Semantics-Guided Synthesis

Jinwoo Kim, Qinheping Hu, Loris D’Antoni, and Thomas Reps
University of Wisconsin-Madison
This is a Paper on Program Synthesis...
This is a Paper on Program Synthesis...
Synthesize Imperative Programs

Encode

Solver 1 (e.g., Sketch)
Synthesize Imperative Programs

Encode

- Solver 1 (e.g., Sketch)
- Solver 2 (e.g., Rosette)
- Write Custom Solver
Synthesize Imperative Programs

Encode

Solver 1 (e.g., Sketch)

Solver 2 (e.g., Rosette)

Write Custom Solver
Synthesize Imperative Programs

Synthesize Regexes

Encode

Solver 1 (e.g., Sketch)

Solver 2 (e.g., Rosette)

Write Custom Solver

Specialized Solver (e.g., AlphaRegex)
Synthesize Imperative Programs

Synthesize Regexes

Synthesize Expressions

Encode

Solver 1 (e.g., Sketch)

Solver 2 (e.g., Rosette)

Write Custom Solver

Specialized Solver (e.g., AlphaRegex)

Encode as a SyGuS problem
Synthesize Imperative Programs

Synthesize Regexes

Synthesize Expressions

Encode

Solver 1
(e.g., Sketch)

Solver 2
(e.g., Rosette)

Write Custom Solver

Specialized Solver
(e.g., AlphaRegex)

Encode as a SyGuS problem
1. Semantics-Guided Synthesis
2. Solving a Semgus Problem
3. Alternative Semantics for Semgus
4. Result Highlights
This Paper:
A **language-agnostic, logic-based** framework for program synthesis problems over **arbitrary semantics**

- Synthesize Imperative Programs
- Synthesize Regexes
- Synthesize Expressions
- Encode as a Sygus problem

Solvers:
- Solver 1 (e.g., Sketch)
- Solver 2 (e.g., Rosette)
- Specialized Solver (e.g., AlphaRegex)
Synthesize Imperative Programs

Synthesize Regexes

Synthesize Expressions

Semantics-Guided Synthesis

General Semgus Solver 1 (e.g., Messy)

General Semgus Solver 2 (e.g., Messy-Enum)

Custom Semgus Solver

Domain Specific Semgus Solver
Synthesize Imperative Programs

Synthesize Regexes

Synthesize Expressions

Semantics-Guided Synthesis

General Semgus Solver 1 (e.g., Messy)

General Semgus Solver 2 (e.g., Messy-Enum)

Custom Semgus Solver

Domain Specific Semgus Solver
Synthesize Imperative Programs

Synthesize Regexes

Synthesize Expressions

Semantics-Guided Synthesis

General Semgus Solver 1 (e.g., Messy)

General Semgus Solver 2 (e.g., Messy-Enum)

Custom Semgus Solver

Domain Specific Semgus Solver
Synthesize Imperative Programs

Synthesize Regexes

Synthesize Expressions

Semantics-Guided Synthesis

General Semgus Solver 1 (e.g., Messy)

General Semgus Solver 2 (e.g., Messy-Enum)

Custom Semgus Solver

Domain Specific Semgus Solver
Semantics-Guided Synthesis Problem

Language Syntax
- Regular Tree Grammar

Language Semantics
- Constrained Horn Clauses

Problem Specification
- Constraints / Examples
Semantics-Guided Synthesis Problem

Language Syntax
- Regular Tree Grammar

Language Semantics
- Constrained Horn Clauses

Problem Specification
- Constraints / Examples
Semantics-Guided Synthesis Problem

Language Syntax
- Regular Tree Grammar

Language Semantics
- Constrained Horn Clauses

Problem Specification
- Constraints / Examples

\[ E ::= x \mid y \mid \text{If } B \text{ then } E \text{ else } E \]
\[ B ::= E < E \mid B \land B \]
Semantics-Guided Synthesis Problem

Language Syntax
- Regular Tree Grammar

Language Semantics
- Constrained Horn Clauses

Problem Specification
- Constraints / Examples

\[ R ::= \epsilon \mid \Sigma \mid R \cdot R \mid (R \mid R) \mid R^* \]
Semantics-Guided Synthesis Problem

Language Syntax
- Regular Tree Grammar

Language Semantics
- Constrained Horn Clauses

Problem Specification
- Constraints / Examples

\[ \text{Start ::= While } B \text{ do } S \]
\[ B ::= E < E \]
\[ E ::= x \mid y \mid E \& E \mid (E \mid E) \]
\[ S ::= S; S \mid x := E \mid y := E \]
Semantics-Guided Synthesis Problem

Language Syntax
- Regular Tree Grammar

Language Semantics
- Constrained Horn Clauses

Problem Specification
- Constraints / Examples

\[ \begin{align*}
\text{Start} & ::= \text{While } B \text{ do } S \\
B & ::= \ E < \ E \\
E & ::= x | y | E \& E | (E | E) \\
S & ::= S; S | x := E | y := E
\end{align*} \]
Semantics-Guided Synthesis Problem

**Language Syntax**
- Regular Tree Grammar

**Language Semantics**
- Constrained Horn Clauses

**Problem Specification**
- Constraints / Examples

**Start ::= While B do S**
**B ::= E < E**
**E ::= x | y | E & E | (E | E)**
**S ::= S; S | x := E | y := E**
Start ::= While B do S
B ::= E < E
E ::= x | y | E & E | (E | E)
S ::= S; S | x := E | y := E
\[\text{Start} ::= \text{While } B \text{ do } S\]
\[B ::= E < E\]
\[E ::= x \mid y \mid E \& E \mid (E \mid E)\]
\[S ::= S; S \mid x := E \mid y := E\]

\text{S: Every term in S has type } State \rightarrow State
Start ::= While B do S
B ::= E < E
E ::= x | y | E & E | (E | E)
\[ S ::= S; S | x := E | y := E \]

$S$: Every term in S has type $State \rightarrow State$

$\llbracket x := x \& y \rrbracket(1,0) = (0,0)$
Start ::= While B do S
B ::= E < E
E ::= x | y | E & E | (E | E)
S ::= S ; S | x := E | y := E

$S: \text{Every term in } S \text{ has type } State \rightarrow State$

$[ x := x \& y ](1, 0) = (0, 0)$

$Sem_S: State \times Term \times State \rightarrow Boolean$
Relation $Sem_S$: Captures the semantics of all terms $t_S \in L(S)$
\[
\text{Start ::= While } B \text{ do } S \\
B ::= E < E \\
E ::= x \mid y \mid E \& E \mid (E \mid E) \\
S ::= S; S \mid x ::= E \mid y ::= E \\
\]

\[S: \text{ Every term in } S \text{ has type } \text{State} \rightarrow \text{State}\]

\[\left[ x ::= x \& y \right]_{(1, 0)} = (0, 0)\]

\[\text{Sem}_S((1, 0), x ::= x \& y, (0, 0)) = \text{True}\]

\[\text{Sem}_S: \text{State} \times \text{Term} \times \text{State} \rightarrow \text{Boolean}\]

Relation \(\text{Sem}_S\): Captures the semantics of all terms \(t_S \in L(S)\)
Start ::= While B do S
B ::= E < E
E ::= x | y | E & E | (E | E)
S ::= S; S | x := E | y := E

S: Every term in S has type State → State

SemS: State × Term × State → Boolean
Relation SemS: Captures the semantics of all terms tS ∈ L(S)

SemS(Γ, tS, Γ′) ⇔ [ tS ](Γ) = Γ′
tS returns Γ′ when executed on Γ
\[ S ::= x := E \]
$S ::= x := E$

$\left[ t_e \right](\Gamma) = v \quad \Gamma_1 = \Gamma[x \mapsto v]$

$\left[ x := t_e \right](\Gamma) = \Gamma_1$
\[ S ::= x := E \]

\[
\frac{[t_e](\Gamma) = v \quad \Gamma_1 = \Gamma[x \mapsto v]}{[x := t_e](\Gamma) = \Gamma_1}
\]

\[
\frac{Sem_E(\Gamma, t_e, v) \quad \Gamma_1 = \Gamma[x \mapsto v]}{Sem_S(\Gamma, x := t_e, \Gamma_1)}
\]
\[ S ::= x := E \]

\[
\begin{align*}
\llbracket t_e \rrbracket(\Gamma) &= v & \Gamma_1 &= \Gamma[x \mapsto v] \\
\llbracket x := t_e \rrbracket(\Gamma) &= \Gamma_1
\end{align*}
\]

\[
Sem_E(\Gamma, t_e, v) & \quad \Gamma_1 = \Gamma[x \mapsto v] \\
Sem_S(\Gamma, x := t_e, \Gamma_1)
\]

\[ \Updownarrow \]

\[ \forall \Gamma, \Gamma_1, t_e, v. \ Sem_E(\Gamma, t_e, v) \land \Gamma_1 = \Gamma[x \mapsto v] \Rightarrow Sem_S(\Gamma, x := t_e, \Gamma_1) \]
\[
S ::= x := E
\]

\[
\begin{align*}
&\llbracket t_e \rrbracket(\Gamma) = v \quad \Gamma_1 = \Gamma[x \mapsto v] \\
&\llbracket x := t_e \rrbracket(\Gamma) = \Gamma_1
\end{align*}
\]

\[
Sem_E(\Gamma, t_e, v) \quad \Gamma_1 = \Gamma[x \mapsto v] \\
Sem_S(\Gamma, x := t_e, \Gamma_1)
\]

\[
\forall \Gamma, \Gamma_1, t_e, v. \ Sem_E(\Gamma, t_e, v) \land \Gamma_1 = \Gamma[x \mapsto v] \Rightarrow Sem_S(\Gamma, x := t_e, \Gamma_1)
\]
\[ S ::= x := E \]

\[
\begin{align*}
\llbracket t_e \rrbracket(\Gamma) &= v \quad \Gamma_1 = \Gamma[x \mapsto v] \\
\llbracket x := t_e \rrbracket(\Gamma) &= \Gamma_1
\end{align*}
\]

\[
Sem_E(\Gamma, t_e, v) \quad \Gamma_1 = \Gamma[x \mapsto v] \\
Sem_S(\Gamma, x := t_e, \Gamma_1)
\]

\[
\forall \Gamma, \Gamma_1, t_e, v. \quad Sem_E(\Gamma, t_e, v) \land \Gamma_1 = \Gamma[x \mapsto v] \Rightarrow Sem_S(\Gamma, x := t_e, \Gamma_1)
\]
\[
\text{Start ::= While } B \text{ do } S
\]

\[
\frac{\text{Sem}_B(\Gamma, t_b, v) \quad v = \text{True}}{\text{Sem}_S(\Gamma, t_s, \Gamma_1)} \quad \frac{\text{Sem}_S(\Gamma_1, \text{While } t_b \text{ do } t_s, \Gamma_2)}{\text{Sem}_\text{Start}(\Gamma_1, \text{While } t_b \text{ do } t_s, \Gamma_2)}
\]

\[
\frac{\text{Sem}_B(\Gamma, t_b, v) \quad v = \text{False}}{\text{Sem}_\text{Start}(\Gamma, \text{While } t_b \text{ do } t_s, \Gamma)}
\]
Start ::= While B do S

\[\begin{align*}
\text{Sem}_B(\Gamma, t_b, v) & \quad v = True \quad \text{Sem}_S(\Gamma, t_s, \Gamma_1) \quad \text{Sem}_{\text{Start}}(\Gamma_1, \text{While } t_b \text{ do } t_s, \Gamma_2) \\
\text{Sem}_{\text{Start}}(\Gamma, \text{While } t_b \text{ do } t_s, \Gamma_2)
\end{align*}\]

\[\begin{align*}
\text{Sem}_B(\Gamma, t_b, v) & \quad v = False \\
\text{Sem}_{\text{Start}}(\Gamma, \text{While } t_b \text{ do } t_s, \Gamma)
\end{align*}\]
Start ::= While B do S

\[
\begin{align*}
Sem_B(\Gamma, t_b, v) & \quad v = True \quad Sem_S(\Gamma, t_s, \Gamma_1) \quad Sem_{Start}(\Gamma_1, While t_b \text{ do } t_s, \Gamma_2) \\
Sem_{Start}(\Gamma, While t_b \text{ do } t_s, \Gamma_2)
\end{align*}
\]

\[
\begin{align*}
Sem_B(\Gamma, t_b, v) & \quad v = False \\
Sem_{Start}(\Gamma, While t_b \text{ do } t_s, \Gamma)
\end{align*}
\]
Semantics-Guided Synthesis Problem

Language Syntax
- Regular Tree Grammar

Language Semantics
- Constrained Horn Clauses

Problem Specification
- Constraints / Examples

Start ::= While B do S
B ::= E < E
E ::= x | y | E & E | (E | E)
S ::= S; S | x := E | y := E

\[ \text{Sem}_E(\Gamma, t_e, v) \Gamma_1 = \Gamma[x \mapsto v] \]
\[ \text{Sem}_S(\Gamma, x := t_e, \Gamma_1) \]
Semantics-Guided Synthesis Problem

Language Syntax
- Regular Tree Grammar

Language Semantics
- Constrained Horn Clauses

Problem Specification
- Constraints / Examples

```
Start ::= While B do S
B ::= E < E
E ::= x | y | E & E | (E | E)
S ::= S; S | x := E | y := E

\[
\begin{align*}
Sem_E(\Gamma, t_e, v) \Gamma_1 &= \Gamma[x \mapsto v] \\
Sem_S(\Gamma, x := t_e, \Gamma_1)
\end{align*}
\]
```
Semantics-Guided Synthesis Problem

<table>
<thead>
<tr>
<th>Language Syntax</th>
<th>Language Semantics</th>
<th>Problem Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regular Tree Grammar</td>
<td>Constrained Horn Clauses</td>
<td>Constraints / Examples</td>
</tr>
</tbody>
</table>

Logical formula: \( f(x, y) = x \oplus y \)
Semantics-Guided Synthesis Problem

Language Syntax
- Regular Tree Grammar

Language Semantics
- Constrained Horn Clauses

Problem Specification
- Constraints / Examples

Examples: \[ f(6, 9) = (15, \_\) \]
Semantics-Guided Synthesis Problem

Language Syntax
- Regular Tree Grammar

Language Semantics
- Constrained Horn Clauses

Problem Specification
- Constraints / Examples

Examples: \[ f(6, 9) = (15, _) \]

CEGIS: We can always generate more counterexamples!
Semantics-Guided Synthesis Problem

Language Syntax
- Regular Tree Grammar

Language Semantics
- Constrained Horn Clauses

Problem Specification
- Constraints / Examples

**Start ::= While B do S**

\[
\begin{align*}
B & ::= E < E \\
E & ::= x \mid y \mid E \& E \mid (E \mid E) \\
S & ::= S; S \mid x ::= E \mid y ::= E
\end{align*}
\]

**Examples:**
- \( f(6, 9) = (15, _) \)
1. Semantics-Guided Synthesis
2. Solving a Semgus Problem
3. Alternative Semantics for Semgus
4. Result Highlights
Semantics-Guided Synthesis Problem

### Language Syntax
- **Regular Tree Grammar**

### Language Semantics
- **Constrained Horn Clauses**

### Problem Specification
- **Constraints / Examples**

#### Start ::= While B do S
#### B ::= E < E
#### E ::= x | y | E & E | (E | E)
#### S ::= S; S | x := E | y := E

\[
\text{Sem}_E (\Gamma, t_e, v) \Gamma_1 = \Gamma [x \mapsto v]
\]

\[
\text{Sem}_S (\Gamma, x := t_e, \Gamma_1)
\]

#### Examples:
\[
f(6, 9) = (15, \_)
\]
Semantics-Guided Synthesis Problem

Language Syntax
- Regular Tree Grammar

Language Semantics
- Constrained Horn Clauses

Problem Specification
- Constraints / Examples

\[ \text{Sem}_{S}(\Gamma, x := t_{e}, \Gamma_{1}) \]

\[ \text{Sem}_{E}(\Gamma, t_{e}, v) \Gamma_{1} = \Gamma[x \mapsto v] \]

**Examples:**
\[ f(6, 9) = (15, _) \]

**Realizable**

\[ t \in L(\text{Start}) \quad \text{Sem}_{\text{Start}}((6, 9), t, (15, _)) \]

Solving a Semgus problem ⇒ Solving a query over CHCs!
Semantics-Guided Synthesis Problem

Language Syntax
- Regular Tree Grammar

Language Semantics
- Constrained Horn Clauses

Problem Specification
- Constraints / Examples

---

\[
\begin{align*}
\text{Start} & ::= \text{While } B \text{ do } S \\
B & ::= E < E \\
E & ::= x \mid y \mid E \& E \mid (E \mid E) \\
S & ::= S; S \mid x := E \mid y := E \\
\end{align*}
\]

\[
\begin{align*}
\text{Sem}_E(\Gamma, t_e, v) \Gamma_1 &= \Gamma[x \mapsto v] \\
\text{Sem}_S(\Gamma, x := t_e, \Gamma_1)
\end{align*}
\]

Examples:
- \(f(6, 9) = (15, \_)

\[
t \in L(\text{Start}) \quad \boxed{\text{Sem}_{\text{Start}}(6, 9, t, (15, \_))}
\]

Realizable

Solving a Semgus problem \(\Rightarrow\) Solving a query over CHCs!
Semantics-Guided Synthesis Problem

Language Syntax
- Regular Tree Grammar

Language Semantics
- Constrained Horn Clauses

Problem Specification
- Constraints / Examples

\[ \text{Start} ::= \text{While } B \text{ do } S \]
\[ B ::= E < E \]
\[ E ::= x \mid y \mid E \& E \mid (E \mid E) \]
\[ S ::= S; S \mid x := E \mid y := E \]

\[ \text{Sem}_E(\Gamma, t_e, v) \Gamma_1 = \Gamma[x \mapsto v] \]
\[ \text{Sem}_S(\Gamma, x := t_e, \Gamma_1) \]

Examples:
- \( f(6, 9) = (15, \_\) \)

\[ \text{Syn}_{\text{Start}}(t) \quad \text{Sem}_{\text{Start}}((6, 9), t, (15, \_)) \]

Realizable

Solving a Semgus problem ⇒ Solving a query over CHCs!
\[ S ::= S; S \]
\[ S ::= S; S \]

Given \( t_1 \in L(S) \) and \( t_2 \in L(S) \): 
\( t_1; t_2 \in L(S) \)
Given $t_1 \in L(S)$ and $t_2 \in L(S)$:
$t_1; t_2 \in L(S)$

$S ::= S; S$

$Syn_S: Term \rightarrow Boolean$
$Syn_S(t) \iff t \in L(S)$
Given $t_1 \in L(S)$ and $t_2 \in L(S)$:

$t_1; t_2 \in L(S)$

\[
\begin{array}{c}
\text{Syn}_S(t_1) \quad \text{Syn}_S(t_2) \\
\hline
\text{Syn}_S(t_1; t_2)
\end{array}
\]
Syntax:
Start ::= While B do S
B ::= E < E
E ::= x | y | E & E | (E | E)
S ::= S; S | x := E | y := E

Semantics:
\[ \text{Sem}_E(\Gamma, t_e, v) \quad \Gamma_1 = \Gamma[x \mapsto v] \]
\[ \text{Sem}_S(\Gamma, x := t_e, \Gamma_1) \]
\[ \ldots \]

\[ \text{Syn}_{\text{Start}}(t) \quad \text{Sem}_{\text{Start}}((6,9), t, (15, _)) \]
Realizable

Specification: \[ f(6,9) = (15, _) \]
Syntax:
Start ::= While B do S
B ::= E < E
E ::= x | y | E & E | (E | E)
S ::= S; S | x := E | y := E

Semantics:
Sem_E(Γ, t, v)  Γ_1 = Γ[x ↦ v]
Sem_S(Γ, x := t, Γ_1)  ...

Syn_{Start}(t)  Sem_{Start}((6, 9), t, (15, _))
Realizable

Specification: [f(6, 9) = (15, _)]
Syntax:
\[
\begin{align*}
\text{Start} &::= \text{While } B \text{ do } S \\
B &::= E < E \\
E &::= x \mid y \mid E \& E \mid (E \mid E) \\
S &::= S; S \mid x := E \mid y := E
\end{align*}
\]

Semantics:
\[
\begin{align*}
\text{Sem}_{E}(\Gamma, t_{e}, v) &\quad \Gamma_{1} = \Gamma[x \mapsto v] \\
\text{Sem}_{S}(\Gamma, x := t_{e}, \Gamma_{1}) &\quad \ldots
\end{align*}
\]

\[
\begin{align*}
\text{Syn}_{\text{Start}}(t) &\quad \text{Sem}_{\text{Start}}(6, 9, t, 15, \_)
\end{align*}
\]

Realizable

Specification: \([f(6, 9) = (15, \_)]\)
Program Synthesis: Proving \textbf{Realizable}

\begin{align*}
\text{Syn}_{\text{Start}}(t) & \quad \text{Sem}_{\text{Start}}((6, 9), t, (15, \_)) \\
& \quad \text{Realizable}
\end{align*}

Specification: $[f(6, 9) = (15, \_)]$

Valid solution $t = \text{While } x < y \text{ do } x := x \mid y$
Program Synthesis: Proving \textit{Realizable}

\[ Syn_{Start}(t) \quad Sem_{Start}((6,9), t, (15, \_)) \]
\[ \text{Realizable} \]

Specification: \([f(6, 9) = (15, \_)]\]

Valid solution \( t = \text{While } x < y \text{ do } x := x \mid y \)
Program Synthesis: Proving *Realizable*

\[ \text{Syn}_B(x < y) \quad \text{Syn}_S(x := x \mid y) \]

\[ \text{Syn}_{\text{Start}}(t) \]

\[ \text{Syn}_{\text{Start}}(t) \quad \text{Sem}_{\text{Start}}((6, 9), t, (15, \_)) \]

\[ \text{Realizable} \]

**Specification:** \[ f(6, 9) = (15, \_)]

**Valid solution** \[ t = \text{While } x < y \text{ do } x := x \mid y \]
Program Synthesis: Proving *Realizable*

\[
\begin{align*}
\text{Syn}_B(x < y) & \quad \text{Syn}_S(x := x \mid y) \\
\text{Syn}_{\text{Start}}(t) & \quad \frac{\text{Syn}_B(b) \quad \text{Syn}_S(s)}{\text{Syn}_{\text{Start}}(\text{While } b \text{ do } s)} \\
\text{Syn}_{\text{Start}}(t) & \quad \text{Sem}_{\text{Start}}((6, 9), t, (15, \_)) \\
\text{Realizable} & \quad \text{Specification: } [f(6, 9) = (15, \_)] \\
\text{Valid solution } t = \text{While } x < y \text{ do } x := x \mid y
\end{align*}
\]
Program Synthesis: Proving \textit{Realizable}

\[
\begin{align*}
\frac{\text{Syn}_E(x) \quad \text{Syn}_E(y)}{	ext{Syn}_B(x < y)} \quad \frac{\text{Syn}_S(x := x \mid y)}{	ext{Syn}_B(E < E)} \\
\text{Syn}_{\text{Start}}(t) \\
\frac{\text{Syn}_{\text{Start}}(t) \quad \text{Sem}_{\text{Start}}((6, 9), t, (15, _))}{\text{Realizable}}
\end{align*}
\]

Specification: \([f(6, 9) = (15, _)]\)
Valid solution \(t = \text{While } x < y \text{ do } x := x \mid y\)
Program Synthesis: Proving *Realizable*

\[
\begin{array}{c}
\text{Syn}_E(x) & \text{Syn}_E(y) \\
\Rightarrow & \\
\text{Syn}_B(x < y) \\
\end{array}
\]

\[
\begin{array}{c}
\text{Syn}_E(x) & \text{Syn}_E(y) \\
\Rightarrow & \\
\text{Syn}_E(x \mid y) \\
\Rightarrow & \\
\text{Syn}_S(x := x \mid y) \\
\end{array}
\]

\[
\text{Syn}_{\text{Start}}(t)
\]

\[
\begin{array}{c}
\text{Syn}_{\text{Start}}(t) & \text{Sem}_{\text{Start}}((6, 9), t, (15, _)) \\
\Rightarrow & \\
\text{Realizable} \\
\end{array}
\]

**Specification:** \([f(6, 9) = (15, _)]\)

**Valid solution** \(t = \text{While } x < y \text{ do } x := x \mid y\)
Program Synthesis: Proving *Realizable*

\[
\begin{align*}
\text{Syn}_{\text{Start}}(t) & \quad \text{Sem}_{\text{Start}}((6, 9), t, (15, \_)) \\
\uparrow & \\
\text{Realizable} & \\
\end{align*}
\]

Specification: \([f(6, 9) = (15, \_)]\)

Valid solution \(t = \text{While } x < y \text{ do } x := x \mid y\)
Program Synthesis: Proving *Realizable*

\[
\begin{align*}
\text{Sem}_B(\Gamma, t_b, v) & \quad v = \text{True} \quad \text{Sem}_S(\Gamma, t_s, \Gamma_1) \quad \text{Sem}_{\text{Start}}(\Gamma_1, \text{While } t_b \text{ do } t_s, \Gamma_2) \\
\text{Sem}_{\text{Start}}(\Gamma, \text{While } t_b \text{ do } t_s, \Gamma_2) \\
\end{align*}
\]

\[
\begin{align*}
\text{Sem}_B((6,9), x < y, \text{True}) & \quad \text{Sem}_S((6,9), x := x \mid y, (15,9)) \quad \cdots \\
\text{Sem}_{\text{Start}}((6,9), t, (15,\_)) \\
\end{align*}
\]

\[
\begin{align*}
\text{Syn}_{\text{Start}}(t) & \quad \text{Sem}_{\text{Start}}((6,9), t, (15,\_)) \\
\text{Realizable} \\
\end{align*}
\]

Specification: \([f(6,9) = (15,\_)]\)

Valid solution \(t = \text{While } x < y \text{ do } x := x \mid y\)
Program Synthesis: Proving *Realizable*

\[
\begin{align*}
\text{Sem}_E((6,9), x, 6) & & \text{Sem}_E((6,9), y, 9) & & \ldots & & \text{Sem}_S((6,9), x := x \mid y, (15,9)) & & \ldots \\
\text{Sem}_B((6,9), x < y, \text{True}) & & & & & & \text{Sem}_\text{Start}((6,9), t, (15,\_)) \\
\text{Syn}_\text{Start}(t) & & \text{Sem}_\text{Start}((6,9), t, (15,\_)) & & \text{Realizable} \\
\end{align*}
\]

Specification: \([f(6, 9) = (15,\_)]\)

Valid solution \(t = \text{While } x < y \text{ do } x := x \mid y\)
Program Synthesis: Proving *Realizable*

Proof tree contains term $t$: Program synthesized!

$$
\begin{align*}
\text{Sem}_E((6, 9), x, 6) & \quad \text{Sem}_E((6, 9), y, 9) \\
\text{Sem}_B((6, 9), x < y, \text{True}) & \quad \text{Sem}_S((6, 9), x := x | y, (15, 9)) \\
\text{Sem}_{\text{Start}}((6, 9), t, (15, \_)) & \\
\text{Syn}_{\text{Start}}(t) & \quad \text{Sem}_{\text{Start}}((6, 9), t, (15, \_)) \\
\text{Realizable} & \\
\end{align*}
$$

Specification: $[f(6, 9) = (15, \_)]$

Valid solution $t = \text{While } x < y \text{ do } x := x | y$
1. Semantics-Guided Synthesis
2. Solving a Semgus Problem
3. Alternative Semantics for Semgus
4. Result Highlights
Semantics-Guided Synthesis Problem

Language Syntax
- Regular Tree Grammar

Language Semantics
- Constrained Horn Clauses

Problem Specification
- Constraints / Examples

Standard Semantics
Semantics-Guided Synthesis Problem

Language Syntax
- Regular Tree Grammar

Language Semantics
- Constrained Horn Clauses

Problem Specification
- Constraints / Examples

- Standard Semantics
- Abstract Semantics
- Bounded Loops
- Define Custom Ops
Semantics-Guided Synthesis Problem

Language Syntax
- Regular Tree Grammar

Language Semantics
- Constrained Horn Clauses

Problem Specification
- Constraints / Examples

Standard Semantics
- Abstract Semantics
- Bounded Loops
- Define Custom Ops

Wang, et al. POPL17
\[ [x](\Gamma) = v_x \quad [y](\Gamma) = v_y \quad v = v_x \& v_y \]
\[ [x\&y](\Gamma) = v \]
\[ \llbracket x \rrbracket(\Gamma) = v_x \quad \llbracket y \rrbracket(\Gamma) = v_y \quad v = v_x \& v_y \]

\[ \llbracket x \& y \rrbracket(\Gamma) = v \]

Abstract Domain: Consider only first bit
Abstract Domain: Consider only first bit

\[
\llbracket x \rrbracket(\Gamma) = b_x \quad \llbracket y \rrbracket(\Gamma) = b_y \quad b = \text{if } (b_x = T \lor b_y = T) \text{ T else } b_x \lor b_y \\
\llbracket x \& y \rrbracket(\Gamma) = b
\]
\[
\boxed{
\begin{align*}
\llbracket x \rrbracket (\Gamma) &= v_x & \llbracket y \rrbracket (\Gamma) &= v_y & v = v_x \land v_y \\
\llbracket x \& y \rrbracket (\Gamma) &= v
\end{align*}
}\]

Abstract Domain: Consider only first bit

\[
\boxed{
\begin{align*}
Sem_E(\Gamma, x, b_x) & \quad Sem_E(\Gamma, y, b_y) \\
& \quad \quad b = \text{if } (b_x = \top \lor b_y = \top) \top \text{ else } b_x \land b_y \\
& \quad \quad \quad Sem_S(\Gamma, x \& y, b)
\end{align*}
}\]
\[ [x](\Gamma) = v_x \quad [y](\Gamma) = v_y \quad v = v_x \land v_y \]
\[ [x \& y](\Gamma) = v \]

Problem \( S \) has no solution using abstract semantics \( \Rightarrow \)
\( S \) has no solution using standard semantics

\[ \text{Sem}_E(\Gamma, x, b_x) \quad \text{Sem}_E(\Gamma, y, b_y) \quad b = \text{if } (b_x = \top \lor b_y = \top) \top \text{ else } b_x \land b_y \]
\[ \text{Sem}_S(\Gamma, x \& y, b) \]
Semantics-Guided Synthesis Problem

Language Syntax
- Regular Tree Grammar

Language Semantics
- Constrained Horn Clauses

Problem Specification
- Constraints / Examples

Standard Semantics

Abstract Semantics

Bounded Loops

Define Custom Ops

Sketch, Rosette
While $b$ do $s$

\[
\begin{align*}
\ll b \gg(\Gamma) &= True \\
\ll s \gg(\Gamma) &= \Gamma_1 \\
\ll While \ b \ do \ s \gg(\Gamma_1) &= \Gamma_2 \\
\ll While \ b \ do \ s \gg(\Gamma) &= \Gamma_2
\end{align*}
\]
While $b$ do $s$

\[
\begin{align*}
\llbracket b \rrbracket(\Gamma) &= \text{True} & \llbracket s \rrbracket(\Gamma) &= \Gamma_1 & \llbracket \text{While } b \text{ do } s \rrbracket(\Gamma) &= \Gamma_2 \\
\llbracket \text{While } b \text{ do } s \rrbracket(\Gamma) &= \Gamma_2
\end{align*}
\]

Bound Loops to $k$ iterations
\(\text{While } b \text{ do } s\)

\[
\begin{align*}
\ll b \gg (\Gamma) &= True \\
\ll s \gg (\Gamma) &= \Gamma_1 \\
\ll \text{While } b \text{ do } s \gg (\Gamma_1) &= \Gamma_2 \\
\ll \text{While } b \text{ do } s \gg (\Gamma) &= \Gamma_2
\end{align*}
\]

Bound Loops to \(k\) iterations

\[
\begin{align*}
\ll b \gg (\Gamma) &= True \\
k &> 0 \\
\ll s \gg (\Gamma) &= \Gamma_1 \\
\ll \text{While } b \text{ do } s \gg (\Gamma_1, k - 1) &= \Gamma_2 \\
\ll \text{While } b \text{ do } s \gg (\Gamma, k) &= \Gamma_2
\end{align*}
\]
While \( b \) do \( s \)

\[
\begin{align*}
\ll b \gg (\Gamma) &= \text{True} \\
\ll s \gg (\Gamma) &= \Gamma_1 \\
\ll \text{While } b \text{ do } s \gg (\Gamma_1) &= \Gamma_2 \\
\ll \text{While } b \text{ do } s \gg (\Gamma) &= \Gamma_2
\end{align*}
\]

Bound Loops to \( k \) iterations

\[
\begin{align*}
\text{Sem}_B(\Gamma, b, \text{True}) & \quad k > 0 \quad \text{Sem}_S(\Gamma, s, \Gamma_1) \quad \text{Sem}_S(\Gamma_1, \text{While } b \text{ do } s, \Gamma_2, k - 1) \\
& \quad \text{Sem}_S(\Gamma, \text{While } b \text{ do } s, \Gamma_2, k)
\end{align*}
\]
While $b$ do $s$

\[
\begin{align*}
\ll b \rr(\Gamma) &= \text{True} \\
\ll s \rr(\Gamma) &= \Gamma_1 \\
\ll \text{While } b \text{ do } s \rr(\Gamma_1) &= \Gamma_2
\end{align*}
\]

\[
\ll \text{While } b \text{ do } s \rr(\Gamma) = \Gamma_2
\]

Term $t$ is a solution using bounded loops $\Rightarrow$

$t$ is a solution using standard semantics

\[
\begin{align*}
\text{Sem}_B(\Gamma, b, \text{True} ) & \quad k > 0 & \quad \text{Sem}_S(\Gamma, s, \Gamma_1) & \quad \text{Sem}_S(\Gamma_1, \text{While } b \text{ do } s, \Gamma_2, k - 1) \\
\text{Sem}_S(\Gamma, \text{While } b \text{ do } s, \Gamma_2, k)
\end{align*}
\]
Semantics-Guided Synthesis Problem

Language Syntax
- Regular Tree Grammar

Language Semantics
- Constrained Horn Clauses

Problem Specification
- Constraints / Examples

Standard Semantics
- Bounded Loops
- Define Custom Ops
- Abstract Semantics
1. Semantics-Guided Synthesis
2. Solving a Semgus Problem
3. Alternative Semantics for Semgus
4. Result Highlights
The Architecture of Messy

Semgus Problem

Messy
The Architecture of Messy

Semgus Problem → Set of CHCs → Z3
The Architecture of Messy

SAT: Program Synthesized

Term can be recovered from proof
The Architecture of Messy

Semgus Problem -> Set of CHCs -> Z3

SAT: Program Synthesized

UNSAT: Unrealizable Problem

Unrealizability:
When a synthesis problem has no solution within the given syntax

Term can be recovered from proof
Benchmarks

60 LIA SyGuS Benchmarks
  • From SyGuS-Comp

132 Unrealizable LIA Benchmarks
  • From Hu, et al. CAV2019

67 Realizable Imperative Benchmarks
  • Some written by hand, some from So, et al. SAS2017

222 Unrealizable Imperative Benchmarks
  • Handwritten variants of the 67 realizable benchmarks
Realizable Benchmarks (60 Expression / 67 Imperative)

Standard Semantics

3 Solved
[0 Exp. / 3 Impv.]
Realizable Benchmarks (60 Expression / 67 Imperative)

- 3 Solved
  - [0 Exp. / 3 Impv.]

- Standard Semantics

- Bounded Loops
  - 3 Solved
  - [0 Exp. / 3 Impv.]
Realizable Benchmarks (60 Expression / 67 Imperative)

6 Solved
[0 Exp. / 6 Impv.]

Bounded Loops
Realizable Benchmarks (60 Expression / 67 Imperative)

12 Solved
[4 Exp. / 8 Impv.]

Messy w/ all Semantics
Unrealizable Benchmarks (132 Expression / 222 Imperative)

Standard Semantics

66 Solved
[56 Exp. / 10 Impv.]
Unrealizable Benchmarks (132 Expression / 222 Imperative)

Standard Semantics

Abstract Semantics

55 Solved
[18 Exp. / 37 Impv.]
Unrealizable Benchmarks (132 Expression / 222 Imperative)

- **Standard Semantics**
  - 38 Solved
  - [28 Exp. / 0 Impv.]

- **Abstract Semantics**
  - 28 Solved
  - [18 Exp. / 10 Impv.]
  - 27 Solved
  - [0 Exp. / 27 Impv.]
Unrealizable Benchmarks (132 Expression / 222 Imperative)

Messy w/ all Semantics

178 Solved [66 Expr. / 112 Impv.]
Different semantics can solve different synthesis problems!
Semantics-Guided Synthesis Problem

Language Syntax
- Regular Tree Grammar

Language Semantics
- Constrained Horn Clauses

Problem Specification
- Constraints / Examples

\[
\begin{align*}
\text{Start} &::= \text{While } B \text{ do } S \\
B &::= E < E \\
E &::= x \mid y \mid E \& E \mid (E \mid E) \\
S &::= S ; S \mid x := E \mid y := E
\end{align*}
\]

\[
\begin{align*}
Sem_E(\Gamma, t_e, v) \Gamma_1 &= \Gamma[x \mapsto v] \\
Sem_S(\Gamma, x := t_e, \Gamma_1)
\end{align*}
\]

Examples:
- \(f(6, 9) = (15, \_)
- \(f(14, 15) = (1, \_)

One more example!

\[
\begin{align*}
\text{Syn}_{\text{Start}}(t) & \quad \text{Sem}_{\text{Start}}((6, 9), t, (15, y_1)) \\
& \quad \text{Sem}_{\text{Start}}((14, 15), t, (1, \_)) \\
\text{Realizable}
\end{align*}
\]
\[ \text{Syn}_{\text{Start}}(t) \quad \text{Sem}_{\text{Start}}((6, 9), t, (15, _)) \quad \text{Sem}_{\text{Start}}((14, 15), t, (1, _)) \]

Realizable

Specification: \([f(6, 9) = (15, _), f(14, 15) = (1, _)]\)
Syntax:

\[ \text{Start} ::= \text{While } B \text{ do } S \]
\[ B ::= E < E \]
\[ E ::= x \mid y \mid E \& E \mid (E \mid E) \]
\[ S ::= S; S \mid x := E \mid y := E \]

\[
\begin{align*}
\text{Syn}_{\text{Start}}(t) \quad & \text{Sem}_{\text{Start}}((6,9), t, (15, _)) \quad \text{Sem}_{\text{Start}}((14,15), t, (1, _)) \\
\text{Realizable} & \quad \text{Specification: } [f(6, 9) = (15, _), f(14, 15) = (1, _)]
\end{align*}
\]
Syntax:
\[\text{Start} ::= \text{While } B \text{ do } S\]
\[B ::= E < E\]
\[E ::= x \mid y \mid E \& E \mid (E | E)\]
\[S ::= S; S \mid x := E \mid y := E\]

Realizable

Specification: \([f(6,9) = (15, _), f(14,15) = (1, _)]\)
Syntax:
\[\text{Start} ::= \text{While } B \text{ do } S\]
\[B ::= E < E\]
\[E ::= x \mid y \mid E \& E \mid (E \mid E)\]
\[S ::= S ; S \mid x ::= E \mid y ::= E\]

\(\text{Sem}_\text{Start}\) satisfies the following lemma:
\[
\forall t, x', y'. \text{Sem}_\text{Start}((14,15), t, (x', y')) \Rightarrow x' \& 4 = 4
\]

\[\text{Syn}_\text{Start}(t) \quad \text{Sem}_\text{Start}((6,9), t, (15,\_)) \quad \text{Sem}_\text{Start}((14,15), t, (1,\_))\]

Realizable

Specification: \([f(6, 9) = (15,\_), f(14, 15) = (1,\_)]\)
Syntax:
Start ::= While B do S
B ::= E < E
E ::= x | y | E & E | (E | E)
S ::= S; S | x := E | y := E

Sem_{Start} satisfies the following lemma:
\forall t, x', y'. Sem_{Start}((14, 15), t, (x', y')) \Rightarrow x' & 4 = 4

14 = 1110_b
15 = 1111_b

Syn_{Start}(t) \quad Sem_{Start}((6, 9), t, (15, _)) \quad Sem_{Start}((14, 15), t, (1, _))
Realizable

Specification: [f(6, 9) = (15, _), f(14, 15) = (1, _)]
Syntax:
\[
\text{Start} ::= \text{While } B \text{ do } S \\
B ::= E < E \\
E ::= x \mid y \mid E \& E \mid (E \mid E) \\
S ::= S ; S \mid x := E \mid y := E
\]

\[\text{Sem}_{\text{Start}}\] satisfies the following lemma:
\[\forall t, x', y'. \text{ Sem}_{\text{Start}}((14, 15), t, (x', y')) \Rightarrow x' \& 4 = 4\]

For the desired output, \(1 \& 4 \neq 4\): Problem is unrealizable!

\[
\begin{array}{c}
\text{Syn}_{\text{Start}}(t) \\
\text{Sem}_{\text{Start}}((6, 9), t, (15, _)) \\
\text{Sem}_{\text{Start}}((14, 15), t, (1, _))
\end{array}
\]

Realizable

Specification: \([f(6, 9) = (15, _), f(14, 15) = (1, _)]\)
Semantics-Guided Synthesis Problem

Syntax: Regular Tree Grammar
Semantics: Constrained Horn Clauses
Specification: Constraints / Examples

Highlights
- Defining the Semgus Framework
- CHC-based solving for Semgus problems
- Alternative semantics in synthesis
- Unrealizability for imperative Programs!

What’s Next?
- More benchmarks for Semgus
- More Semgus solvers
- Practical issues: Efficiency, Scalability
- Theoretical questions: Can we further rely on semantics to guide synthesis?