Guarantees in Program Synthesis

Qinheping Hu, Jason Breck, John Cyphert, Loris D'Antoni, Thomas Reps
Program Synthesis is unpredictable.
Program Synthesis is Unpredictable

\[(\text{define-fun } ((x \text{ BitVec 8})) (y \text{ BitVec 8}))) (\text{bvand} (\text{bvlshr} \text{ DD } x \text{ #x02}) (\text{bvlshr} \text{ DD } y \text{ #x06})))\]
Program Synthesis is **Unpredictable**

- **Specification**
- **Search space**
- **Synthesizer**
- **Solution Program**

*Ability to prefer a solution when there are multiple solutions*
Program Synthesis is **Unpredictable**

Specification: $f(x) = x$

Search space
constant programs: 1,2,3,⋯
Program Synthesis is **Unpredictable**

- **Search-based synthesizer**
  - Solution Program?
  - timeout

- **Search space**
- **Solution space**
Guarantees in Program Synthesis

Synthesizer

- Specification
- Search space

Solution Program

Ability to prefer a solution when there are multiple solutions

Ability to answer no solutions

Make program synthesis predictable
Syntax-Guided Synthesis with Quantitative Objectives [CAV18]
Syntax-Guided Synthesis (SyGuS)

\[ \varphi(max(x, y), x, y): max(x, y) \geq x \land max(x, y) \geq y \land (max(x, y) = x \lor max(x, y) = y) \]

Specification

Search space

Synthesizer

Solution Program

\( e \in L(G) \) such that \( \forall x, y. \varphi(e, x, y) \)

Start = +(Start, Start)
| ITE(BExpr, Start, Start)
| x | y | 0 | 1

BExpr = Not(BExpr)
| > (Start, Start)
| And(BExpr, BExpr)

\[ \max(x, y) = ITE(> (x, y), x, ITE(< (x, y), y, x)) \]
A limitation of SyGuS

What we expected

(define-fun ((x (BitVec 8)) (y (BitVec 8))) (bvand (bvlsli (DD x) #x02) (bvlsr (DD y) #x06)))

Output of the CVC4 solver
<table>
<thead>
<tr>
<th>Features you want</th>
<th>Optimization Objectives</th>
</tr>
</thead>
<tbody>
<tr>
<td>Readable</td>
<td>Smallest size</td>
</tr>
<tr>
<td>Efficient</td>
<td>Least number of multiplication</td>
</tr>
<tr>
<td>Most likely</td>
<td>Largest probability</td>
</tr>
</tbody>
</table>
Guarantees in SyGuS

- Specification
- Search space
- Quantitative objectives

SyGuS solver

Solution Program
Adding Quantitative Objective to SyGuS

Start = Start + Start
  | if (BExpr) then Start else Start
  | x  | y  | 0  | 1

BExpr = Start > Start
  | not BExpr
  | BExpr and BExpr
Adding Quantitative Objective to SyGuS

Start = Start + Start
   \mid if (BExpr) then Start else Start
   \mid x \mid y \mid 0 \mid 1

BExpr = Start > Start
   \mid not BExpr
   \mid BExpr and BExpr

Quantitative objective: Minimize number of if-statement
Adding Quantitative Objective to SyGuS

Quantitative objective: Minimize number of if-statement

\[
\begin{align*}
\text{Start} &= \text{Start} + \text{Start} /0 \\
&\quad | \ if (\text{BExpr}) \ then \ \text{Start} \ else \ \text{Start} /1 \\
&\quad | \ x/0 \ | y/0 \ | 0/0 \ | 1/0 \\
\text{BExpr} &= \text{Start} > \text{Start} /0 \\
&\quad | \ not \ \text{BExpr} /0 \\
&\quad | \ \text{BExpr} \ and \ \text{BExpr} /0
\end{align*}
\]
If(x > y) then if(x > x) then 0 else x Else y

\[
\text{Weight} = 2
\]
Adding Quantitative Objective to SyGuS

Weighted grammar

\[
\text{Start} = \text{Start} + \text{Start} /0 \\
| \text{if (BExpr) then Start else Start} /1 \\
| x/0 | y/0 | 0/0 | 1/0
\]

\[
\text{BExpr} = \text{Start} > \text{Start} /0 \\
| \text{not BExpr} /0 \\
| \text{BExpr and BExpr} /0
\]

Quantitative objective: Minimize number of if-statement
Adding Quantitative Objective to SyGuS

Weighted grammar

\[
\text{Start} = \text{Start} + \text{Start} /0 \\
\quad | \text{if} (\text{BExpr}) \text{ then } \text{Start} \text{ else Start} /1 \\
\quad | x/0 \ | y/0 \ | 0/0 \ | 1/0
\]

\[
\text{BExpr} = \text{Start} > \text{Start} /0 \\
\quad | \text{not BExpr} /0 \\
\quad | \text{BExpr and BExpr} /0
\]

Quantitative objective: Minimize number of if-statement
Minimize weight

QSyGuS: (specification, weighted grammar, Quantitative objective)
Solving QSyGuS
QSyGuS

WTG $W$
Constraint $\phi$
Minimize $weight$
WTG $W$
Constraint $\phi$
Minimize weight

ignore weight

CFG $G$
Constraint $\phi$
Minimize weight

QSyGuS

WTG $W$
Constraint $\phi$

SyGuS

CFG $G$
Constraint $\phi$

Solution's weight $c_1$

CFG $G_{<c_1}$
Constraint $\phi$
QSyGuS

WTG $W$
Constraint $\phi$
Minimize weight

SyGuS

CFG $G$
Constraint $\phi$

CFG $G_{<c_1}$
Constraint $\phi$

Solution's weight $c_2$

CFG $G_{<c_2}$
Constraint $\phi$
WTG $\mathcal{W}$
Constraint $\phi$
Minimize weight $\text{weight}$
Program $P$ has weight $< 2$ iff $G_{<2}$ accept $P$
Grammar Reduction

Idea: *keep track of the weight in the non-terminals*

Start = Start + Start /0
  | if (BExpr) then Start else Start /1
  | x/0 | y/0 | 0/0 | 1/0

BExpr = Start > Start /0
  | not BExpr /0
  | BExpr and BExpr /0

weight < 2
Grammar Reduction

Idea: keep track of the weight in the non-terminals

\[
\text{Start} = \text{Start} + \text{Start} \mid 0 \\
| \text{if} (\text{BExpr}) \text{ then} \text{Start} \text{ else } \text{Start} \mid 1 \\
| x/0 | y/0 | 0/0 | 1/0
\]

\[
\text{BExpr} = \text{Start} > \text{Start} \mid 0 \\
| \text{not} \text{BExpr} \mid 0 \\
| \text{BExpr and} \text{BExpr} \mid 0
\]

weight < 2

\[
(\text{Start, < 2}) = (\text{Start, 0}) \mid (\text{Start, 1})
\]
Grammar Reduction

Idea: keep track of the weight in the non-terminals

$$\text{Start} = \text{Start} + \text{Start} / 0$$

$$| \text{if (BExpr) then Start else Start} / 1$$

$$| x/0 \ | y/0 \ | 0/0 | 1/0$$

$$\text{BExpr} = \text{Start} > \text{Start} / 0$$

$$| \text{not BExpr} / 0$$

$$| \text{BExpr and BExpr} / 0$$

weight < 2

$$(\text{Start, } < 2) = (\text{Start, } 0) \ | (\text{Start, } 1)$$

$$(\text{Start, } 1) = (\text{Start, } 0) + (\text{Start, } 1) \ | (\text{Start, } 1) + (\text{Start, } 0)$$

$$| \text{if (BExpr, 0) then (Start, 0) else (Start, 0)}$$

$$(\text{Start, } 0) = (\text{Start, } 0) + (\text{Start, } 0) \ | x \ | y \ | 0 \ | 1$$

...
Handling complex weight constraints

Tree grammars are closed under Boolean operations

Minimization \rightarrow Linear search

$3 < weight$ \rightarrow \text{Complement of } G_{<4}$

$2 < weight < 5$ \rightarrow $G_{<5} \cap G_{>2}$

$3 < weight_1 \text{ and } weight_2 < 0.5$ \rightarrow $G_{weight_1>3} \cap G_{weight_2<0.5}$
Evaluation
Evaluation

QSyGuS

WTG $W$

Constraint $\phi$

Minimize weight

SyGuS

CFG $G$

Constraint $\phi$

SyGuS solvers

CVC4

ESolver
Evaluation

26 Benchmarks taken from SyGuS
1. minimize number of specified operator, minimize solution size
2. maximize solution probability
3. find sorted optimal for (# of specified operators, size)
4. find Pareto optimal for (# of specified operators, size)

Find solution with better cost for 16/26 SyGuS benchmarks
Find optimal in 14/26 (couldn’t prove optimality for 2 benchmarks)
Average time 3.1x Compared to SyGuS
Conclusion

QSyGuS

WTG $W$

Constraint $\phi$

Minimize weight

SyGuS

CFG $G$

Constraint $\phi$

CFG $G_{<c_1}$

Constraint $\phi$

CFG $G_{<c_2}$

Constraint $\phi$

Solution's weight $C_2$

optimal

unrealizable
Guarantees in SyGuS

- Specification
- Search space

SyGuS solver

- Solution Program
- Ability to answer no solutions
- Ability to prefer a solution when there are multiple solutions
- Answering unrealizable

Quantitative objectives
Proving Unrealizability for Syntax-Guided Synthesis [CAV19]
A Syntax-Guided Synthesis (SyGuS) is

**Specification**

\[ \varphi(f(x, y), x, y) : \]
\[ f(x, y) \geq x \land f(x, y) \geq y \land \\
(f(x, y) = x \lor f(x, y) = y) \]

**Search space G:**

\[ \text{Start} = + (\text{Start, Start}) \]
\[ \mid \text{ITE} (\text{BExpr, Start, Start}) \]
\[ \mid x \mid y \mid 0 \mid 1 \]

\[ \text{BExpr} = \text{Not} (\text{BExpr}) \]
\[ \mid > (\text{Start, Start}) \]
\[ \mid \text{And} (\text{BExpr, BExpr}) \]

**Goal:** find a program \( e \in L(G) \) such that \( \forall x, y. \varphi(e, x, y) \)

\[ \max(x, y) = \text{ITE} (> (x, y), x, y) \]
Unrealizable SyGuS Problems

\[ \forall x, y. \max(x, y) \geq x \land \max(x, y) \geq y \land (\max(x, y) = x \lor \max(x, y) = y) \]

\[
\text{Start} = + (\text{Start}, \text{Start}) \\
| x | y | 0 | 1
\]

No Solution
CEGIS-based Framework

Example Set $E$

Verifier Z3

ESolver

P is a solution

SAT
new example $e$

UNSAT
CEGIS-based Framework

Example Set $E$

Reduction $R e^E$

Seahorn

Unreachable

UNREALIZABLE

ESolver

Verifier Z3

$P$

SAT

new example $e$

UNSAT

$P$ is a solution
SyGuS is unrealizable
\[ \iff \text{reachability problem } Re^E \text{ is unsatisfiable} \]
Guarantees in Program Synthesis

**Guarantees**

- Ability to prefer a solution when there are multiple solutions
- More quantitative objectives
  1. Semantic quantitative objectives
  2. Resource bounded synthesis
- Ability to answer no solutions
- Answering unrealizable
  1. Tue 12:10 come to my talk
  2. Beyond SyGuS