Smart Contracts Verification (89400)
Lecture 2

Yoni Zohar – Bar Ilan University

Seminar
Outline

1. Updates
2. Automated Reasoning
3. Reading a Paper
4. Presenting a Paper
5. Schedule
Updates

- Welcome, new students!
- Introductions
- New papers
- Updated Class Structure

Classes

- Each lecture is 45 minutes
- Each class there are two options:
  - Two lectures
  - A single lecture + discussion
  - Discussion:
    - Questions and elaboration on current lecture
    - Questions and elaboration on previous lectures
    - Other
### Automated Reasoning

#### Tools
- SAT-solvers
- SMT-solvers
- Theorem-provers
- Proof assistants
- Synthesizers
- ...

#### Applications
- Scheduling Problems
- Software Verifications
- Hardware Verification
- Compiler Optimization
- Test Generation
- Smart Contracts Verification
SAT-solvers?

Reminder

- SAT problem: \((x_1 \lor \neg x_2 \lor x_3) \land \ldots \land (x_5 \lor \neg x_6)\)
- Is the input formula SAT?

SAT

- SAT is NP-complete
- Best known algorithm is exponential time (worst case)
- Still, there are SAT-solvers
- Yearly Competition
- Used in many applications
How is that possible

- Smart Algorithms (DPLL, CDCL, local search, etc.)
- Much better in practice than naive search
- Heuristics (e.g., what variable to guess)
- Implementation details (e.g., caching, data structures)

Applications

- Equivalence Checking
- Search Problems
- Verification
- Math
- NP complete – Every NP problem is polynomial time reducible to SAT
- But Useful! – Every NP problem is polynomial time reducible to SAT
Pythagorean Triples

- https://www.comp.nus.edu.sg/~gregory/sat/
- https://www.cs.utexas.edu/~marijn/ptn/
- Can you color $1, \ldots, n$ in blue and red with no monochromatic Pythagorean triple?
- $a^2 + b^2 = c^2$

Example

- $n = 5$: nly triple is 3, 4, 5
- Make sure these don’t have the same color 1 2 3 4 5

Example

- $n = 10$: triples – 3, 4, 5 and 6, 8, 10
- 1 2 3 4 5 6 7 8 9 10
Pythagorean Triples

Theorem [Heule et al. 2016]
- There exists $n$ for which no such coloring exists.
- $n = 7825$

Proof
- Using a SAT solver

Encoding
- Boolean variables: $x_1, x_2, \ldots$
- $x_i$ is true iff $x_i$ is red. Otherwise $x_i$ is blue.
- Being non-mono-chromatic $\equiv$ Having both blue and red
  - $x_3 \lor x_4 \lor x_5$
  - $\neg x_3 \lor \neg x_4 \lor \neg x_5$
  - $\ldots$
- ./ptn-encode 13

Smart Contracts Verification
SMT-solvers?

SMT

- Satisfiability Modulo Theories
- SAT allows only to use Boolean variables
- SMT is much more general and flexible
- e.g. $x + y < 5 \land y^2 = \text{len}(s)$

How Is That Possible?

- In general, SMT is undecidable
- Still, SMT-solvers exist
- Integrated in many verification tools
- Yearly competition
Demo

**SMT for Solidity**

- [https://cvc4.github.io/app/](https://cvc4.github.io/app/)

```lisp
(set-logic ALL)
(declare-const a0 Int)
(declare-const b0 Int)
(declare-const b1 Int)
(declare-const b2 Int)
(declare-const b3 Int)
(declare-const b4 Int)

(assert (<= 0 a0))
(assert (< a0 (^ 2 256)))
(assert (<= 0 b0))
(assert (< b0 (^ 2 256)))
(assert (= a0 0) (<= b0 100))
(assert (= b1 1000))
(assert (= b2 10000))
(assert (= b3 (ite (= a0 1) b1 b2)))
(assert (= b4 (ite (= a0 0) b0 b3)))
(assert (not (<= b4 100000)))

(check-sat)
```

Smart Contracts Verification
## Summary

- Exciting field
- Many applications
- Theoretical hardness vs. Practical feasibility
- Theory and implementation

## Challenges

- Active field of research
- Current Challenges
  - certifying results (proofs)
  - scalability
  - **Smart Contracts Verification**
Tips – 1

- Start early
- Read background material
- Papers are rarely fully self-contained
- Ask for help, via email or a meeting
- Start Early

Tips – 2

- Look for references in the paper
  - for background material
- Look for references of the paper
  - for a more general understanding
  - google scholar
The Three Pass Approach

Read more than once

- [https://web.stanford.edu/class/ee384m/Handouts/HowtoReadPaper.pdf](https://web.stanford.edu/class/ee384m/Handouts/HowtoReadPaper.pdf)
- Reading once from start to finish often does not work
- Ideas need to be absorbed
- Understanding requires time
Three Passes

First Pass:
- title, abstract
- section titles
- references
- contributions

Second Pass:
- “normal” reading
- write notes
- mark notions, questions, important parts
- ignore proofs / low level details
- summarize

Third Pass:
- critical thinking
- trying to ”re-create” the details
- deeper understanding
- low-level details
Presenting a Paper

Tips 1
- Start after or during the reading of the paper
- What would you have asked?
- What might be unclear?
- Keep it simple (effects)
- Go deep (content)

Tips 2
- Many examples
- Examples may come before definitions
- presentation \neq handout
  - Short bullets
  - Do not include long summaries
  - Graphs, plots, illustrations
  - Demos
Preparing a Presentation

Preparing Slides

- Know the paper well
- Remember the audience
- What are the key takeaways?
- Rely on previous lectures
Structure Your Talk

**Structure**

- **Intro/Background:**
  - What is the paper about?
  - Motivation
  - Terminology and notions from previous presentations
  - Main Contribution

- **Body**
  - Main results
  - Significance
  - Methods / Tools / Techniques
  - Examples and Demos
  - Advanced material

- **Conclusion**
  - Repeat the main message
  - What was done
  - What is left to do
Presenting

Presenting Slides

- Practice
- Writing ≠ Speaking
- Time yourself
- Not too fast, not too slow
- Engage
Schedule

### Remaining Papers
- We will try to schedule now
- Notify me before until next class about your preferences, if you weren’t scheduled by the end of this class
- No preference – I assign arbitrarily

### Sanity Check
- Make sure you have access to the paper you are assigned to
- Do this early
- Preferably this week

### Tentative Schedule
- [https://u.cs.biu.ac.il/~zoharyo1/sc-seminar/index.html](https://u.cs.biu.ac.il/~zoharyo1/sc-seminar/index.html)
- Short summary of potential papers