Linear Lambda Calculus for Hardware Description
임정표, POSTECH

Program Analysis in the Field
조규진, Fasoo.com

CUDA Programming Assist Tool Development
이승용, POSTECH

PLASSE Research Lab 연구진행 상황
도경구, 한양대학교

보다 친숙한 Coq 증명을 위한 방안 연구
이계식, 서울대학교

Complex Network Modeling and Analysis
정교민, KAIST

태초부터 자동으로 : 반복문 불변식을 자동 유추할 때 사용할 부품식들의 자동 유추
이원찬, 서울대학교

실시간 플랫폼을 위한 정형모델
김진현, 고려대학교

Hidden Markov Model Principles and Applications
곽남주, KAIST

공부의 언어
이광근, 서울대학교

디지털 컴퓨터의 소수점 표현 방법
허기홍, 서울대학교

토픽 모델을 사용한 트위터 데이터 분석
허우람, KAIST

네트워크 상에서 정보가 퍼져나갈 때 생기는 Threshold 현상
이슬기, KAIST

재빠르고 실용적인 C 프로그램 메모리 분석기 개발
정승철, 한양대학교

비슷한 소스 코드 빨리 찾기
이상훈, POSTECH

파일시스템 기반 C 분석기 “Yoyak” 메모리 한계를 뛰어넘자
신재호, 서울대학교

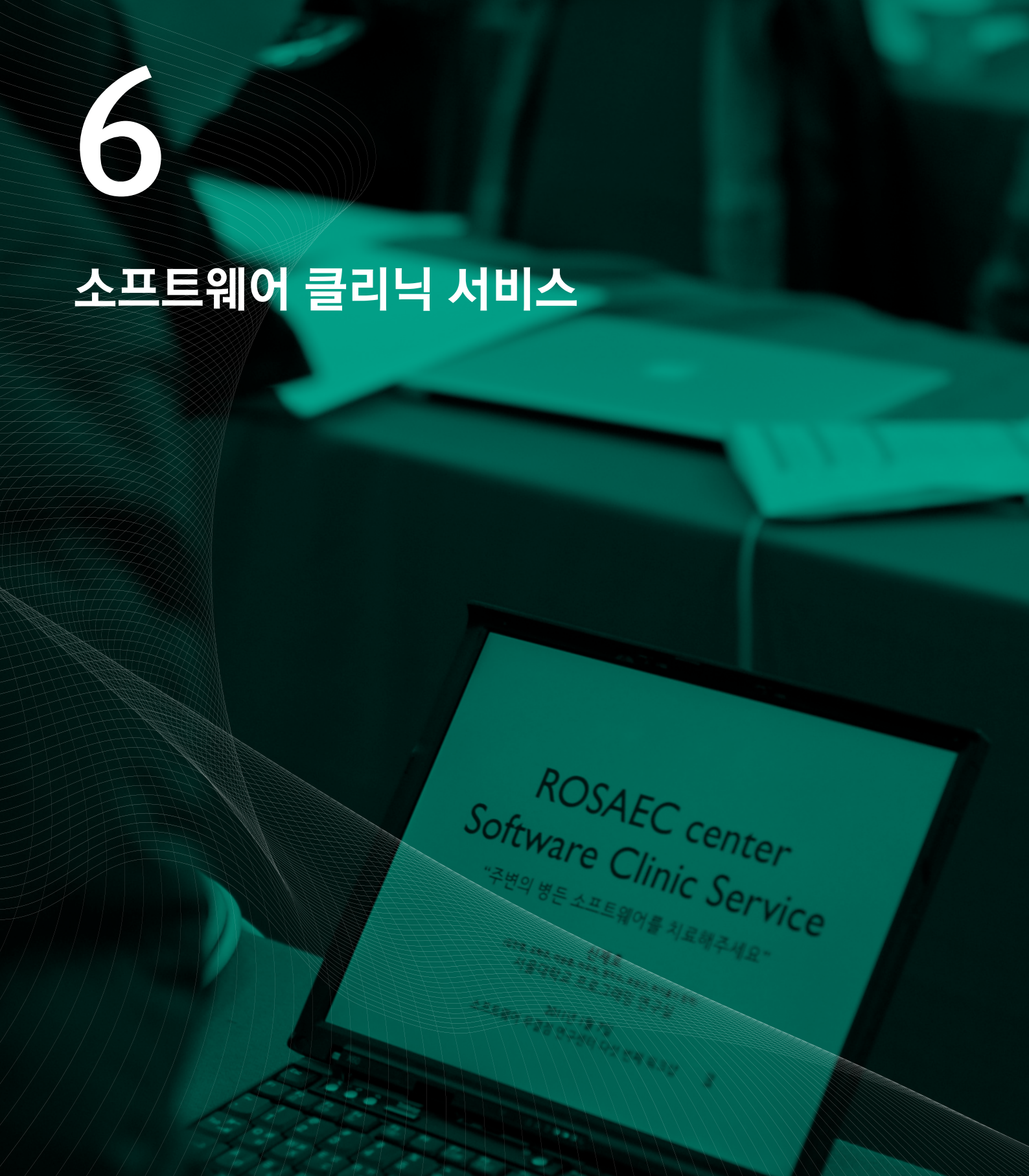
CCR : Code Clone Ransacker
이효섭, 한양대학교

경로민감분석에서 알람군집구하기
이우석, 서울대학교

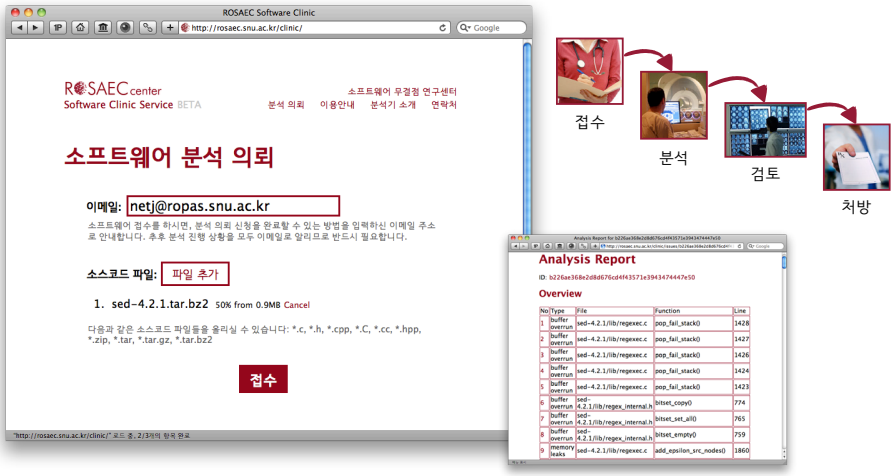
On Supporting Effective Web Extraction
이진수, 경북대학교

6

소프트웨어 클리닉 서비스



소프트웨어 클리닉 서비스



소프트웨어 무결점 연구센터에서는 클리닉 서비스를 통해 누구나 자신의 프로그램의 코드에 어떠한 오류들이 숨어있는지 찾아내어 고칠 수 있도록 정적분석기술을 널리 보급하는 노력을 기울이고 있습니다.

C로 작성되어 있는 소프트웨어의 소스코드를 센터 홈페이지를 통해 접수받아서 분석기들로 검출한 오류들을 전문가가 검토하여 결과보고서를 돌려주는 서비스입니다. 현재는 GNU Make로 빌드 가능한 C 소스코드에서 버퍼오버런 및 메모리누수 오류 검출만을 지원하고 있으나 대상 언어나 환경, 오류 종류를 점차 확대해나갈 계획입니다. 접수된 소스코드는 지금의 분석기술을 더욱 효과적이고 효율적으로 만들어가는 향후 연구에 소중한 자료로 쓰이게 됩니다.

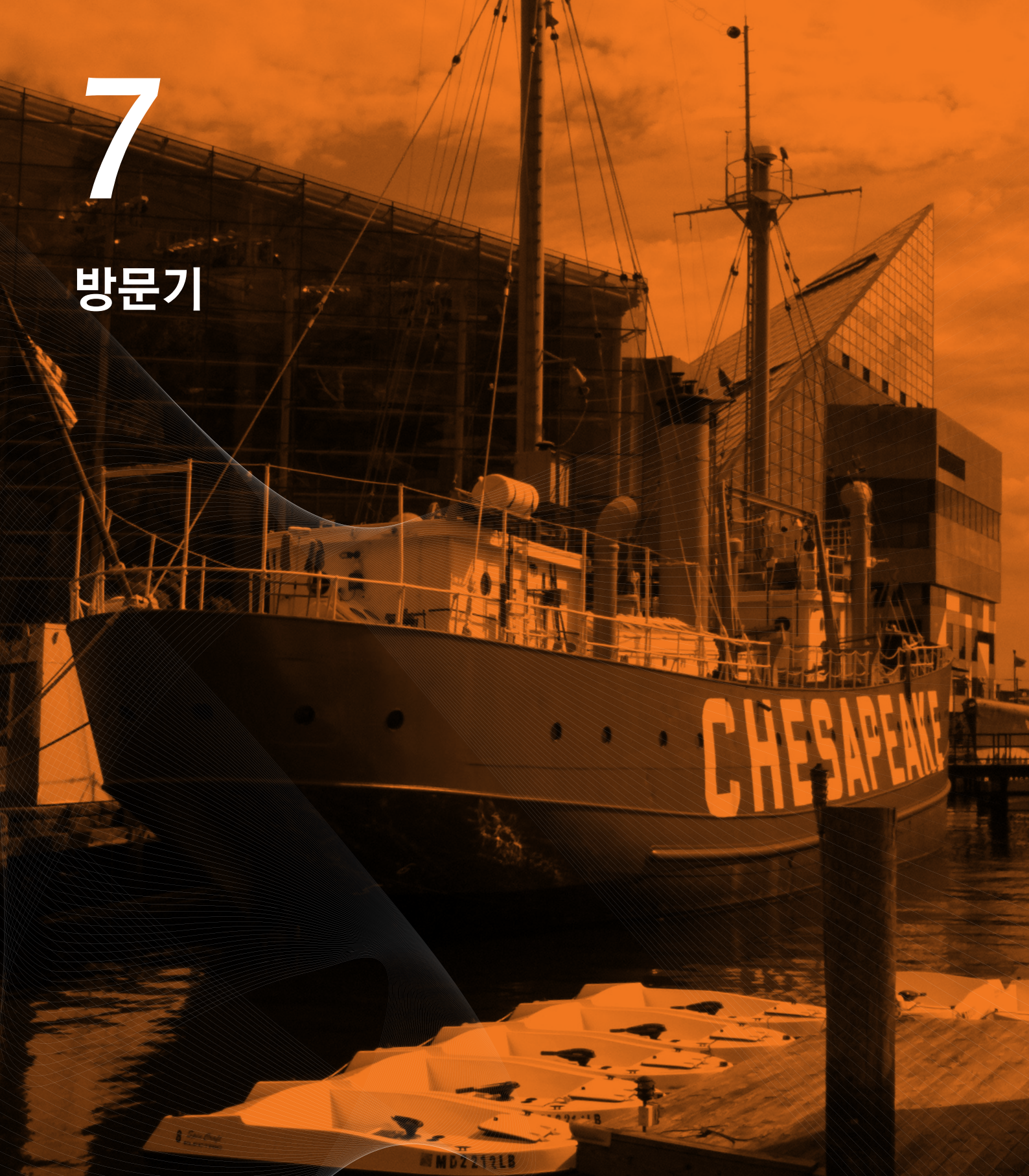
http://rosaec.snu.ac.kr/clinic/

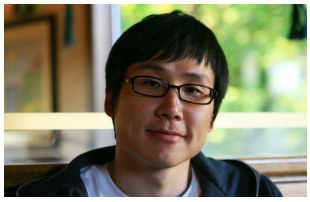
여러분의 소프트웨어에 숨어있는 오류도 진단 받으시고 저희의 연구에도 기여하세요!

소프트웨어 클리닉 팀

7

방문기





OPLSS 2010

글쓴이 임현승
POSTECH 전산학과 박사과정
방문지 Eugene, Oregon, USA
방문기간 2010-06-15 ~ 06-25

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

1. Proof Theory Foundations

by Frank Pfenning

이번 여름학교는 Pfenning 교수님의 강의와 함께 시작했습니다. 박성우 교수님의 지도교수님이라서 그런지 강의 스타일과 내용이 박성우 교수님의 Logic in Computer Science 수업과 매우 흡사했습니다. 덕분에 강의 내내 편하게 복습하는 기분이었습니다. 그러나 Natural deduction, sequent calculus, focusing 등에 대한 배경지식이 없던 친구들은 다소 어려워 했습니다. :)

2. Type Theory Foundations

by Robert Harper

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

3. Essential Coq from Scratch

by Andrew Tolmach

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

4. Software Foundations in Coq

by Benjamin Pierce

Types and Programming Languages(TAPL)의 저자이자 Harper 교수님의 제자로 PL을 공부하는 사람이라면 이름만 대도 알만한 Pierce 교수님께서 Tolmach 교수님 수업에 이어 Coq을 이용하여 programming languages의 meta-theory를 mechanize하는 방법에 대해 강의해주셨습니다. 전체 강의는 Software Foundataions in Coq 후반부 내용 위주로 진행되었습니다. 이번 여름학교에 KAIST 김지응군과 황준형군이 함께 참석했었는데 김지응군이 다소 깨끗한 TAPL 책에 Pierce 교수님의 사인을 받았습니다. 조금 부럽더군요. 제 지저분한 TAPL 책을 가져왔다면 정말 좋았을텐데 하는 아쉬움이 남았습니다. :)

5. Programming Language Methods for Compositional Security

by Anupam Datta

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

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

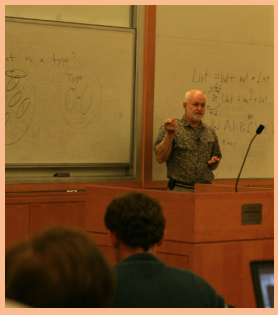
by Robert Constable

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

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

by Xavier Leroy

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



correctness를 증명하는 방법에 대해서도 강의해주셨습니다. 끝으로 realistic compilers의 formal verification에 대해서 연구하는 Compcert Verified Compiler 프로젝트에 대해서도 간략하게 소개해주셨습니다. 여담으로 다른 교수님 강의 시간에 Leroy 교수님 근처에 앉은 적이 있었는데 옆에서 보니 Coq 코딩을 마치 워드로 타이핑하듯이 하시더군요. Automated theorem prover가 따로 필요 없으신 분 같으셨습니다. 덜덜덜...(;)

8. Ynot Programming

by Greg Morrisett

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

9. Dependently Typed Programming

by Conor McBride

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

맺음말

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

ICFP 2010

글쓴이 이원찬
소프트웨어 무결점 연구센터 연구원
방문지 Baltimore, Maryland, USA
방문기간 2010-09-27 ~ 10-02

미국 볼티모어에서 열렸던 ICFP 2010에 다녀왔다. 첫 국외 학회를 혼자 다녀온 만큼 체험이 조금 남달랐다. 여행기를 통해 보고 느끼고 배운 점들을 나누고자 한다. 또한, 혼자 학회를 나가게 될 다른 사람들에게도 도움이 되었으면 좋겠다. 감상 이후에는 견문을 실었다. 전체 논문들에 대한 내 나름의 개괄 이후에는 즐겁게 읽었던 논문들을 실었다. 이번 ICFP 2010에 실린 논문을 읽고자 하는 사람들에게 도움이 되었으면 한다.

여정과 감상

참석학회

크게 ICFP 본 학회와 작은 워크샵, 심포지엄들로 구성되어 있었다. 그 중 Haskell 심포지엄과 CUFP(Commercial Users of Functional Programming) 학회에 참석했다. Haskell 언어가 함수형 언어에 관심을 갖게 된 계기였던 만큼 Haskell 학회 위주로 참석했다.

가장 절실했던 것

학회는 사람을 만나는 장소라는 점을 절실히 느꼈다. 20분 남짓의 발표에서 사실 많은 것을 기대하기 어렵다. 오히려, 세션 사이사이의 휴식시간에 발표자를 중심으로 많은 것들이 일어난다. 자기소개에 이은 질문에서부터 자신의 관심사와의 교류. 그 과정에서 새로운 연구가 시작되기도 한다. (실제로 허충길 박사님은 학회에서 만난 Neel이라는 친구와 새 연구를 시작하기로 했었다.) APLAS때는 절실히 느끼지 못한 부분이었다. APLAS는 한국에서 열렸고 연구실 사람들이 모두 참석한 상황이었기 때문이었을 것이다.

두 가지가 중요하다고 느꼈다. 나를 알리는 것과 남을 아는 것.

학회를 어떻게 준비할 것인가?

가장 좋은 준비는 논문을 내는 것이다. 더 나아가서는 논문이 되는 일이다. 나를 알리는 가장 좋은 방법이다. 이광근 교수님께서 염려하시는 말씀으로 논문을 안내고 가면 겉돌 수 있다고 하셨다. 그 만큼 학회는 발표하는 사람들이 주인공이다. 발표하지 않는 상황이라면 내 연구를 확실히 소개할 수 있어야 할 것이다. 다른 학회에 낸 논문이 있다면 15초안에 소개할 수 있게끔 엘리베이터 스피치(elevator speech)를 준비하면 좋겠다. 실제로, 처음 만난 사람들은 모두가 내게 두 가지를 물어보았다. 나의 연구 분야와 발표한 논문이 있는지. 나는 연구 소개는 잘할 수 있었지만, 아직 발표한 논문이 없어 덜 매력적으로 들렸을 것이다.

연구에 임하는 우리의 자세

나를 알리는 가장 좋은 방법이 논문을 내는 것인 만큼 연구도 그것에 맞게 준비해야겠다는 생각이 들었다. 논문을 내기 위해 연구를 하자는 말이 아니다. 연구에 집중하고 그 과정을 중간마다 사람들에게 알릴 수 있는 형태로 만들어야 한다는 것이다. 좋은 연구를 하는 것도 중요하지만, 사람들에게 내 연구를 알리는 것 또한 절대 소홀해서는 안 된다고 생각한다.

"사람이 일을 하는 것이 아니라 기한이 일을 하는 것이다."라는 말은 내가 좋아하는 말이다. 기한이 될 때까지 일을 미루자는 것이 아니라 구체적인 목표를 가지고 일을 하자는 말이다. 그러니 지금이라도 당장 목표로 하는 학회를 잡고 구체적인 연구 일정을 잡아보는 것은 어떨까? 다음 ICFP는 일본에서 열린다고 하니 한 번 도전해볼 만하다.

그 밖의 것들

당연한 말이지만, 학회에 참석했다면 적극적으로 임해야 한다. 동료와 함께 왔더라도 따로 떨어져 학회 분위기에 녹아들 것을 권한다. 허충길 박사님과 함께 온 MPI 학생들은 비록 한 팀이었지만 함께 있는 것을 거의 못 봤다. 한편, 일본 학생들은 서로 똘똘 뭉쳐 다른 사람들과 대화를 나누는 것을 거의 보지 못했다. 교류가 자연스러운 학회 분위기가 동양권 학생들이 넘어야 할 가장 큰 산이 아닐까 하는 생각이 들었다.

식사 때도 용기를 내서 모르는 사람들 사이로 비집고 들어가 보는 것이 좋다. 학회 가서 놀랐던 것은 서로 모르는 사이에도 쉽게 옆자리에 앉아 대화를 나누곤 한다는 것이다. 나도 용기를 내서 Galois사에서 온 Sally와 대화를 나누었다. Galois는 Haskell을 가장 활발히 쓰는 회사로 알려졌다. 주요 제품은 Cryptol이라는 언어로 미국 정부의 암호화 알고리즘을 안전하게 기술하고 구현하기 위해 고안되었다. Cryptol의 특이한 점은 표현 가능한 비트 수로 매개화된 자료형을 사용하여 계산된 값이 특정 비트 수를 넘지 않는 것을 보장할 수 있다는 점이었다. Cryptol 프로그램에 사용된 배열에 대해 정적 분석을 사용한다는 얘기를 듣고 자세히 물어보았다. 아쉽게도 그분은 기술 담당이 아니라서 이 부분에 대해서는 자세히 이야기 듣지 못했다. 하지만, 대답해 줄 수 있는 사람들의 메일 주소를 적어주었다. 대화를 나누지 않았다면 얻을 수 없었던 정보였을 것이다.

영어 공부가 중요하다는 말은 아무리 강조해도 지나치지 않은 듯하다. 우리 분야는 필연적으로 외국인들과 교류를 할 일이 많기 때문이다. 유창할 필요는 없다고 생각하지만 내 생각을 정확하게 말할 수는 있어야 한다.

만일, 학회에 혼자 참석하게 되었다면 룸메이트를 구해볼 것을 권한다. Haskell, ML, Coq 메일링 리스트같이 CFP(Call For Paper)가 많이 올라오는 곳에서 룸메이트를 찾을 수 있을 것이다. 내 경우에는 Haskell 메일링 리스트에서 룸메이트를 구할 수 있었다. (룸메이트는 David Terei라는 친구로 GHC에 LLVM backend를 추가한 인물로 유명하다.) 한 방을 같이 쓴 덕분에 David과 많이 친해질 수 있었다. 다음 ICFP에도 만나자, 그리고 좋은 연구주제가 있다면 함께 하자고 약속했다.

만일, 함수형 언어를 사용하는 일자리를 얻고 싶다면 ICFP를 꼭 참석하는 것이 좋다. Ocaml을 가장 실용적으로 쓰는

회사 중 하나인 Jane Street가 이번 학회를 대대적으로 지원했다. 티셔츠도 나누어주고 두 번에 나누어 취업 설명회도 하는 등 좋은 인재 확보에 열성이었다. 다른 후원사인 Erlang solutions와 Galois도 함수형 언어를 사용하는 회사들이었다.

마지막으로 될 수 있으면 우리 분야의 좋은 학회에 많이 참석하는 것이 좋다고 생각한다. 학계가 돌아가는 분위기도 알 수 있고 사람들과 친해질 수 있는 좋은 계기가 될 수 있다. 잘하는 사람들을 보면서 자극도 받을 수 있다.

ICFP 2010에 발표된 논문들

이번 ICFP 학회에는 99편의 논문 중 33편의 논문이 채택되었다. 33%의 채택률(acceptance ratio)로 학회의 인지도에 비해 도전해볼 만한 수치인 것 같다. 33편의 논문을 내 나름대로 분류해본 결과는 다음과 같다.

분석	2편	프로그램 검증	4편
타입 체계 구현	3편	함수 프로그래밍 및 라이브러리	4편
타입 이론	5편	부분 실행	2편
타입을 사용한 보안	2편	Functional Pearl	2편
Bidirectionalization	3편	Experience Report	3편
병렬 프로그래밍	1편	기타	2편

우선 분석 논문이 많지 않다는 점이 놀라웠다. (2편 중에 한 편은 우리 연구실 사람들이라면 익히 알고 있을 Matt Might의 논문이다.) 아무래도 정적 분석은 주로 안전하지 않은 언어를 대상으로 하기 때문인 것 같다. 또, 강력한 타입 체계에 대한 자부심도 일조한 것 같다. 함수형 언어의 정적 분석에 대해 좋은 논문을 쓰면 학계에 대한 큰 공헌이 될 것 같다. (개인적으로 Haskell의 메모리 사용량에 대한 정적 분석은 시도해볼 만한 주제라고 생각한다.)

두 편의 Functional pearl 논문은 그 자체로도 쉽고 잘 쓴 글이었지만 단순히 pearl 수준에서 머무는 것이 아니라 굉장히 실용적인 것들을 다루고 있었다. 두 편의 내용은 뒤에서 좀 더 설명하도록 하겠다.

타입 체계 구현을 다룬 세 편의 논문이 인상적이었다. 하나는 타입이 없는 언어인 Racket (구 PLT Scheme)에서 타입을 유추하는 시스템이다. 다른 하나는 Haskell의 타입 클래스 사용 시 문제가 되는 겹치는 타입 클래스 구현(overlapping type class instance)와 관련된 것이다. 이 둘은 뒤에서 좀 더 설명하도록 하겠다.

나머지 하나는 semantic subtyping에 관한 것이었다. 자료형의 생김새만으로는 하위 타입인지를 검사할 수 없는 경우 SMT solver를 사용하는 것이 핵심 아이디어다. 예를 들어, 기본 타입이 `int`, `float`, `string`밖에 없는 경우 `x : !{f:int} && !{f:float}`는 `x : {f:string}`와 동등하지만, 생김새만으로는 이를 알 수 없다. 이럴 때 `!{f:int} && !{f:float}`가 `{f:string}`을 의미(imply)하는 것으로 하위 타입 관계를 검사하겠다는 것이다. 이 semantic subtyping은 구현되어 MS의 dminor라고 하는 자료처리 언어에 사용되고 있다고 한다.

Bidirectionalization과 관련된 논문이 세 편으로 꽤 활발하다는 인상을 받았다. Bidirectionalization은 내가 이해하기로 주어진 함수의 역함수를 구하는 방법을 말한다. 세 편의 논문 중 Benjamin Pierce의 그룹에서 한 Boomerang 프로젝트가 인상적이었다. Boomerang은 문서 처리를 위한 DSL이다. 주어진 문서를 뷰를 통해 갱신할 때에도 원래 문서의 형태가 깨지지 않게 하는 것이 목적이다. 뷰에서 재배열이 일어나는 경우가 문제가 되는데, 이를 원래 문서와 뷰의 사상 관계를 관리하는 렌즈(lens)로 해결한다. 실용적인 목적을 잘 기술하고 있어 같은 세션의 다른 논문들보다 인상적이었다.

많은 수의 논문이 Coq, Isabelle과 같은 theorem prover로 자신의 정리들을 증명했음을 보였다. 갈수록 더욱 보편화 될 것이기 때문에 공부할 필요가 있다는 생각이 들었다.

타입 이론에 대한 발표는 배경지식이 일천하여 제대로 이해한 발표가 없었다. 발표장에 앉아 있는 내내 부끄럽기 짝이 없었다. 모든 분야를 다 잘 아는 것은 물론 힘든 일이지만 적어도 알아들을 수는 있어야겠다고 생각했다.

두 편의 functional pearl 논문들

"Functional Pearl: Every Bit Counts"는 가장 재밌게 읽었던 논문이다. 타입이 있는 람다 프로그램을 부호화하고 복호화하는 라이브러리에 관한 것이다. 게임을 사용한다는 점이 재밌는데, 게임은 프로그램에 대한 예/아니오로 답할 수 있는 질문으로 구성된다. 부호화는 질문을 반복함으로써 예, 아니오의 리스트를 얻어내는 과정이다. 반대로 복호화는 질문의 답으로부터 프로그램을 되살려내는 과정이 된다. 프로그램 serialization, 코드 손상 검사를 위한 체크섬 등에 사용할 수 있는 매우 실용적인 아이디어다. 논문의 아름다운 점은 예/아니오 질문 게임을 만들기 위한 일반적인 라이브러리를 구축했다는 점이다. 라이브러리는 간단한 게임들로 복잡한 게임을 만들 수 있는 조합기(combinator)들로 구성되어있다.

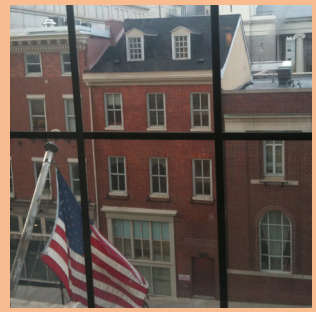
한 가지 걱정되었던 점은 예/아니오로 대답할 수 있는 질문만 지원한다는 점이었다. 예/아니오 질문을 연속해서 할 때 게임을 표현하는 트리는 이진 트리의 형태가 된다. 따라서, 질문을 많이 반복한다면 메모리 사용량이나 처리 속도 측면에서 문제가 있을 것 같았다. 저자인 Dimitrios에게 이를 물었으나 메모리 사용량은 염려되는 부분이지만 처리 속도는 크게 신경 쓰지 않는 것 같았다. 그의 말로는 질문 하나하나의 처리가 매우 간단하여서 최적화될 것이라고 했다. 실제 사용되는 언어들에 적용해볼 예정이라고 하는데 그 결과가 기대된다.

두 번째 functional pearl인 "A Play on Regular Expressions"는 Haskell로 구현한 정규 표현식 라이브러리에 관한 것이다. 기술적인 내용에는 특별히 놀라운 점은 없었다. 하지만, Haskell을 사용해서 고성능 정규 표현식 라이브러리를 만들 수 있었다는 점이 놀라웠다. (Google에서 최근에 만든 C로 된 정규 표현식 라이브러리보다 빠르게 동작했다고 한다.) 매치되는 문자열을 모으는 기능과 다른 문자열로의 치환 기능이 추가되면 정말 쓸만해 질 것 같다. 또 다른 놀라운 점은 논문이 희곡 형태로 구성되어 있다는 점이다. 논문을 Haskell 전문가, 추상화(abstraction) 전문가, 오토마타 전문가 셋이서 라이브러리를 완성해가는 과정에 나눈 대화로 그리고 있다. 형식에 구애받지 않고 사람들이 쉽게 접할 수 있게 만든 점이 훌륭했다. 그 덕분에 사람들이 매우 좋아했던 논문이었다. 개인적으로도 가장 pearl다운 pearl이 아니었나 하고 생각한다.

"Instance Chains: Type Class Programming without Overlapping Instances"

타입 클래스를 쓸 때 문제가 되는 겹치는 구현(overlapping instances) 문제를 해결하기 위한 한 가지 시도를 다루고 있다.

겹치는 구현 문제는 다음과 같다. 타입 클래스는 타입 혹은 타입의 관계에 대한 집합으로 볼 수 있다. 타입들 간의 관



계를 새로이 집합에 가담시키는 방법은 구현을 추가하는 것이다. 문제는 서로 다른 구현이 하나의 타입에 동시에 해당할 수 있다는 점이다. 예를 들어, C라고 하는 타입 클래스가 있고 `([a], b)`와 `(a, [b])`에 해당하는 구현이 있다고 하자. (`[a]`는 a의 리스트를 말한다.) 만일, `([a], [b])`에 대한 구현을 찾을 때 `([a], b)`, `(a, [b])`가 동시에 해당된다. Haskell의 타입 클래스 구현이 나중에 추가될 수 있다는 사실은 문제를 더욱 악화시킨다. 왜냐하면, 처음에는 `([a], b)`에 대한 구현만 있다가도 나중에 다른 사용자가 `(a, [b])`에 해당하는 구현을 추가할 수 있기 때문이다.

겹치는 구현이 존재할 때에 GHC는 주어진 타입에 더 가까운 구현을 선택한다. 가까운 구현이라 함은 그 구현의 타입에서 주어진 타입에 이르기까지 필요한 타입 변수의 치환 횟수를 말한다. 예를 들어, `[Int]`는 `[a]`보다 `[Int]`와 더욱 가깝다. 하지만, 앞에서 설명한 경우는 어느 한 쪽도 다른 쪽보다 더 가깝다고 말할 수 없다.

논문은 해결책으로 적용할 구현 사이에 고정된 순서를 두는 방법을 제안한다. 즉, 주어진 타입이 `([a], b)`로 표현될 수 있는지 먼저 살펴보고 그다음 `(a, [b])`를 적용하는 것이다. 이를 위해 겹칠 수 있는 구현은 한꺼번에 선언할 수 있게만 제한한다. 이 덕분에 다른 곳에서 추가된 구현은 다른 어떤 구현과도 겹치지 않는 다는 것을 보장할 수 있다. 이 타입 클래스 체계가 안전함을 증명하기 위해 타입 클래스의 의미를 엄밀히 정의한 것이 논문의 핵심이다.

“Logical Types for Untyped Languages”

타입이 없는 Racket언어에서 타입을 유추하기 위한 타입 체계의 구현을 다룬 논문이다. 타입을 정적으로 알아내기 위해서 조건문에서 사용하는 타입 검사 구문에서 정보를 얻는다. 예를 들어, `(cond [(number? x) (body)] ...)`와 같은 프로그램에서 `body`에서의 x의 타입은 숫자라는 것을 확실히 알 수 있다. 이 같은 조건문마다 만족 될 경우와 아닌 경우에서 타입을 모아 각 문장의 타입을 결정한다. (이와 같은 타입 체계를 occurrence typing이라고 부른다고 한다.) 이 아이디어를 가져다가 많이 쓰이는 Python이나 Ruby에 적용해보면 재밌을 것 같다는 생각이 들었다.

저자 중 한 명인 Matthias Felleisen은 프로그래밍을 가르치는 일에도 열심인 분이다. 이 논문에서도 그의 노력을 엿볼 수 있었다. 타입이 없는 언어를 잘 만들어서 아이들에게도 가르칠 수 있게 한다. 다른 한편으로는 타입 체계를 지원함으로써 안전한 구현도 가능하게 만든다. 두 가지를 모두 잘할 수 있는 언어로 Racket을 발전시키고 있구나 하는 생각이 들었다.

Haskell 심포지엄과 CUFP

“Hoopl: A Modular, Reusable Library for Dataflow Analysis and Transformation”

사실 이 논문은 발표되기 훨씬 전에 읽어본 논문이다. (Hoopl을 사용하여 “구간 도메인 분석기, 5분 안에 만들기”라는 튜토리얼 자료를 만들기 위해서였다.) 데이터 흐름 분석을 위한 일반적인 프레임워크로 그래프 형태의 중간 언어를 대상으로 하고 있다. 지난 POPL과 ICFP 모두에서 떨어지고 우여곡절 끝에 이번 Haskell 심포지엄에 채택된 비운의 논문이다.

Hoopl은 2002년 POPL논문인 “Composing dataflow anlayses and transformations”을 구현하고 있다. 핵심은 데이터 흐름 분석의 결과로 코드를 최적화하고, 최적화된 코드에 분석을 다시 수행하는 것을 반복하는 것이다. 사용자로 하여금 분석과 코드 재작성을 잘 기술할 수 있게끔 필요한 부분에만 빈칸을 만들어두었다.

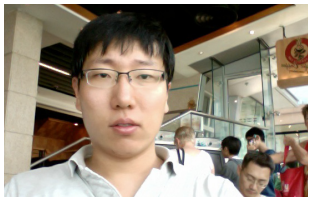
Haskell 논문답게 타입 체계의 위력을 곳곳에서 발휘하고 있다. 가장 눈에 띄는 것은 타입을 통해 잘못된 형태의 코드를 만들지 못하게 했다는 점이다. GADT를 사용해서 각 블록이 부모, 자식 블록에 대해 열려 있는지를 타입 정보로 부호화했다. 예를 들면, 함수를 시작하는 블록은 부모에 대해서는 닫혀 있고 자식에 대해서는 열려 있는 타 입을 가진다. 반대로 함수를 끝마치는 블록은 자식에 대해 닫혀 있고 부모에 대해서는 열려있다. 그다음, 서로 열려 있는 블록들 간의 연결만 유효하도록 연결 함수의 타입에 제한을 두었다. 즉, 자식에 대해 열려 있는 블록과 부모에 대해 열려있는 블록만 연결할 수 있다. 이같이 형태에 대한 제약을 타입에 부호화하여 정적으로 프로그램 그래프의 생김새를 검사할 수 있게 된다.

Simon Peyton Jones가 직접 발표해서 더욱 기억에 남는다. Haskell을 만든 사람이니 심포지엄에 오겠거니 했지만 직접 발표할 줄은 꿈에도 몰랐다. Hoopl을 구성하는 각 모듈의 타입만 가지고 왜 타입이 그러해야만 하는가를 일목요연하게 설명해나갔다. 크리스티나가 왜 Simon Peyton Jones의 발표가 인상 깊었다고 했는지 알 것 같았다.

High performance Haskell

내가 이번 학회에서 가장 기대했던 세션이었다. 산학 과제에 Haskell을 적용하면서 겪었던 문제들의 해답을 찾고 싶었다. 하지만, 대부분 내가 알고 있는 사실들을 재확인했던 자리라 조금은 실망스러웠다. 고성능 Haskell 코드를 작성하기 위한 지침을 정리하면 다음과 같다:

1. 될 수 있으면 적극적 계산(strict evaluation)을 활용하라. Haskell은 지연 계산(lazy evaluation)을 하는 언어이다. 모든 계산은 필요하기 전까지 포장된 thunk 형태로 보관된다. 따라서, 당장 결과가 필요한 경우 이는 순수 비용이 된다. (아이러니하게도 대부분 결과가 당장 필요하다.) 따라서, 다음의 두 가지 방법을 통해 적극적 계산을 유도하는 것이 필요하다. 첫 번째는 추상 자료형(abstract data type)에 적극적으로 계산하는 필드를 두는 것이다. 적극적으로 계산되는 필드는 자료형 앞에 느낌표가 붙어 있다. 두 번째는 함수 인자 앞에 느낌표를 붙이는 것이다. (이를 bang pattern이라고 부르는데 사용을 위해서는 GHC의 확장된 기능을 활성화 해야 한다.) 적극적으로 계산되는 필드 혹은 인자에 대해서는 값이 묶이기 전에 계산이 완료되게 된다.
2. Unpack pragma를 사용하라. Haskell의 모든 자료형은 타입 정보와 함께 포장되어 있다. 단일 타입의 적극적으로 계산되는 필드는 타입 정보가 필요 없으므로 떼어버릴 수 있다. 자료형을 떼고 난 다음에는 부모 자료형에 곧바로 내장시킬 수 있게 된다. 예를 들어, Foo Int로 선언한 경우 Foo는 타입정보 저장을 위해 4바이트, Int에 대한 포인터 4바이트를 차지하고, 추가로 Int 자료형 또한 8바이트를 차지한다. (4바이트는 타입정보, 4바이는 실제 정수 값) 한편, Foo !Int의 경우 Int 자료형을 위한 8바이트를 절약할 수 있게 된다.
3. 함수의 각 인자의 계산 여부는 반환되는 값 (return value) 계산에 쓰이는지로 알 수 있다. 반환 값에서부터 역으로 추적하면서 인자가 그 계산에 이바지하는지 확인하면 된다.
4. 끝 재귀호출(tail-recursive)을 사용하라. 다른 함수형 언어와 마찬가지로 Haskell에서도 끝 재귀호출에 대해 최적화를 수행하기 때문이다.
5. Core 언어로 변환된 결과를 참조하라. GHC의 -ddump-simpl 옵션을 사용하면 core로의 변환 결과가 출력된다. Core에서는 case문에서만 계산이 일어나기 때문에 내가 만든 코드의 동작을 더 쉽게 예상할 수 있다. 실제로 많은 Haskell 해커들이 한 화면에는 Haskell 소스를, 다른 화면에는 core를 띄워놓고 작업한다고 한다.
6. 프로파일링을 사용하라. GHC는 다양한 프로파일링 옵션을 지원한다. 시간 및 메모리를 많이 사용하는 함수를 순위별로 출력해준다. 또, 시간에 따른 메모리 변화량을 그래프로 그려주기도 한다. 메모리 사용량은 함수별, 자료형 별로 출력 가능하다. 프로파일링 결과를 통해 문제를 일으키는 함수를 찾는데 도움을 얻을 수 있다. 실제로, Haskell로 프로그램을 만들 때 메모리 사용량이 항상 문제가 되는데, 이때 프로파일링이 도움된다.



APLAS 2010 & 칭화대

글쓴이 정영범
서울대학교 컴퓨터공학부 박사과정
방문지 중국 상하이, 베이징 칭화대학교
방문기간 2010-11-27 ~ 12-04

APLAS 2010

November 27 – December 1, Shanghai, China

APLAS 학회는 벌써 이번이 세 번째 참가입니다. 그래서 그런지 사람들도 낯이 익고, 포근한 분위기를 느꼈습니다. 지금까지 그 어떤 학회보다도 즐겁게 보낸 학회가 아닌가 생각합니다. 그 이유는 다음과 같습니다.

첫째, 좋은 사람들을 만날 수 있었습니다. 같이 논문을 쓴 왕보요 (Bow-yaw Wang) 교수님과 Cristina David를 다시 볼 수 있었고, 스페인에서 만났었던 성격 좋은 Liqian Chen도 만날 수 있었습니다. 작년 APLAS에 발표하러 왔었던, 프린스턴 대학의 Andrew Appel 교수님 제자인 Robert Dockins는 역시 Appel 교수님 제자였다가 지금은 싱가포르 국립대학에 교수로 있는 Aquinas Hobor와 함께 올해에는 A Logical Mix of Approximation and Separation이라는 제목으로 Tutorial을 맡았습니다. Robert Dockins는 작년에 포스터로 발표한 우리의 연구를 잘 기억하고 있었습니다. 그리고, 2005년 런던 SAS 학회에서 발표를 해서 아는 Axel Simon도 논문을 발표했습니다. 재밌는 것은 제가 따로 알고 있던 Liqian Chen과 공동 저자이고, 저와 공동 저자인 Cristina David와도 상당한 친분이 있다는 것입니다. 역시 비슷한 일을 하는 사람들에게는 이 지구가, 이 바닥이 그다지 크지 않은 것 같습니다. 또, 독일에서 resource analysis를 연구하고 있는 Jan Hoffmann도 만났습니다. 한국의 분단 문제에 대해서 독일의 경우와 비교하며 재밌는 이야기를 나누었습니다. 생각보다 한국에 많은 관심을 가지고 있었습니다.

둘째, 맛있는 음식을 다양하게 즐길 수 있었습니다. 괜히 상해진미(?)라는 말이 있는 것이 아니었습니다. 학회에서 아침 점심 저녁을 모두 챙겨 주었는데 끼니마다 색다른 요리를 맛볼 수 있었습니다. 너무 행복해서 관광을 못하는 아쉬움은 들지도 않더군요. 요리들이 한국식 중국집과 다른 것은 말할 것도 없고, 한국에 있는 진짜 중국인들이 운영하는 중국 음식점과도 달랐습니다. 재료가 다르기 때문이기도 하거니와 워낙 다양한 종류의 요리들이 넘쳐나기 때문이기도 하겠지요.

셋째, 만족스러운 발표를 할 수 있었습니다. 학회 마지막날 오전 세션의 마지막 발표였습니다. 저는 항상 발표를 하기 전에는 항상 긴장을 하는데 막상 시작을 하면 마음이 편안해집니다. 무대 공포증의 반대말이 있다면 제가 거기에 해당하겠습니다. 대학교 시절에 연극부를 동기들과 만들어서 활동을 했었는데 항상 연습이나 총연습 때는 망쳐서 연출과 다른

스태프들을 걱정시켰는데 관객 앞에만 서면 항상 제 실력보다 더 좋은 연기를 하곤 했습니다. 항상 그러니까 나중에는 총 연습 때 무대를 부수는 정도의 실수를 해야 걱정을 하더군요. 제 발표는 다른 것은 몰라도 가장 많은 사람이 집중한 발표가 아니었나 생각합니다. 발표가 끝나고 세션 좌장이었던 Aquinas Hobor가 "It was a really nice talk!"라고 직접 와서 이야기 해주었습니다. 이광근 교수님께서도 "Your talk was superb!"이라고 과찬을 해주시고, 다른 교수님들께서도 칭찬해 주셨습니다. 정말 뿌듯했습니다.

넷째, 기쁜 소식을 들을 수 있었습니다. 항상 마음속으로 동경하고 있는 양홍석 교수님이 옥스퍼드 (Oxford) 대학으로부터 교수직 제안을 받았다는 소식을 이광근 교수님께 전해 들었습니다. 정말 가슴이 벅차오르는 느낌을 받았습니다. 아주 기뻐서 눈물이 날 것 같더군요. 제가 겉보기에는 덩치가 커서 감수성이 없을 것 같지만 의외로 감수성이 풍부하답니다. 내년 APLAS의 program chair를 양홍석 교수님이 맡는다고 하니 우리나라에서 논문을 많이 제출했으면 좋겠습니다. 내년 APLAS는 2004년과 같이 대만에서 한다고 합니다. 그때도 음식이 정말 훌륭했으니 내년에도 기대하셔도 될 것 같네요. 타이베이는 아니고 대만 남쪽 도시에서 한다고 합니다. 타이베이의 엄청난 매연을 피해서 좋은 경치를 볼 수 있다고 하니 그 또한 좋은 일이 아닐 수 없습니다. 타이베이의 매연은 30년을 서울에서 살아온 저에게도 엄청나게 벅찼습니다. 오죽하면 서울에 돌아와서 맨 처음 한 일이 심호흡이었을까요.

자, 이제 인상 깊게 들었던 발표들을 정리해보겠습니다.

A Logical Mix of Approximation and Separation

by Aquinas Hobor, Robert Dockins, and Andrew W. Appel

새로운 프로그램 논리에 대한 튜토리얼입니다. 이 논리는 Concurrent C minor project에 사용된 것으로 요약과 분리 (approximation and separation)을 지원하는 논리입니다. Concurrent 프로그램을 모형화 (modeling)하는데 좋은 논리를 제공하고 있습니다. 요약과 분리를 모두 지원하도록 둘 사이에 적절한 소통을 설정함으로써 많은 프로그래밍 언어의 의미를 표현할 수 있습니다. Coq으로도 구현되어 공개가 되어 있으니 필요하신 분은 참고하세요. (http://msl.cs.princeton.edu)

Automatically Inferring Quantified Loop Invariants by Algorithmic Learning from Simple Templates

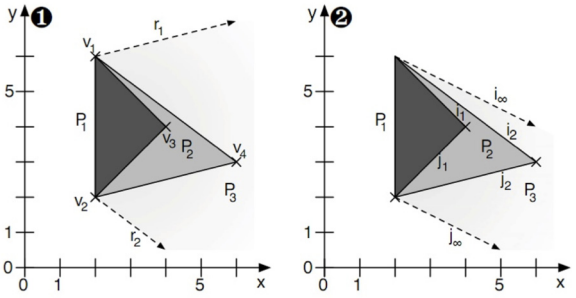
by Soonho Kong, Yungbum Jung, Cristina David, Bow-yaw Wang, and Kwangkeun Yi

제가 발표했던 논문입니다. 올해 VMCAI에 발표되었던 논문을 확장한 논문입니다. 이전 연구에서는 정량 루프 불변식 (quantified invariant)을 찾을 수 없었는데 이번에 간단한 거푸집 (template)을 이용하여 정량자와 그 형태를 힌트로 받아 그 거푸집에 맞는 루프 불변식을 찾게 됩니다. 기존에 있는 여러 기술을 조합하여 어떻게 루프 불변식을 찾을 수 있는지 보여주는 연구입니다.

Simple and Precise Widening for H-Polyhedra

by Axel Simon and Liqian Chen

H-polyhedra domain에서 사용하는 축지법 (widening)을 개선하는 연구입니다. 더 정확하고 그러면서도 중복된 부등식을 제거함으로써 더 빠른 축지법을 고안했습니다. 기본 아이디어는 간단한데 그림으로 보시면 이해가 빠릅니다. 그림 1이 기존의 다른 논문에서 소개된 축지법입니다. P_1에 P_2로 축지법을 적용하면 선분 v_2v_4로 평행한 반직선 r_1을 만들고 선분 v_1v_4로 평행한 반직선 r_2를 만들어서 이 두 반직선과 원래 선분 v_1v_2에 포함되는 다각형 영역 P_3을 축지법을 적용한 최종 상태로 계산합니다. 그림 2는 이 논문에서 제시하는 축지법입니다. 평행 직선을 찾는 것이 아니라 선분 i_1과 i_2의 차이만큼 회전을 시켜 i_∞를 만들고 마찬가지로 선분 j_1과 j_2로부터 j_∞를 만들어 냅니다. 이로써 더욱 정확한 상태 P_3를 얻게 됩니다.



아이디어는 간단한데 (제목에도 simple이) 여러 가지 heuristics를 쓰기 때문에 상세한 내용은 복잡합니다. Cousot 교수님 영향권 하에 있는 연구자들이 수 도메인 (numerical domain)의 분석에 대해서 논문을 쓰는 걸 보면 작은 걸음걸음으로 보이지만 궁극에는 어디까지 갈지 궁금하기도 합니다.

An Interactive Tool for Analyzing Embedded SQL Queries

by Aivar Annamaa, Andrey Breslav, Jevgeni Kabanov, and Varmo Vene

자바 프로그램에 문자열 (string)로 표현되는 SQL 질의문이 문법에 맞는지 찾아주는 도구를 구현한 tool paper입니다. 흥미로운 점은 한양대의 도경구 교수님, 김현하 학생, David Schmidt 교수님께서 개발하여 SAS 학회에 실린 abstract parsing을 구현했다는 점입니다. 그리고 무한히 생성될 수 있는 문자열이 있을 때에도 분석이 끝날 수 있도록 스택의 깊이를 제한하는 방법을 썼는데 이것은 공순호, 최원태가 이광근 교수님과 함께 작년 GPCE에 실은 논문에 나오는 방법입니다. 실험결과를 보면 생각보다 많은 곳에서 문자열이 안전하게 생성된다고 증명을 해냅니다.

Data Structure Fusion

by Peter Hawkins, Alex Aiken, Kathleen Fisher, Martin Rinard, and Mooly Sagiv

Martin Rinard 교수님께서 저자에 있어서 좋기도 했지만 발표도 잘해서 가장 재밌게 들었던 발표입니다. 실제 Linux 같은 프로그램들은 여러 가지 자료구조들이 서로 공유해서 복잡한 구조를 만드는 경우가 있습니다. 이 논문에는 그런 자료 구조를 어떻게 상위레벨 (high-level)에서 표현하고, 그런 자료 구조가 만족해야 하는 조건을 표현할지를 기술하고 있습니다. 또, 그 상위레벨에서의 조건을 어떻게 구현의 실제인 하위레벨까지 안전하게 전달할지도 설명하고 있습니다.

마지막 날 마지막 세션은 재끼고 이계식 박사님, 왕 교수님과 같이 예원 (위위안)에 다녀왔습니다. 예원은 명 왕조시대에 판원단이라는 관료가 부모님을 위해 18년 동안 만든 개인정원이라고 합니다. 근데 규모가... 개인 소유 정원이라면서 양심도 없이 무지하게 큽니다. 정원의 둘레에는 용의 기와가 있는데, 일반인이 용 문양을 사용하는데 화가 난 황제가 잘못을 추궁하자 용은 발톱이 5개인데 이 짐승은 자신이 상상하는 동물로 발톱이 3개라고 말해 위기를 모면했다고 합니다. 딱 봐도 용인데 황제가... 말이 개인정원이지 안에 좁은 길로 들어가면 마을 사람들 다 모여서 식사를 할 수 있는 공간까지 있더군요. 대륙의 크기가 무엇인지 보고 왔습니다.

Tsinghua University

December 2 – 4, Beijing, China

왕 교수님이 이광근 교수님을 칭화대학교에 초빙해서 저도 함께 방문하게 되었습니다. 그곳에서 Jean-Pierre Jouannaud 교수님을 만나뵐 수 있었습니다. 장피에르 교수님은 term rewrite system으로 굉장히 유명하신 분이더군요. 만나뵐 때는 몰랐습니다만... 무척 유쾌하시고 연세가 예순이 넘으셨는데도 정정하셨습니다. Formal method나 type system 관련해서 이광근 교수님과 많은 이야기를 나누셨는데 제 미천한 지식으로는 따라가지 못해서 정신을 못 차렸었습니다. 그리고 제자이자 지금은 포닥으로 있는 Frédéric Blanqui 박사를 만나 그의 연구에 대해서 들었습니다. term rewriting rule의 종료 (termination)를 판단하는 도구들이 경쟁하는 termination competition 학회가 있다고 합니다. 블랑키 박사가 한 일은 도구들이 찾은 종료 조건 혹은 종료되지 않는 조건을 기술한 명세서 (specification)이 실제로 맞는지를 자동으로 coq을 이용하여 증명하는 일입니다. 분명히 term rewriting rule의 종료 조건을 증명하는 일이 일반적인 프로그램의 종료 조건을 증명하는 일로 확장이 가능할 것 같은데요. Andrey Rybalchenko의 C 프로그램 종료 결정 도구인 terminator랑 비교하면 어떨지 궁금했습니다. term rewriting rule에서 문제를 풀면 이용하기 쉬운 기술들이 물론 있겠지요. SMT solver가 푸는 문제와 SAT solver가 푸는 문제의 차이처럼요. 또, 칭화대에 포닥 한 명과 박사과정 한 명을 만났습니다. Higher-order Type system에 대해 발표를 해주었는데 역시나 저에게는 벅찼습니다.

이광근 교수님께서는 2개의 POPL 논문의 내용을 같이 발표했습니다. 총 20명 정도가 참석했었습니다. 발표를 저는 무척 재밌게 들었는데 역시나 POPL 논문을 두 개 더군다나 다단계 프로그램 (multi-staged program)에 대한 내용을 사전 지식 없이 따라가는 것이 힘들었나 봅니다. 여러 학생이 정신을 멀리 보내고 몸만 앉아있는 모습을 보았습니다. 장피에르 교수님과 왕 교수님은 재밌게 들었다고 하셨습니다.

첫날은 북경오리를 맛볼 수 있었습니다. 전취덕 (취엔쥐더)이라는 레스토랑에 갔었는데 메뉴판이 스케치북만큼 컸습니다. 왕교수님이 아는 진짜 레알 북경오리를 파는 레스토랑 (리췬 카오야디엔)은 시간상 가지 못했습니다. 전취덕도 충분히 인상적이었습니다. 북경오리도 물론 훌륭했지만 (오리의 피부는 고소하면서도 식감이 독특하더군요.) 제가 시켰던 가지요리가 예술이었습니다. 한국의 중국 음식점에 가면 항상 시키는 지삼선 (띠싼쓰?)이라는 가지 요리가 있습니다. 감자, 피망, 가지 이 세 가지를 짭짤한 검은색 소스에 볶아서 나오는 요리인데 맛있습니다. 그 지삼선이 없어서 대신시킨 요리인데 가지 위에 고기 완자 같은 것을 얹어서 구운 요리입니다. 눈물 나게 맛있습니다. 실제로 눈물을 흘렸습니다. 물론 뜨거움도 눈물에 기여를 하긴 했지 만도요. 가지의 부드러움과 촉촉함에 고기 완자의 고소함과 까끌까끌한 식감까지 환상적이었습니다. 그리고 다른 음식들도 모두 훌륭했습니다.

둘째 날은 장피에르 교수님 댁에 가서 와인을 마시며 집안을 구경했습니다. 집이 넓고 고급스러웠습니다. 사모님은 40대로 보이는데 재혼을 하셨다고 들었습니다. 왕 교수님과 같이 대만분이라고 합니다. 집안에 사모님이 직접 그린 그림들도 보고, 와인 냉장고도 구경했습니다. 와인을 마시고 훠궈집을 갔습니다. 갈 때도 와인잔과 와인을 들고 가셨습니다. 역시 프랑스 분이시라 와인 전문가의 면모를 보여주시더군요. 와인잔은 목이 없는 형태였는데 무척 가벼웠습니다. 와인의 종류에 따라 와인잔도 그것에 맞게 쓴다고 합니다. 혀의 어느 부분에 닿는지를 와인잔의 모양과 크기가 결정하기 때문이라고 합니다. 이광근 교수님께서 이것으로 농담을 하셨습니다. 빼갈도 와인잔에 담아 마시면서 혀의 정확한 부분에 술이 닿는다고... 훠궈는 원앙훠궈 (매운 국물과 담백한 국물을 반반씩 담은)를 시켰는데 들어가는 음식재료가 역시 다양하더라고요. 우리나라에서는 샤부샤부를 시키면 주로 한 가지 종류의 소고기를 넣어서 먹는데 소고기 종류도 다양하고, 양고기도 부위별로, 두부도 다양했습니다. 새우가 인상적이었습니다. 탱글탱글 입안 가득 만족감을 줍니다. 매운 국물에는 우리나라 고추와는 다른 고추가 들어가는데 이건 맵다기보다는 혀를 마비시킵니다. 결국 나중에는 맛을 제대로 느낄 수 없게 해서 안타까웠습니다. 끝에는 담백한 육수가 더 맛있더군요. 식사중에 대화에는 재밌게도 4가지 언어가 사용되었습니다. 한국어, 중국어, 영어, 프랑스어... 사모님은 한국어를 빼고 다 구사하시더군요.

이광근 교수님께서 장피에르 교수님을 한국으로 초빙하겠다고 하셨고 장피에르 교수님께서도 흔쾌히 응하셨습니다. 그날이 기대됩니다. 북경에서는 관광은 전혀 못했지만 맛있는 음식을 먹고, 장피에르 교수님을 알게 되어 뿌듯했습니다. 쓰고 보니 연구에 관한 이야기보다는 음식에 관한 이야기가 더 많네요.

8

블로그 : ROSAEC minutes

Workshop scene & schedule

2011-01-10

안녕하십니까.

지난 주 통영에서의 웍샵 모두 즐겁게 보내셨는지요.
케이블카 미륵산 정상서 본 남해가 그렇게 아름다울진 상상못했습니다.
(못 본 분들껜 죄송)

- 이번 웍샵에서 포스터와 뇌폭풍 모임이 잘 작동하는 것을 겪었습니다.
  다음 웍샵에 더욱 발전시킬 예정입니다.

- 발표들도 실질적이고 구체적인 것들이 주인공이 되었습니다.
  더욱더 겉치레발표가 없는 워크샵 방안을 고민중입니다.

- 4회 웍샵에서 꺼낸 "모국어 발표"의 취지에 호응하는 학생들이 많았습니다.
  어려운 연구일수록 일상어로 편하고 쉽게 발표하는 깊은 내공, 갖춰갑시다.

그리고 일정에 참고해 주십시요.

- 다음 여름 웍샵은 6월말에 가지겠습니다.

- 다음 겨울 웍샵은 2012년 1/11(수)-14(토) 홍콩과기대에서 열립니다.
  HKUST 교내의 방문객 숙소와 설비를 제공받기로 했습니다.

즐겁게 지내십시요,

-이광근 드림

Happy New Year

2011-01-05

교수님들께,
교수님 랩의 대표 논문들을 모집하겠습니다.
세부 사항은 행정원이 공지드리겠습니다.
센터 웹과 뉴스레터에 실리게 됩니다.
뉴스레터는 종이의 장점을 살린 인쇄물로 만들겠습니다.
대표 논문을 보내실 때, 한가지 부탁드리겠습니다.
대표 논문마다 작은 “이야기”를 보내주십시요. 웹과 뉴스레터에 논문마다 한 단락으로 붙이겠습니다.
100자 이내로, 연구에 대해서 공유하고픈 에피소드, 키워드, 리뷰 구절 등을 써 주십시요.

이유가 있습니다.
* 우리가 즐겨야겠습니다. 재미없는, 에피소드없는 연구는 피하자, 가 구호입니다. (억지로라도 만드셔서 같이 즐길수있게)
* 마음을 움직이면 그 분야가 발전합니다, 가 가설입니다. 연구에 대한 지극히 개인적인 “이야기”들,
  우리의 웹/뉴스레터에 그런 것들이 드러나도 좋다고 봅니다.
* 센터 평가에도 도움될 것입니다.

한 예를 첨부합니다. 교수님들의 개성을 발휘해주십시요.
예) “논문제목”
3개월간 방문했던 XXX교수 와의 공동연구 결과. 좋은 연구를 짧은 기간에 같이하게 된 아주 즐거운 경우였다.
연구중에 이거다 싶은 느낌의 연구였다. YYY이라는 분야를 프로그램 검증에 사용할 수 있는 시초를 만든 연구였다.
논문리뷰중에 “The paper is NNNN…”라는 평을 받았다.
키워드: AAA, BBB, CCC

이광근 드림

5th Workshop

2010-12-28

안녕하십니까,

연말에 겨울다운 날씨입니다.

오는 1/6-9일 통영에서의 웍샵이 기대됩니다.

알찬 씨를 뿌리고 키우는 시간이 될겁니다.

몇가지 전해드립니다.

- 포스터 발표가 의외로 적습니다.
  교수님들께서는 포스터 발표를 독려해 주십시요.

  지난 웍샵에서 포스터 발표를 통해서 많이 교류하고 좋았다고 합니다.
  예를들어 "Lightning"발표를 포스터로도 준비하면 좋지 않을까합니다.

- 발표 상을 준비합니다.
  우리말자료/좋은내용/재미 등등을 기준으로 시상합니다.

- 경품 추첨 + 단체 야외활동을(sports)하고 그 성적에 따라 푸짐한
  상품이 준비됩니다.

새해 남해바다에서 모이겠습니다,

-이광근 드림

This winter workshop

2010-11-22

안녕하십니까,

이번 겨울 ROSAEC센터 웍샵을 1월 초에 가지려고 합니다.

지난 여름 설악산 울산바위 웍샵에서는 좋은 추억들이 많았습니다.

웍샵 프로그램은 계속 진화합니다.

지난번 포스터 세션이 여러모로 좋았다는 의견이 많아서 확대하려고 합니다.

- 프로그램 위원장은 POSTECH의 박성우 교수님이 다시 맡아주시기로 했습니다 (감사합니다)

- 행정실에서 좋은 날짜를 교수님들께 poll해서 결정할 것입니다.

- 장소는 남해안 어딘가가 될것같습니다.

기대됩니다,

-이광근 드림

RDM report & EBS documentary

2010-10-26

교수님들께,

10월인데 춥습니다.

어제는 외국(미국과 한국 기독교)자본으로 설립된 평양과기대가 첫 수업을 시작했다는 기사를 외국신문(뉴욕타임즈)에서 봤습니다. 1면과 6면에 걸친 큰 기사였습니다. 국내 신문에는 전혀 보도되지 않았던데 의아했습니다.

- 교수님들의 RDM 리포트를 잘 받았습니다.
  학생들이 작성한 경우, 논문이 conference 논문인지 journal논문인지 혼동스러운 것들이 있습니다. 교수님들이 제출 전 검토해 주셨으면 했습니다.

- 몇 주 전에는 그동안 이야기되던 [컴퓨터분야 다큐멘터리]에 대한 발표가 있었습니다. EBS본사에서였고, EBS사장과 제작/편성/기획국장들과 피디 그리고 정보과학회 회장단이 모이셨습니다.

  EBS에서 공감하시고 예산배정 추진하시겠다고 합니다.
  회식에서 이런저런 의견/의기투합 막걸리잔도 몇 순 돌았습니다.
  실현될른 지 두고 볼 일입니다.
  어쨌거나 같이 탈 배를 강둑엔 올려논것같습니다.

  어떻게 내용을 채워가야 할 지 고민입니다.
  좋은 의견 주십시요.

-이광근 드림

9

연구실 소개 & 유학생 소식

KAIST 고신뢰성 소프트웨어 연구실

김문주 교수님 연구실 소개

KAIST 고신뢰성 소프트웨어 연구실은 2006년에 설립되어 현재 연구실의 수장인 김문주 교수님, 3 명의 박사과정 학생과 1명의 석사과정 학생, 그리고 방문 교수로 오신 University of Nebraska-Lincoln 의 Gregg Rothermel 교수님으로 구성된 젊은 연구실입니다. KAIST 고신뢰성 소프트웨어 연구실 구성원은 '어떻게 하면 이 땅의 소프트웨어 개발자가 쉽고, 빠르게 소프트웨어 버그를 잡을 수 있는지' 를 밤낮으로 고민하면서, 더 좋은 버그 찾는 기술을 개발하고 이를 실제 개발 환경에서 활용하는 방법을 연구하고 있습니다.

소프트웨어의 버그를 쉽고 빠르게 잡기 위해 저희 연구실에선 소프트웨어 테스트 케이스를 자동으로 생성하는 기법을 연구하고 있습니다. 기존에는 소프트웨어를 테스트하기 위해 개발자가 직접 테스트 케이스를 작성해야 해서, 개발자의 시간을 너무 많이 빼앗고, 개발자가 예측하지 못하는 경우를 테스트할 수 없는 단점이 있었습니다. 그래서 정적 분석 기법과 동적 분석 기법의 장점을 융합한 동적 기호 실행 기법을 통한 테스트 케이스 자동 생성 기법을 사용해서 소프트웨어 테스트에 들어가는 개발자 노력을 줄이고 테스트 효과는 높이고 있습니다. 최근에는 좀 더 복잡한 소프트웨어의 테스트 케이스를 효과적으로 생성하기 위한 연구에 박차를 가하고 있습니다. 수십, 수백 개의 계산 노드를 갖는 클라우딩 컴퓨팅 환경을 활용하는 분산 동적 기호 실행 기법을 개발해서, 테스트 케이스 생성 속도를 극대화 하는 기법을 개발, 적용하여 기존에는 너무 시간이 오래 걸려서 테스트 케이스를 생성하기 어려웠던 복잡한 소프트웨어도 빠른 시간 내에 테스트 케이스를 생성하고 있습니다.

저희 연구실에서는 새로운 버그 잡는 기술을 개발하는 것 못지 않게 버그 잡는 기술을 실제 개발 환경에 적용하는 것에도 많은 노력을 기울이고 있습니다. 특히 지난 2010년부터 삼성전자와 함께 모바일 폰 소프트웨어 플랫폼을 대상으로 동적 기호 실행 기법을 적용하는 연구를 공동으로 수행하면서, 동적 기호 실행 기법이 실제 개발 현장에 적용될 수 있는지, 개발자들에게 직접 도움을 주려면 어떤 연구를 더 수행해야 하는지를 중점으로 연구를 진행하고 있습니다. 특히 삼성에서는 실제로 자동 생성된 테스트 케이스가 기존에 잡지 못했던 버그를 잡는 모습을 보고 큰 관심을 보였으며, 앞으로 2012년까지 모바일 폰 소프트웨어 플랫폼에 특화된 동적 기호 실행 기법과 개발자들이 쉽게 사용하고 접근할 수 있는 도구를 같이 개발하기로 하였습니다.

연구자로서 최고 수준의 버그 잡는 기술을 개발하는 것은 KAIST 고신뢰성 연구실 구성원들의 당연한 꿈입니다. 하지만, 여기서 그치는 것이 아니라, 최고 수준의 버그 잡는 기술을 사용해서 개발자들의 짐을 덜어주고, 주변 세상을 더 편하게 바꾸어 나갈 수 있을 때 비로소 KAIST 고신뢰성 연구실의 꿈을 이뤘다고 할 수 있습니다. 지금도 꿈을 향해 한 발 한 발 나아가는 KAIST 고신뢰성 연구실 소개를 마칩니다.

서울대학교 비행역학 및 제어 연구실

김유단 교수님 연구실 소개

비행역학 및 제어 연구실은 故 위상규 교수님이 1955년에 부임하신 이후로 설립되어, 1991년까지 약 37년 동안 박사학위 1명, 석사학위 약 35명이 배출되었다. 이후 김유단 교수가 1992년 새로 부임하여 연구실을 운영하고 있으며, 박사학위 19명, 석사학위 68명을 배출하였다. 현재 박사과정 12명, 석사과정 9명으로 구성되어 있다. 비행역학 및 제어 연구실에서는 항공기, 인공위성, 발사체(로켓/미사일) 등의 항공우주 시스템을 대상으로 하여 동역학적 특성을 분석하고, 주어진 임무를 수행하기 위한 비행 제어시스템 설계에 관한 연구를 수행하고 있다.

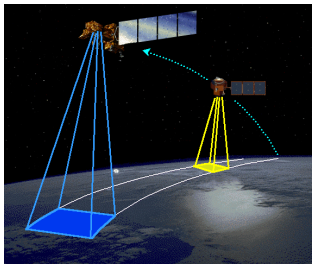
무인항공기의 자동이착륙 기술개발

본 연구에서는 무인항공기가 자율적으로 비행하여 임무를 수행한 후, 영상정보를 사용하여 목표 지점에 자동으로 착륙할 수 있도록 하기 위한 기술을 개발하는 것을 연구 목표로 하고 있다. 이를 위해 영상처리, 자동착륙을 위한 유도제어 알고리듬 개발, 비행제어 하드웨어 개발, PISL/HILS 지상시험 및 비행시험 부분으로 나누어 연구를 진행하고 있다. PILS/HILS 시스템은 실제 사용할 프로세서와 센서, 구동기 등 하드웨어를 시스템의 루프 상에 직접 연결하여 시뮬레이션을 수행함으로 지상에서 비행제어 시스템의 성능을 검증하는 과정이다.

복수무인기 분산시스템 연구

복수 무인항공기 시스템은 단일 무인항공기 시스템에 비하여 협동을 통하여 임무를 효율적으로 수행할 수 있으며, 개별 무인항공기에 고장이 발생해도 임무를 수행할 수 있기 때문에 단일 시스템에 비해 안정적이고 효율적으로 임무를 수행할 수 있다. 본 연구에서는 다양하고 복잡한 임무를 효율적으로 수행할 수 있도록 다수의 무인항공기들의 임무할당, 비행경로 생성, 그리고 충돌회피를 고려하여 군집비행을 수행하기 위한 알고리즘을 개발하고 있으며, 비행시험을 통해 성능을 검증하고 있다.

다수 위성의 정렬비행 연구



위성 정렬비행은 다수의 소형위성들이 편대를 이루기 때문에 위성의 지상 관측 효율을 증가시켜 주고, 동시 관측을 통해 수십 킬로미터의 기저선을 확보할 수 있는 장점을 제공한다. 또한 3차원 공간의 입체관측 등 신개념의 임무수행을 가능하게 해준다. 이와 같은 장점으로 인하여 미국을 비롯한 우주선진국에서는 1990년대 말부터 연구하기 시작하였으며, NASA에서는 위성 정렬비행 기술을 신개념의 위성임무 수행기술로 규정하여 다양한 관련 프로그램을 운용하고 있다. 본 연구에서는 위성의 정렬비행에 대한 알고리즘 개발 연구를 수행하고 있다.

지능형 자율비행체 시스템 개발

최근에 개발되고 있는 무인항공기는 높은 생존성 및 자율성을 요구하고 있다. 이러한 요구조건을 충족하기 위해서는 다양한 비행상황에 대해서 무인항공기가 스스로 판단하고 실시간으로 대처할 수 있는 자율비행제어 시스템 기술의 중요성이 증대하고 있다. 본 연구에서는 이와 같은 지능형 자율비행제어 시스템을 구축하기 위해 비행체간 및 지형지물과의 충돌을 방지할 수 있는 충돌회피 알고리즘, 예상하지 못한 고장에 대처할 수 있도록 고장진단 및 재형상 제어를 통한 고장여유 기술, 그리고 비행환경 변화에 적절히 대처하는 지능형 적응제어 기술에 대한 연구를 수행하고 있다.

POSTECH 정보및데이터베이스시스템연구실

황승원 교수님 연구실 소개

포항공과대학교 정보 및 데이터베이스 시스템 연구실은 황승원 교수님의 지도하에 박사과정 3명, 통합과정 7명, 석사과정 1명으로 구성된 연구실입니다. 본 연구실은 웹 환경 및 대용량 데이터베이스에서 검색 최적화 분야의 연구를 진행하고 있습니다. 또한 정보 검색 및 자연어 처리, 데이터 마이닝과 소프트웨어 개발 분야로 연구를 확장하고 있습니다. 구체적으로 본 연구실에서 진행하는 연구 분야를 소개하면 다음과 같습니다.

순위화 질의 및 스카이라인 질의의 최적화

대량의 데이터에서 사용자에게 가장 적합한 소량의 결과만을 추출하기 위하여 주어진 질의에 지능적으로 답변하는 질의 방법론을 개발하는데 목적이 있습니다. 구체적으로, 가장 활발히 연구되는 순위화 질의와 스카이라인 질의에 초점을 두고 이를 빠르게 계산하는 방법 및 이를 활용하는 방안에 대해서 연구를 진행하고 있습니다.

대용량 검색 결과의 분류화 및 군집화

대량의 검색 결과를 사용자에게 효과적으로 보여주기 위한 인터페이스로써, 분류화와 군집화 방법을 활용하여 사용자가 대량의 검색에서 자신의 원하는 결과를 빠르게 찾을 수 있게 도와주는데 목적이 있습니다. 이를 위해 전체 검색 결과에서 관련 주제들을 유사성을 측정하여 효과적으로 이를 정리하는 방법에 대해서 연구를 진행하고 있습니다.

사회적 그래프를 이용한 이름 번역 및 관계 유추

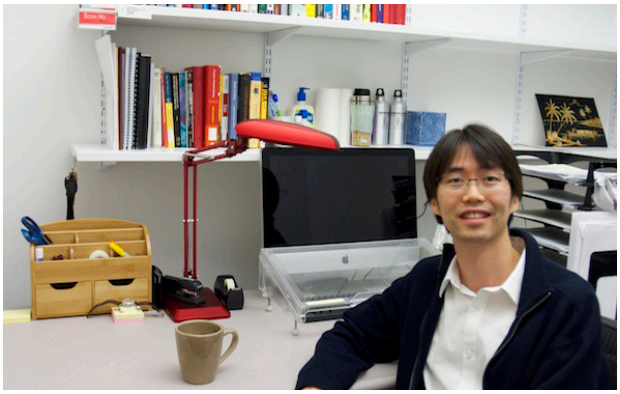
웹 문서에 등장하는 인명이나 기관명과 같은 개체들이 하나의 문서에 동시에 등장할 때, 개체들 간의 암시적 관계로 인지하여 그래프를 구성함으로써, 다양한 정보를 유추해 낼 수 있습니다. 구체적으로, 서로 다른 언어로부터 구성된 두 개의 개체 그래프를 매칭하고, 매칭된 개체 쌍을 마이닝하여 이를 이름 번역에 활용하는 방법에 연구가 진행 중이며, 또한, 이를 통해 추가적으로 알려지지 않은 관계를 유추 방법에 대해서 관심을 가지고 있습니다.

효율적인 소프트웨어 개발을 위한 소스 코드의 색인, 오류 보고서 및 라이선스 관리

이미 개발된 대규모의 소프트웨어 코드에서 코드들간의 구조적 유사도에 의한 유사 검색을 실행하기 위해, 코드의 구조를 벡터로 근사하여 표현하는 기법에 기반하여 효율적 색인 기법을 연구하고 있습니다. 또한 발생한 버그를 관련 개발자에게 정확하게 매칭하는 방법과 복제된 오픈 소스 코드에서 잠재적 라이선스 위반을 감지 및 개발자에게 경고하는 방법 등 데이터베이스와 소프트웨어 개발 분야가 연계된 연구가 진행 중입니다.

유학생 소식

Carnegie Melon University, Computer Science Department, Ph.D 과정, 공순호



소개

안녕하세요, 저는 현재 카네기 멜론 대학교(이하 CMU) 컴퓨터 과학과에서 Edmund Clarke 교수님의 지도를 받으며 프로그램 분석분야에서 박사과정을 밟고 있는 공순호 입니다. 저는 2007년부터 2009년까지 서울대학교 프로그래밍 연구실에서 이광근 교수님의 지도로 석사과정을 졸업하였고, 그 이후 1년간 ROSAEC 센터에서 연구원으로 근무하였습니다. 2010년 8월에 이곳 카네기 멜론으로 유학을 왔고, 한 학기를 무사히 마무리하였습니다. 새로운 곳에 와서 적응하면서 저는 자연스럽게 제가 예전에 머물던 학교/연구실과 이곳의 차이점에 주목하게 되었습니다. 저는 이 글을 통하여서 그러한 차이점들을 여러분과 나누고자 합니다.

느슨하게 연결된 연구 그룹

CMU에 와서 처음 느낀 것은 연구 그룹 구성원들 간의 관계가 더 개인적이고 연구 중심적이라는 것입니다. 특히 지도 교수님을 공유하는 학생들 간의 교류가 부족하다는 것이 눈에 띄는 차이점이었습니다. 한 지도 교수님의 학생들이라고 해도, 프로젝트나 세부 연구 관심사가 겹치지 않으면 서로의 연구 현황에 대해서 잘 알지 못하는 경우가 많습니다. 처음 이러한 것을 발견했을 때는 놀라웠지만, 대신에 그들은 세부 연구 관심사를 공유하는 다른 지도 교수님의 학생이나, 다른 학교 학생/연구진들과 교류한다는 것을 알 수 있었습니다.

줄어든 발표의 기회

CMU에서 아쉬운 것 중의 하나는 저의 연구 결과를 주기적으로 발표할 기회를 찾기 어렵다는 것입니다. 지도 교수님과의 연구미팅에서 아이디어를 설명하고 이야기를 나눌 수는 있지만, 여러 사람 앞에서 발표하는 것과는 차이가 있습니다.

프로그래밍 연구실에 있었을 때에는 매주 만나서 연구실 사람들의 연구 성과를 발표하는 “ROPAS Show & Tell” 행사가 있었습니다. 적어도 두 달에 한 번 정도는 내가 현재 진행하는 연구를 사람들에게 설명하고 이에 대한 다른 사람들의 평가를 받을 수 있었습니다. 이를 통해서 부족한 부분을 보완하고, 핵심 아이디어를 더 잘 설명하는 방법을 찾아냈습니다. 거꾸로 다른 사람의 연구 발표를 들으면서 새로운 연구 주제를 찾아가는 계기를 마련하기도 했습니다. CMU에서도 비슷한 행사를 조직해보려고 합니다.

잘조직된 논문읽기 모임

이곳에는 여러 분야의 논문 읽기 모임이 오랜 역사를 통해서 잘 조직되어 있습니다. 프로그래밍 연구실에 있는동안 논문 읽기 모임을 시도했었는데 지속적으로 유지되지 못했던 기억이 있습니다. 프로그래밍 언어 분야의 논문 읽기 모임인 “ConcertRG”의 경우에는 다음과 같이 진행됩니다. 먼저, 모임의 회장이 투표를 통해서 다음 주에 함께 읽을 논문을 결정합니다. 모임은 다 함께 모여서 논문에서 잘 이해되지 않았던 것들의 목록을 작성하면서 시작됩니다. 보통 10개 이상의 질문들이 모이게 되고, 질문에 답할 수 있는 사람이 돌아가면서 궁금한 것들을 설명하게 됩니다. 논문 읽기 모임을 통해서 다양한 논문들을 접하게 되는 것이 좋았고, 고년차 박사과정 학생들의 설명을 통해서 얻게 되는 배경지식이 유용했습니다.

맺음말

글을 통해서 차이점에 대해서 말씀드렸지만, 사실 연구 과정에서 많은 공통점을 발견하게 됩니다. 능동적으로 주도권을 가지고 연구에 참여해야 한다는 것, 이론적인 결과물에 머무르는 것이 아니라 실제로 사람들에게 도움을 주는 유용한 도구의 개발까지 이어져야 한다는 것을 강조하는 Edmund Clakre 교수님의 충고를 들으면서, 이광근 교수님께서 예전에 똑같은 말씀을 해주셨던 것을 생각했습니다.

10 참여교수 연락처

제1총괄과제

1-1세부과제	교수 **백윤흥**	02-880-1748		ypaek@ee.snu.ac.kr	서울대학교 전기공학부
1-2세부과제	교수 **이광근**	02-880-1857		kwang@ropas.snu.ac.kr	서울대학교 컴퓨터공학부
	교수 **우치수**	02-880-6573		wuchisu@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	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
	교수 **김유단**	02-880-7398		ydkim@snu.ac.kr	서울대학교 기계항공공학부
2-4세부과제	교수 **최광무**	042-350-3520		choe@kaist.ac.kr	KAIST 전산학과

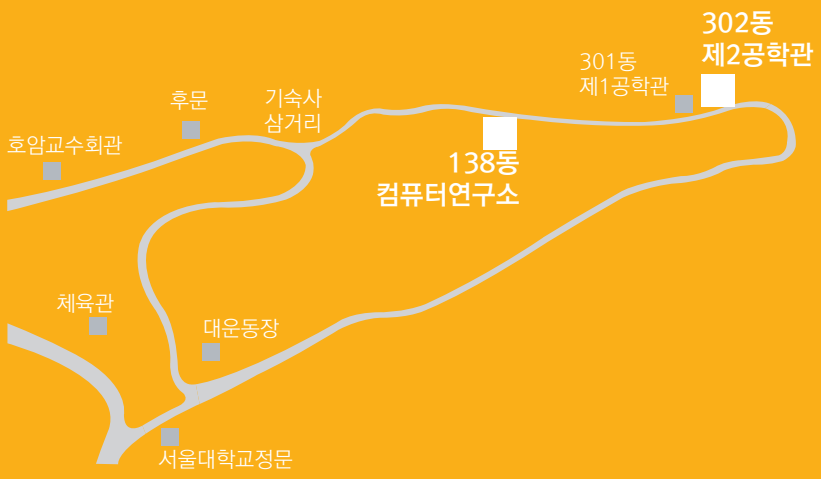
제3총괄과제

3-1세부과제	조교수 **박성우**	054-279-2386		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 **최진영**	02-3290-3200		choi@formal.korea.ac.kr	고려대학교 컴퓨터통신공학부
3-3세부과제	조교수 **황승원**	054-279-2385		swhwang@postech.ac.kr	POSTECH 컴퓨터공학과
	부교수 **한욱신**	053-950-7572		wshan@knu.ac.kr	경북대학교 컴퓨터공학과

위탁과제

위탁과제1	부교수 **안준선**	02-300-0144		jsahn@kau.ac.kr	한국항공대학교 정보통신공학부
위탁과제2	부교수 **이승용**	054-279-2245		leesy@postech.ac.kr	POSTECH 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 | 2011년 1월
발행인 | 이광근
편집인 | 신재호
발행처 | ROSAEC center

서울시 관악구 관악로 599 서울대학교 138동 202호
tel 02-880-7290
fax 02-882-7234
mail rosaec@snu.ac.kr
web rosaec.snu.ac.kr

ROSAEC center

Research On Software Analysis for Error-free Computing

소프트웨어 무결점 연구센터 NRF ERC

151-742 서울시 관악구 관악로 599 서울대학교 138동 202호
Rm 202 Bldg 138, Seoul National University, 599 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