EECS 219C: Potential Project Topics: A Discussion Starter!

Sanjit A. Seshia

EECS, UC Berkeley

Categories

- Theory / Algorithms
  - Syntax-Guided Synthesis for new Theories
  - Random Sampling for (Program) Synthesis
  - Crowdsourced Verification/Synthesis
  - Quantitative Verification
- [New] Applications
  - Robotics
  - Education
  - Networking
- Theory + Application

Random Sampling for Verification & Synthesis

- Given a SAT problem, generate solutions (nearly) uniformly at random
- Recent progress on efficient uniform/weighted sampling algorithms
  - Chakraborty et al., CAV’13, DAC’14, TACAS’15
- Can we leverage this to improve synthesis?
  - Extend to verification by synthesizing artifacts like invariants
  - Counterexample-guided inductive synthesis (CEGIS)
  - SAT formulas encode program/artifact space
  - Uniformly sample multiple programs (rather than single one)
  - Apply to bit-vector program synthesis (Jha, ICSE’10, PLDI’11)

Syntax-Guided Synthesis (SyGuS)

- Formalism (and common format) for expressing synthesis problems
  - Relies on SMT solver for some underlying logical theories
  - Currently only bit-vector and linear arithmetic SyGuS solvers exist
- Problem: Create SyGuS solver for uninterpreted functions + BV/LA
- Background: Read Alur et al., FMCAD'13

http://www.eecs.berkeley.edu/~sseshia/pubs/b2hd-alur-fmcad13.html

Grammatical Inference in SyGuS

- Syntax-guided synthesis problem requires one or more grammars from which expressions are to be synthesized
- How do we generate these grammars in the first place?
- Idea: Use techniques from grammatical inference (learning from examples)
- Background: SyGuS FMCAD'13 paper + books on Grammatical Inference (e.g., by Colin de la Higuera)
  – http://pagesperso.lina.univ-nantes.fr/~cdlh/slides/

CrowdSourced Specification/Verification/Synthesis

- Specification/Synthesis:
  - Develop methodology to crowdsource the generation of formal logical specifications/assertions for use in program verification
  - See papers: [Li, Seshia, & Jha DAC'12]; POPL'15 paper
- Education/Virtual labs:
  - Create game-based environment for students to collaboratively solve problems, or to play against each other
  - Online synthesis of algorithmic game strategy (e.g. learn from student plays)
  - See http://CPSGrader.org

Specification Mining for Signal Temporal Logic (STL)

- STL has proved a versatile logic to capture requirements on cyber-physical systems
- Previous work has had an impact on industrial practice (esp. automotive)
  - Jin et al., HSCC 2013 http://www.eecs.berkeley.edu/~sseshia/pubs/b2hd-jin-hscc13.html
  - But limited to fixed templates with variable numeric parameters
- Problem: extend to a grammar of STL formulas

Simulation-Based Auto-Grading for Circuits Courses

- Customize CPSGrader (existing Virtual Lab auto-grader) for other “EE” courses
  - E.g. Analog / Digital Circuits
- Use “Time-Frequency Logic” to specify properties
- See http://CPSGrader.org

Synthesis of Multi-Robot Motion Plans (in Adversarial Contexts)

- Initial work based on SMT solving: Saha et al., IROS 2014
  - Static environment
  - http://www.eecs.berkeley.edu/~sseshia/pubs/b2hd-saha-iros14.html
- Extend to deal with dynamic environments (adversarial agents)
  - Idea 1: Encode to SyGuS rather than SMT
  - Idea 2: Use Model-Predictive Control (Raman et al, CDC’14, HSCC’15)

Formal Methods for Networking

[w/ G. Varghese]

- Header Spaces: a major cause for state-space explosion for network verification
  - See https://www.usenix.org/system/files/conference/nsdi12/nsdi12-final8.pdf
- Sets of Packet Headers need to be represented compactly
  - What data structure to use?
  - BDDs? DDNF?
  - Compare empirically and/or theoretically

Formal Methods for Networking

[w/ G. Varghese]

- Create cyber-physical (hybrid) model of network flow control
  - E.g., See “rethinking of TCP”: http://conferences.sigcomm.org/sigcomm/2011/papers/sigcomm/p50.pdf
  - The above paper’s analysis is ad-hoc
- Need formal analysis/synthesis:
  - Prove theorem about the number of flows whose deadlines are met for deterministic models of flows
  - *Synthesize* control algorithms for packets to meet deadlines.