FMCAD paper


SyGuS-Comp
Organizers:

R. Alur, D. Fisman, R. Singh, A. Solar-Lezama
Program Synthesis

Given a specification $S$, produces a program $P$ satisfying $S$. 
New Trends in Synthesis

Emerging Synthesis Tools

Augment the specification $S$, with some syntactic restrictions $R$ on the domain of possible solutions.

Motivation:
- Tractability
- Combine human expert insights with computers exhaustiveness & rapidness
- Benefit progress SAT & SMT Solvers

Synthesizer

- Specification $S$ given by logical constraints
- Syntactic restrictions $R$ on the solutions domain

Program P
Low Level “HOW”
SAT and SMT Solvers

Satisfiability

Is formula $\varphi$, satisfiable?

- **SAT**
  Formulas use Boolean variables + Logical connectives

- **SMT** *(Satisfiability Modulo Theories)*
  Formulas are interpreted in some theory, e.g. BitVectors, Integers, Uninterpreted functions.
From Satisfiability to Synthesis

Recent trends in program synthesis:

Problem
(verification/synthesis nature)

Syntactic Restrictions on solution domain

SAT/SMT Solver
Common theme to many recent efforts

- Sketch (Bodik, Solar-Lezama et al)
- FlashFill (Gulwani et al)
- Super-optimization (Schkufza et al)
- Invariant generation (Many recent efforts...)
- TRANSIT for protocol synthesis (Udupa et al)
- Oracle-guided program synthesis (Jha et al)
- Implicit programming: Scala^Z3 (Kuncak et al)
- Auto-grader (Singh et al)

But no way to have a generic solver for all 😞
Talk Outline

- Motivation for Syntax-Guided Synthesis
- Formalization of Syntax-Guided Synthesis
- Solution Strategies
- Conclusions + SyGuS Competition
Syntax-Guided Synthesis (SyGuS) Problem

- Fix a background theory $T$: fixes types and operations
- Function to be synthesized: name $f$ along with its type
  - General case: multiple functions to be synthesized
- Inputs to SyGuS problem:
  - Specification $\varphi$
    - Typed formula using symbols in $T$ + symbol $f$
  - Context-free grammar $G$
    - Characterizing the set of allowed expressions $E$ (in theory $T$)

**Computational problem:**
Find expression $e$ in $E$ such that $\varphi[f/e]$ is valid (in theory $T$)
SyGuS - example

- Theory QF-LIA
  - Types: Integers and Booleans
  - Logical connectives, Conditionals, and Linear arithmetic
  - Quantifier-free formulas

- Function to be synthesized \( f \) (int \( x \), int \( y \)) : int

- Specification: \((x \leq f(x,y)) \& \& (y \leq f(x,y)) \& \& (f(x,y)=x \; | \; f(x,y)=y)\)

- Candidate Implementations: Linear expressions
  - \( \text{LinExp} := x \; | \; y \; | \; \text{Const} \; | \; \text{LinExp} + \text{LinExp} \; | \; \text{LinExp} - \text{LinExp} \)

- No solution exists
SyGuS - example

- Theory QF-LIA
  - Types: Integers and Booleans
  - Logical connectives, Conditionals, and Linear arithmetic
  - Quantifier-free formulas

- Function to be synthesized \( f(\text{int } x, \text{int } y) : \text{int} \)

- Specification: \((x \leq f(x,y)) \land (y \leq f(x,y)) \land (f(x,y)=x \lor f(x,y)=y)\)

- Candidate Implementations: Conditional expressions with comparisons

  \[
  \text{Term} := x \mid y \mid \text{Const} \mid \text{If-Then-Else} (\text{Cond}, \text{Term}, \text{Term})
  \]

  \[
  \text{Cond} := \text{Term} \leq \text{Term} \mid \text{Cond} \land \text{Cond} \mid \neg \text{Cond} \mid (\text{Cond})
  \]

- Possible solution:
  \(\text{If-Then-Else} (x \leq y, y, x)\)

- Common theme to many recent efforts
  - Sketch (Bodik, Solar-Lezama et al)
  - FlashFill (Gulwani et al)
  - Super-optimization (Schkufza et al)
  - Invariant generation (Many recent efforts...)
  - TRANSIT for protocol synthesis (Udupa et al)
  - Oracle-guided program synthesis (Jha et al)
  - Implicit programming: Scala^Z3 (Kuncak et al)
  - Auto-grader (Singh et al)
Motivation for Syntax-Guided Synthesis

Formalization of Syntax-Guided Synthesis

Solution Strategies

Conclusions + SyGuS Competition
SyGuS as Active Learning

Concept class: Set $E$ of expressions

Examples: Concrete input values
SyGuS Solutions

- **CEGIS** - Counter-Example Guided Inductive Synthesis approach (Solar-Lezama, Seshia et al)

- Related work: Similar strategies for solving quantified formulas and invariant generation

- Learning strategies based on:
  - **Enumerative** (search with pruning): Udupa et al (PLDI’13)
  - **Symbolic** (solving constraints): Gulwani et al (PLDI’11)
  - **Stochastic** (probabilistic walk): Schkufza et al (ASPLOS’13)
SyGuS - Recap

Program optimization

Program sketching

Programming by examples

Invariant generation
SyGuS - Recap

Program optimization
Program sketching
Programming by examples
Invariant generation

SyGuS IF
Generic Solvers
Benchmark + Compare + Compete
=> Boost improvement
SyGuS Competition

- Initiated a competition of SyGuS solvers at FLoC Olympic Games, 30 June 2014

- Competition is going to be offline, using StarExec.

- Single track, solvers should support full SyGuS-IF format for the theories of bit-vector and integer linear arithmetic.

- Caveat for "let" - Solvers do not supporting "let" will be evaluated against other such solvers

- Benchmarks:
  - Bit-manipulation programs from Hacker’s delight
  - Integer arithmetic: Find max, search in sorted array
  - Challenge problems such as computing Morton’s number
SyGuS-Comp

- **Awards:**
  - Kurt Gödel Silver Medals
  - $1000 sponsored graciously by Microsoft Research

- **Important Dates:**
  - Solver and Benchmark registration is now open
  - Solver registration deadline: 15 June 2014
  - Competition takes place: 30 June 2014
  - SyGuS session on SYNT: 23-24 July 2014
  - FLoC Award ceremony: 21 July 2014

Website: sygus.org
Thank you!

Questions?