IMPLEMENTING BRAHMA:
A LOOP-FREE PROGRAM SYNTHESIZER



Motivation

- ADVANTAGE OVER 3 PS DIMENTIONS

Methods

- BUILDING 4 MAIN COMPONENTS
- CEGIS-BASED ALGORITHM

Intentions

- WORKING WITH Z3
- FUTURE INTENTIONS

OUTLINE

BRAHMA PROS OVER 3 PS DIMENSIONS

Behavioral Constraints | Structural Constraints | Search Strategies

- Using base components:
  - Useful for hardware developers
  - Scalable verification of modular-designed systems
  - Program is correct by construction

BRAHMA PROS OVER 3 PS DIMENSIONS

Behavioral Constraints | **Structural Constraints** | Search Strategies

- Bitvector programs:
  - Hard to discover; even for experts
  - Can reduce vulnerability in software:

$(x + y)/2$

Sensitive to overflow!

$(x|y) - ((x \oplus y) \gg 1)$

BRAHMA PROS OVER 3 PS DIMENSIONS

Behavioral Constraints | **Structural Constraints** | **Search Strategies**

- Leveraging constraint-based search:
  - Way much better than enumerative search
  - Exploitation of the SAT and SMT solvers strength

BUILDING THE CONSTRAINTS (AN EXAMPLE)

$$\psi_{wfp}(L) \wedge \left(\phi_{lib}(T) \wedge \psi_{conn}(\vec{I}, O, T, L) \rightarrow \phi_{spec}(\vec{I}, O) \right)$$

- **P1(x)** : Turn-off the rightmost 1 bit

$$0011100 \longrightarrow 0011000$$

BUILDING THE CONSTRAINTS (AN EXAMPLE)

$$\psi_{wfp}(L) \wedge \left(\phi_{lib}(T) \wedge \psi_{conn}(\vec{I}, O, T, L) \rightarrow \phi_{spec}(\vec{I}, O) \right)$$

$$\phi_{\mathrm{spec}}(I, O) \quad := \quad \bigwedge_{t=1}^{b} \left(\left(I[t] = 1 \wedge \bigwedge_{j=t+1}^{b} I[j] = 0 \right) \Rightarrow \left(O[t] = 0 \wedge \bigwedge_{j \neq t} O[j] = I[j] \right) \right)$$

BUILDING THE CONSTRAINTS (AN EXAMPLE)

$\psi_{wfp}(L) \wedge (\phi_{lib}(T) \wedge \psi_{conn}(\vec{I}, O, T, L) \rightarrow \phi_{spec}(\vec{I}, O))$

$$\psi_{\text{conn}} := \bigwedge_{x,y \in \mathbf{P} \cup \mathbf{R} \cup \vec{I} \cup \{O\}} (l_x = l_y \Rightarrow x = y)$$

BUILDING THE CONSTRAINTS (AN EXAMPLE)

$$\psi_{wfp}(L) \wedge \left(\phi_{lib}(T) \wedge \psi_{conn}(\vec{I}, O, T, L) \rightarrow \phi_{spec}(\vec{I}, O) \right)$$

$$\phi_1(I_1, O_1) \quad := \quad O_1 = (I_1 - 1)$$

$$\phi_2(I_2, I_2', O_2) \quad := \quad O_2 = (I_2 \,\&\, I_2')$$

BUILDING THE CONSTRAINTS (AN EXAMPLE)

$$\psi_{wfp}(L) \wedge \left(\phi_{lib}(T) \wedge \psi_{conn}(\vec{I}, O, T, L) \rightarrow \phi_{spec}(\vec{I}, O) \right)$$

$$\psi_{\mathtt{wfp}} := \psi_{\mathtt{cons}} \wedge \psi_{\mathtt{acyc}} \wedge \bigwedge_{x \in \mathbf{P}} (0 \leq l_x \leq 2) \wedge \bigwedge_{x \in \mathbf{R}} (1 \leq l_x \leq 2)$$

where $\psi_{\mathtt{cons}} := (l_{O_1} \neq l_{O_2})$

and $\psi_{\mathtt{acyc}} := (l_{I_1} < l_{O_1}) \wedge (l_{I_2} < l_{O_2}) \wedge (l_{I_2'} < l_{O_2})$

BUILDING THE CONSTRAINTS (AN EXAMPLE)

$$\psi_{wfp}(L) \wedge \left(\phi_{lib}(T) \wedge \psi_{conn}(\vec{I}, O, T, L) \rightarrow \phi_{spec}(\vec{I}, O) \right)$$

$$\exists L : (\psi_{\mathtt{wfp}}(L) \wedge \forall \vec{I}, O, \mathbf{P}, \mathbf{R} :$$
$$\phi_{\mathtt{lib}}(\mathbf{P}, \mathbf{R}) \ \wedge \ \psi_{\mathtt{conn}}(\vec{I}, O, \mathbf{P}, \mathbf{R}, L) \Rightarrow \phi_{\mathtt{spec}}(\vec{I}, O))$$

$$\exists L \forall \vec{I}, O, T : \psi_{\mathtt{wfp}}(L) \wedge$$
$$(\phi_{\mathtt{lib}}(T) \wedge \psi_{\mathtt{conn}}(\vec{I}, O, T, L) \Rightarrow \phi_{\mathtt{spec}}(\vec{I}, O))$$

ITERATING THROUGH INPUTS (CEGIS-BASED ALGORITHM)

```
   ExAllSolver(ψ_wfp, φ_lib, ψ_conn, φ_spec):
1  // ∃L∀I⃗,O,T : ψ_wfp ∧ (φ_lib ∧ ψ_conn ⇒ φ_spec)
   //        is a synthesis constraint
2  // Output: synthesis failed or values for L
3  S := {I⃗_0} // I⃗_0 is an arbitrary input
4  while (1) {
5     model := T-SAT(∃L,O_1,...,O_n,T_1,...,T_n : ψ_wfp(L)∧
                    ⋀_{I⃗_i∈S}(φ_lib(T_i) ∧ ψ_conn(I⃗_i,O_i,T_i,L)
                          ∧φ_spec(I⃗_i,O_i)));
      if (model ≠ ⊥) {
6        currL := model|_L
      } else {
7        return("synthesis failed");
      }
8     model := T-SAT(∃I⃗,O,T :ψ_conn(I⃗,O,T,currL)∧
                          φ_lib(T) ∧ ¬φ_spec(I⃗,O));
      if (model ≠ ⊥) {
9        I⃗_1 := model|_I⃗; S := S ∪ {I⃗_1};
      } else {
10       return(currL);
      }
11 }
```

OUR WORKS…

OUR WORKS…

FUTURE WORK

- Paper implementation
  - Achieving the results stated in the paper
- Improving running time
  - Comparing solvers (e.g. Z3, CVC5, …)
  - Inspecting different strategies for transforming specs
- Testing solution on other examples
- Reasoning about UNSAT cases

Motivation

- ADVANTAGE OVER 3 PS DIMENTIONS

Methods

- BUILDING 4 MAIN COMPONENTS
- CEGIS-BASED ALGORITHM

Intentions

- WORKING WITH Z3
- FUTURE INTENTIONS

OUTLINE