First International
SAT/SMT Solver Summer School 2011

Sunday, June 12- Friday, June 17, 2011
MIT, CAMBRIDGE, MA, USA

서울대학교 프로그래밍 연구실
조성근

이 여름학교는 이번이 처음 열린 것이었습니다. SAT/SMT solver에 관련한 여러 주제(solver 개발, 복잡도, non-CDCL 알고리즘, 멀티코어 이용 등)를 다루었습니다. 여름학교는 우리의 기대와는 달리 SAT/SMT solver 초심자를 위한 것은 아니었습니다. 대부분의 발표는 solver의 기본적인 내부 동작을 알고 있음을 가정하고 진행되었습니다. 여름학교에 참가하기 전에 SAT/SMT solver를 사용할 기회가 있었으면 더욱 좋았겠지만, 반대로 solver들의 적용 사례들을 많이 접할 수 있어서 좋았습니다.

여름학교에서의 발표는 크게 세 가지 종류로 분류되었습니다.[1]

1. SAT/SMT solver의 기초 (Foundational Aspects of SAT/SMT Solvers)
2. 특화된 SAT/SMT solver (Description of SAT/SMT Solvers with tight focus on an application)
3. SAT/SMT solver를 사용하는 툴 (Description of tools using SAT/SMT Solvers)

1에 속한 발표는 주로 복잡도나 알고리즘과 같은 이론적인 내용이었고, 2에 속한 발표는 SAT/SMT solver의 개발자들의 solver 소개가 주를 이루었습니다. 3에 속한 발표에서는 SAT/SMT solver의 적용 사례를 알 수 있었습니다. 5일 동안 있었던 발표 중 인상 깊은 발표들을 정리해 보았습니다.

SAT/SMT solver 의 기초

Introduction to Satisfiability Solving with Practical Applications

Niklas Een, University of California, Berkeley, USA

SAT solver의 기본적인 동작 원리와 적용 사례를 보여준 발표였습니다. SAT 문제를 정의하고 DPLL알고리즘과 CDCL(conflict analysis)를 설명하였습니다. 적용 사례로는 slither link라는 퍼즐 문제와 회로 검증을 보여주었습니다. 복잡한 연구 내용을 다루고 있지는 않지만 저와 같은 초심자들이 SAT solver의 원리를 이해하는 데에 큰 도움을 주는 발표였습니다. 관련자료를 발표 슬라이드 페이지에서 볼 수 있습니다.

Non-DPLL Approaches to Boolean SAT Solving

Bart Selman & Carla Gomes, Cornell University, Ithaca, USA

대부분의 SAT/SMT solver가 DPLL에 기초하고 있었기 때문에 non-DPLL 접근법의 사용은 흥미로운 주제였습니다. 발표의 전반부에는 non-DPLL 알고리즘인 WalkSat 알고리즘을 소개하였고, 후반부에는 WalkSat 알고리즘을 사용하는 Walksat solver와 MiniSat

[1] 발표자료를 다음 페이지에서 볼 수 있습니다.
http://people.csail.mit.edu/vganesh/summerschool/lectures.html

solver[2]를 혼합하여 사용하는 방법에 대하여 소개하였습니다. 여기서는 WalkSat 알고리즘의 기본 아이디어를 간단히 소개하겠습니다.

WalkSat 알고리즘은 random walk 알고리즘에 지역적인 편견(greedy bias)을 가미한 방법입니다. Random Walk 알고리즘은 이름과 같이 임의의 선택으로 답을 찾습니다. False의 값을 가지는 clause 중에서 임의의 clause를 선택하고, 그 clause에서 임의의 literal을 선택하여 값을 바꾸는 동작을 반복합니다.

WalkSat 알고리즘은 Random Walk 알고리즘에 두 가지 변수 선택 방법이 추가됩니다.

- Freebie: c에서 값이 바뀌어도 True인 clause의 값을 바꾸지 않는 literal이 있다면 그 값을 뒤집습니다.
- 가장 작은 break value를 가지는 literal: 어느 literal의 값이 바뀌었을 때 True였다가 False가 되는 clause의 개수를 그 literal의 break value라고 하는데 그것이 작은 literal을 선택하여 값을 뒤집습니다.

WalkSat 알고리즘은 서로 의존하는 변수(dependent variable)들이 많지 않은 문제들, 예를 들면, random k-SAT, graph coloring, circuit synthesis 등의 문제에서는 효과적이라고 합니다. 하지만 의존하는 변수가 많은 하드웨어/소프트웨어 검증과 같은 문제에서는 덜 효과적이라고 합니다.

문제의 해가 존재하지 않는 다는 것을 보이기에는 적합하지 않지만, 문제를 바라보는 새로운 시각을 제시한다는 점에서 의미가 크다고 생각합니다. 우리가 접하는 문제들도 언제나 새로운 시각에서 바라볼 수 있도록 연습이 필요하다고 생각하였습니다.

특화된 SAT/SMT

HAMPI: A Solver for String Theories

Vijay Ganesh, MIT, Cambridge, USA

HAMPI는 테스팅, 분석, 취약성 탐지를 위한 문자열 solver입니다. 즉, 문자열에 대한 식을 받았을 때 이 식을 만족하는 문자열이 존재하는지 존재한다면 어떠한 문자열이 있는지 알려주는, 문자열 논리를 포함하고 있는 SMT solver입니다. 이 발표에서 Vijay Ganesh는 HAMPI의 문자열 논리와 동작 원리, 실험 결과를 소개하였습니다.

---

[2] Niklas Eén 과 Niklas Sörensson 가 만든 SAT solver 로서 2005 SAT competition 의 세 부문에서 상을 받았습니다. 다음 페이지에서 이용할 수 있습니다. http://minisat.se/

HAMPI의 가장 큰 특징은 대상으로 하는 문자열의 크기를 20으로 제한한다는 점입니다. SMT solver에서 문자열의 길이를 짧게 제한하여도 테스트를 위한 문자열을 만들거나 취약성 탐지를 위해서는 충분히 유용하다는 것이 Vijay 연구팀의 주장입니다. 실험 결과로 보여준 SQL 끼워넣기 분석(injection analysis)에서는 대부분의 분석이 1초 안에 끝났습니다.

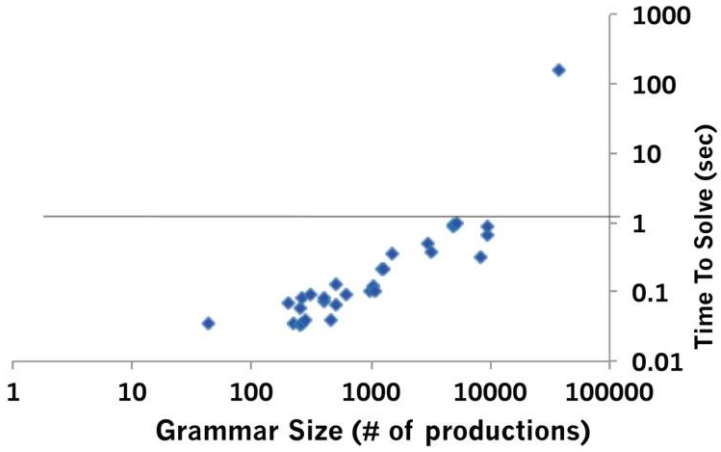


그림 1 문법의 크기와 분석시간 (Vijay 의 발표자료로부터)

간단한 아이디어와 좋은 결과. 훌륭한 연구라고 느꼈습니다. ISSTA에 제출된 HAMPI 논문은 2009년도에 ACM 최고 논문 상을 받았습니다.

Yices and Applications

Bruno Dutertre, SRI International, Menlo Park, US

이번 여름학교에는 여러 SAT/SMT solver 개발자들의 발표가 있었는데, 그 중 Yices라는 SMT solver에 대한 발표입니다. 발표의 전반부에는 Yices의 성장 과정과 현재 Yices의 특징들에 대해 이야기하였고, 발표의 후반부에는 적용 사례에 대한 이야기를 하였습니다.

먼저 이전 버전인 Yices1과 Yices2의 구성과 성능을 비교하였습니다. 이전 버전에서 최대한 많은 종류의 이론(theory)들을 지원하는 것이 목표였다면은 새롭게 개발된 Yices2의 주된 목표는 사용의 유연함과 성능 향상에 있었습니다. 또한 Yices의 이전 버전이 너무 복잡해져서 새롭게 디자인을 시작했다는 이야기를 통해 solver 자체의 확장성에도 신경을 쓰고 있음을 알 수 있었습니다.

하지만 기업에서 만드는 solver이기 때문인지 내부 구현에 대해서는 자세한 이야기를 들을 수 없었습니다.

적용 사례. 실시간 분산 시스템에서의 스케쥴링

보여준 문제는 분산 시스템에서의 통신 스케쥴링 문제였습니다. 분산 시스템에서 통신이 이루어질 때 어떤 제약들이 지켜지는지 확인하는 것입니다. 그런 제약들은 다음과 같이 SMT solver의 입력으로 변환될 수 있습니다.

- 서로 다른 프레임(f, g)이 같은 링크를 동시에 사용해서는 안 된다.
  $\text{offset}_{f,i} + f.\text{period} \leq \text{offset}_{g,i}$ 또는 $\text{offset}_{g,i} + g.\text{period} \leq \text{offset}_{f,i}$
- 중간 노드는 정보를 받은 후에야 그 정보를 보낼 수 있다.
  $\text{offset}_{f,j} - \text{offset}_{f,i} \geq \text{maxhopdelay}$
- 경로가 $i_0, i_1, \cdots, i_n$일 때 최고 지연 시간이 제한된다.
  $\text{offset}_{f,i_n} - \text{offset}_{f,i_0} \leq \text{maxlatency}$

Yices1에서는 약 120개의 링크가 존재하는 문제를 풀 수 있었지만, Yices2에서는 1000개까지의 링크가 존재하는 문제도 풀 수 있도록 성능이 개선되었다고 합니다.

SAT4J: pseudo-boolean optimization & dependency management problems

Daniel Le Berre, Université d'Artois, France

SAT4J는 Java의 SAT solver 라이브러리입니다. 발표자는 소프트웨어의 의존성을 해결하는 데에 SAT solver인 SAT4J를 이용하였습니다. 발표에서는 소프트웨어의 의존성 문제가 어떻게 SAT 문제로 변환되는 지와 실험 결과를 보여주었습니다. 이 연구 결과는 2008년 7월부터 Eclipse의 플러그인의 의존성을 해결하는데 사용되고 있다고 합니다. 겨우 5개의 Eclipse 플러그인을 설치하는 데에 다음과 같이 많은 프로그램과의 의존성을 검사해야 합니다.

그림 2 Eclipse 에서 플러그인 의존 그래프. (Daniel 의 발표자료로부터)

여기서 소프트웨어의 의존성은 크게 두 가지로 정하고 있습니다. 1) 어느 프로그램이 설치되기 위해 설치되어 있어야 하는 프로그램들, 2) 함께 설치될 수 없는 프로그램들. 2)는 버전이 다른 같은 프로그램이 설치될 때 만족되어야 하는 조건입니다. 이 두 문제는 SAT 문제로 쉽게 변환될 수 있습니다.

이 연구의 목적은 의존성을 만족하는 어떤 답 하나 만을 얻고자 하는 것이 아니라, 그런 의존성을 만족하는 좋은 답을 얻는 것을 목표로 하고 있습니다. 예를 들면,

1. 설치되는 프로그램의 개수를 최소화,
2. 설치되는 프로그램의 크기를 최소화,
3. 가능한 최신 프로그램의 설치.

이러한 문제를 표현하고 해결하는 방법으로 진리값을 이용하는 부등식으로 문제를 해결하는 PBO(Pseudo Boolean Optimization), 가중치를 주어 전체 합의 최대값을 구하는 PWMS(Partial Weighted MaxSat)이 있을 수 있습니다. 발표자는 이 두 방법을 비교하고 장단점을 이야기하였습니다. PBO에 대해서 자세히 설명하였는데 PBO 문제를 해결하는 방법이 CDCL 알고리즘을 거의 그대로 사용하고 있다고 설명하였습니다. 즉, 풀어야 하는 진리값의 부등식들을 이용해서 새로운 부등식을 이끌어내어 해를 구하는 데에 이용하는 방식입니다.

SAT/SMT solver 를 사용하는 툴

Sketching: Program Synthesis using SAT Solvers

Armando Solar-Lezama, MIT, Cambridge, USA

프로그램 합성에 SAT/SMT solver가 어떻게 사용되는가에 대한 발표였습니다. 먼저 프로그램 합성 방법에 대해 간단히 설명하고 직접 코딩을 함으로써 sketch라는 프로그램 합성툴의 사용 과정을 보여주었습니다. 발표 내용이 SAT/SMT solver와 직접적 관련은 적었지만 다양한 곳에서 SAT/SMT solver의 계산 능력을 이용할 수 있다는 느낌을 받을 수 있었습니다.

프로그램 합성은 사람의 직관과 컴퓨터의 계산능력을 이용하는 데에 그 초점이 맞추어져 있습니다. 사람이 만들어질 프로그램의 틀과 생성된 프로그램이 만족해야 할 조건들을 입력하면 컴퓨터가 그에 맞는 프로그램을 찾는 것입니다. 만들어질 프로그램의 틀을 제공함으로써 컴퓨터가 프로그램을 생성하기 위해 찾아야 하는 문제 공간의 크기를 줄입니다. 또한 컴퓨터가 조건에 맞는 프로그램을 찾게 되므로 사람이 직접 프로그램을 작성했을 때 발생할 수 있는 버그를 줄일 수 있습니다.

프로그램 합성 = curve fitting

발표자는 프로그램 합성을 curve fitting (주어진 조건에 가장 적절한 함수를 찾는) 과정으로 설명하였습니다. 그 과정에서 CEGIS(CounterExample-Guided Inductive Synthesis)라는 방법을 사용하게 됩니다. 간단히 설명하면 합성된 프로그램의 입력 중 조건을 만족하

지 않는 반례를 통해서 프로그램을 정교하게 합성해가는 방법입니다. 조건을 만족하는 함수를 찾는 것이 코너 케이스에 집중한 몇 개의 입력만으로 충분하다는 것이 발표자의 주장입니다.

발표자는 sketch라는 프로그램 합성툴을 소개하고 SAT/SMT solver가 이용되는 부분 소개하였습니다. 현재는 bit연산 관련 부분에 SAT solver가 사용되고 있고, 정수 연산 관련 부분에 SMT solver가 사용되고 있습니다.

Symbolic Execution and Automated Exploit Generation

David Brumley, CMU

여름학교 발표 중 가장 마지막 발표였습니다. 5박 6일의 긴 일정이 있었기에 집중도가 떨어질 만도 했는데 사람들에게서 탄성을 자아내는 실험을 보여주었습니다. 단, 몇 초 만에 리눅스에서 사용되는 iwconfig 프로그램을 기호실행(symbolic execution)으로 분석하고, 운영자 권한을 얻어내는 프로그램 입력을 자동으로 생성하여 실제 운영자 권한이 얻어지는 모습을 보여주었습니다.

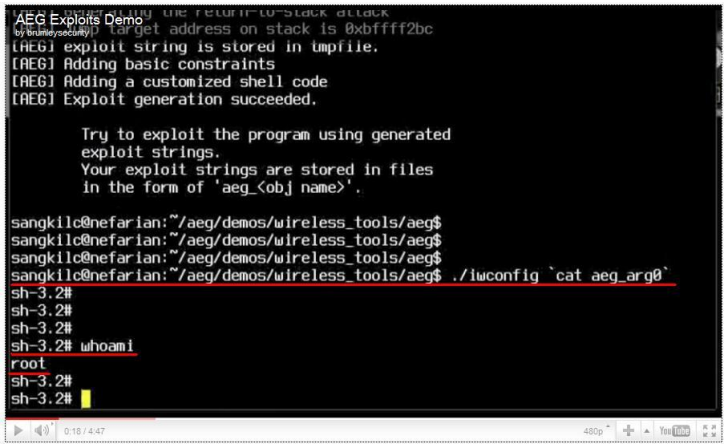


그림 3 iwconfig 프로그램으로부터 운영자 권한을 얻는 장면[3]

핵심 아이디어는 소스코드 분석과 바이너리코드 분석을 동시에 이용하는 것입니다. 소스코드 분석 만으로는 프로그램이 실행될 때의 메모리 상태를 알기 어렵습니다. 반면, 바이너리코드 분석만으로는 커다란 프로그램에 대한 정교한 분석이 어렵습니다. 기호실행을 통하여 소스코드를 분석하여 코드에서 잠재적인 버그를 가진 부분까지의 입력을 만들

[3] 관련 동영상을 웹에서 확인할 수 있습니다.
http://security.ece.cmu.edu/aeg/index.html

어내고, 바이너리코드를 분석하여 운영자 권한이 얻어지기 위한 적절한 입력을 찾아내어, 잠재적인 버그가 정말 악성 사용자들로부터 악용 가능한(exploitable) 것인지 확인합니다.

또한 두 번째 중요한 아이디어는 효과적인 기호실행에 있었습니다. 소스코드를 기호실행하는 과정에서 오직 악용 가능할 것 같은 경로에 집중합니다. 이를 위해서 경로에 우선순위를 매기는 방법(path prioritization)을 제안하였고, 그 정보를 이용하는 기호실행(preconditioned symbolic execution)을 제안하였습니다.

Brumley 연구팀은 총 16개의 프로그램으로부터 취약점을 발견하여 이를 이용하는 코드를 생성하였는데, 이 중 두 개(htget, expect)는 처음으로 발견된 것이라고 합니다. 실제 적용 가능한 이로운 연구를 하고 싶다는 생각을 다시금 하게 되었습니다.

관련 논문은 CAV 2011에 tool paper로 뽑혔습니다. 연구 내용도 훌륭했지만 발표 전반에 인상적인 실험 결과를 보여줌으로써 청중들을 사로잡는 모습이 인상적이었습니다.

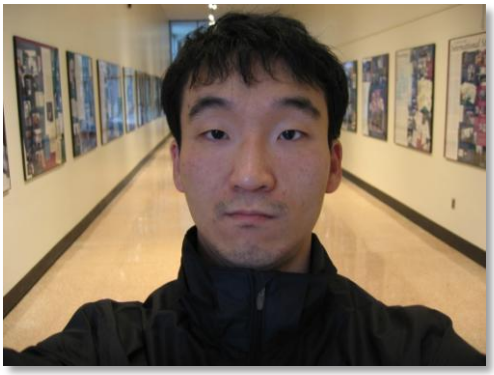
구경 거리

운 좋게도 이번 미국 방문 기간 중 축제가 두 개나 있었습니다. 하나는 게이 축제인 "Boston Pride"라는 축제였고, 다른 하나는 Boston Bruins라는 아이스하키 팀이 어느 대회에서 우승한 것을 기념하는 축제였습니다.

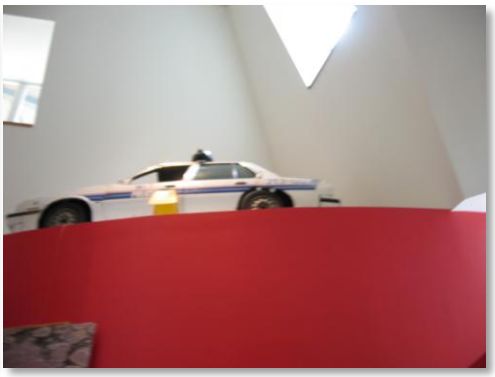
"Boston Pride"는 축제는 미국에 도착한 다음 날 승중이와 주변 공원을 산책하려고 나섰다가 운 좋게 볼 수 있었습니다. 비가 추적추적 오는데도 퍼레이드를 하는 사람들과 그것을 구경하려는 사람들로 거리가 가득했기에 활기를 느낄 수 있었습니다. 게이 축제라고 하여 남자들만 득실거리는 축제는 아니었습니다. 남녀노소가 모두 축제에 참여하는 그런 분위기였습니다. 거기에 정치가들도 상당 부분 차지하고 있었는데 그 분들은 특별히 재미있지는 않았습니다. 그 날 따라 손을 잡고 다니는 남남 쌍을 쉽게 볼 수 있었습니다. 승중이와 저는 최대한 떨어져 걷기로 하였습니다.

축제는 모두 퍼레이드 형식으로 진행되었습니다. 이는 매우 건전한 그리고 건강한 놀이라는 생각이 들었습니다. 주말에 밖에 나가서 사람들을 만나고 대화하며 함께 걸으며 스트레스를 해소하는 것. '이런 건전한 문화가 우리나라에도 자주 있다면 얼마나 좋을까?' 하고 생각했습니다. 집에서 컴퓨터 게임으로 스트레스를 풀던 제 자신도 돌아보게 되었습니다.

첫 날엔 덕환 선배가 MIT의 이곳 저곳을 구경시켜 주었습니다. 미국 가기 바로 전 날까지 기말고사가 있었던 저희는 관광에 대한 많은 준비를 하지 못 했었는데 덕환 선배 덕분에 많은 곳을 구경할 수 있었습니다. MIT의 도서관들, 교실, 복도, 연구실 등등.

도서관은 사람이 많지 않아서, 그리고 딱딱한 분위기가 아니라서 상당히 인상적이었습니다. 덕환 선배는 "미국 학생들은 어디서나 공부를 잘 하기 때문에 꼭 공부하러 도서관에 오지는 않는 것 같다."고 하였습니다. 그래서인지 저희 같은 외부인도 도서관에 쉽게 들어가서 책을 보고 시설을 이용할 수 있었습니다.

복도에는 여러 볼 거리가 있었습니다. 학생들의 기발한 작품들과 학생들의 무모하고 기발한 작품들. 위에서 왼쪽의 사진은 종이로 만든 하얀 나비들이 공중에 매달려 있는 복도입니다. 단지 하얀 종이였을 뿐인데 한참을 서서 바라보았습니다. 자신들의 공간을 이렇게 열심히 꾸미는 모습을 보니 부럽다는 생각을 하였습니다. 오른쪽 사진은 엄청 높은 곳에 올라가 있는 자동차입니다. 미국 학생들은 '자동차를 왜 올렸는가?' 보다는 '어떻게 올렸을까?' 를 먼저 떠올리나 봅니다. 이런 무모하면서 기발한 작품들이 곳곳에 자리 잡고 있어 어디서나 구경하고 생각할 것이 많았습니다. 다만, 사진을 더 많이 찍지 못 한 것이 아쉽습니다.

생활

숙소에서 발표장까지의 거리가 어느 정도 있었기 때문에 지하철을 이용하였습니다. 그 전에 미국에 방문했을 때에는 묵었던 호텔에서 발표가 있었기 때문에 바깥 세상을 구경할 기회가 거의 없었는데 이번에는 달랐습니다. 지하철 표도 사고 지하철을 타도 여기저기 구경도 할 수 있었습니다. 앞으로 또 기회가 있다면 숙소를 약간은 멀리 잡는 것이 좋겠다고 생각하였습니다.

인상적이었던 것은 미국 사람들의 '인사' 였습니다. 잘 모르는 사람끼리도 눈이 마주치면 쉽게 인사와 미소를 건넬 수 있었습니다. 지하철에서나 숙소에서나. 우리나라에서는 내가 시선을 돌리다가 행여 눈이라도 마주칠까, 날 이상한 사람으로 생각하지는 않을까 조마조마하였었는데 말이죠. 덕분에 낯선 곳이었지만 생활하는 것이 그렇게 불편하지는 않게 느껴졌습니다. 눈을 마주치고 못 본 척 할 필요가 없었습니다.

글을 맺으며

이렇게 좋은 기회에 견문을 넓힐 수 있도록 기회를 주신 교수님과 ROSAEC 센터에 감사를 드립니다. 또한 낯선 곳에서 쉽게 적응할 수 있도록 큰 도움을 주시고 여러 조언을 아끼지 않으신 덕환 형님께도 감사를 드립니다.