Finding Polynomial and Array Invariants using Dynamic Analysis

ThanhVu (Vu) Nguyen*
Deepak Kapur*, Westley Weimer†, Stephanie Forrest*

*University of New Mexico, †University of Virginia

June 8, 2012
Ensuring that a program behaves correctly is critical.
Ensuring that a program behaves correctly is critical

Dynamic analysis discovers invariants from traces

- Input: traces (observed values of variables)
- Output: relations among variables
Introduction

Ensuring that a program behaves correctly is critical

Dynamic analysis discovers invariants from traces

- Input: traces (observed values of variables)
- Output: relations among variables

Our dynamic system discovers two forms of invariants

- Polynomials
  - Equalities: $x - y^q = r$, $x^{10} = -10^{23478}$
  - Inequalities: $x^2 - \varepsilon \leq y \leq x^2 + \varepsilon$

- Arrays
  - Nested relations: $A[i][j] = B[C[i]+j][C[3][j]]$
Introduction

Ensuring that a program behaves correctly is critical.

Dynamic analysis discovers invariants from traces
- Input: traces (observed values of variables)
- Output: relations among variables

Our dynamic system discovers two forms of invariants
- Polynomials
  - Equalities: \( x - yq = r, x^{10} = -10.23478 \)
  - Inequalities: \( x^2 - \varepsilon \leq y \leq x^2 + \varepsilon \)
- Arrays
  - Nested relations: \( A[i][j] = B[C[i + j]][C[3j]] \)
Daikon

- Comes with a large set of pre-defined templates
  - Polynomials: \( x + 2y - 3z + 4 = 0, x = y^2 \)
  - Arrays: sorted\((A)\), member\((a, A)\), reverse\((A, B)\), \(A = B\)

- User-defined: \( x = y^2 + 10, x = y^3 \)

- Filters out templates from traces
- Cannot find general linear or nonlinear relations
- Has limited support for relations among arrays
Overview of the System
Example: Cohen Integer Division

```python
1  def intdiv(x, y):
2      q = 0
3      r = x
4      while r ≥ y:
5          a = 1
6          b = y
7          while r ≥ 2b:
8              [L]
9              a = 2a
10             b = 2b
11             r = r - b
12             q = q + a
13             return q
```
```python
def intdiv(x, y):
    q = 0
    r = x
    while r >= y:
        a = 1
        b = y
        while r >= 2*b:
            [L]
            a = 2*a
            b = 2*b
            r = r - b
            q = q + a
    return q
```

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>y</td>
<td>a</td>
<td>b</td>
<td>q</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>15</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>15</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>15</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>4</td>
</tr>
</tbody>
</table>
Example: Cohen Integer Division

```python
def intdiv(x, y):
    q = 0
    r = x
    while r ≥ y:
        a = 1
        b = y
        while r ≥ 2b:
            a = 2a
            b = 2b
            r = r - b
        q = q + a
    return q
```

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>a</th>
<th>b</th>
<th>q</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>15</td>
</tr>
<tr>
<td>15</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>0</td>
<td>15</td>
</tr>
<tr>
<td>15</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>4</td>
</tr>
</tbody>
</table>

Invariants at L:
\{ b = ya, x = qy + r, r ≥ 2ya \}
Example: Cohen Integer Division

```python
def intdiv(x, y):
    q = 0
    r = x
    while r ≥ y:
        a = 1
        b = y
        while r ≥ 2b:
            [L]
            a = 2a
            b = 2b
            r = r - b
            q = q + a
            return q
```

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>a</th>
<th>b</th>
<th>q</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>15</td>
</tr>
<tr>
<td>15</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>0</td>
<td>15</td>
</tr>
<tr>
<td>15</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>4</td>
</tr>
</tbody>
</table>

Invariants at L: \( \{ b = ya, x = qy + r, r ≥ 2ya \} \)
Nonlinear Equations

Examples

Integer division: \( x = qy + r \)

Extended gcd: \( \gcd_{A,B} = iA + jB \)

Find equations of the form

\[
c_0 + c_1x + c_2y + c_3xy + \cdots + c_nx^dy^d = 0, \quad c_i \in \mathbb{R}
\]

Method

- Generates equations from program traces
- Solves equations using a standard equation solver
Finding Nonlinear Equations using Equation Solver

Terms and degrees

\[ V = \{r, y, a\}; \quad \text{deg}_{\text{max}} = 2 \quad \Rightarrow \quad T = \{1, r, y, a, ry, ra, ya, r^2, y^2, a^2\} \]
Finding Nonlinear Equations using Equation Solver

- Terms and degrees

\[ V = \{r, y, a\}; \quad \text{deg}_{\text{max}} = 2 \quad \Rightarrow \quad T = \{1, r, y, a, ry, ra, ya, r^2, y^2, a^2\} \]

\[ T = \{\ldots, \log(r), a^y, \sin(y), \ldots\} \]
Finding Nonlinear Equations using Equation Solver

- **Terms and degrees**
  \[ V = \{ r, y, a \}; \quad \text{deg}_{\text{max}} = 2 \quad \Rightarrow \quad T = \{ 1, r, y, a, ry, ra, ya, r^2, y^2, a^2 \} \]

- **Equation template**
  \[ c_1 + c_2 r + c_3 y + c_4 a + c_5 ry + c_6 ra + c_7 ya + c_8 r^2 + c_9 y^2 + c_{10} a^2 = 0 \]
Finding Nonlinear Equations using Equation Solver

- Terms and degrees
  \[ V = \{r, y, a\}; \quad \text{deg}_{\max} = 2 \quad \Rightarrow \quad T = \{1, r, y, a, ry, ra, ya, r^2, y^2, a^2\} \]

- Equation template
  \[ c_1 + c_2 r + c_3 y + c_4 a + c_5 ry + c_6 ra + c_7 ya + c_8 r^2 + c_9 y^2 + c_{10} a^2 = 0 \]

- System of linear equations
  trace 1 : \{r = 15, y = 2, a = 1\}
  eq 1 : \[ c_1 + 15 c_2 + 2 c_3 + c_4 + 30 c_5 + 15 c_6 + 2 c_7 + 225 c_8 + 4 c_9 + c_{10} = 0 \]
  :.
  :.
Finding Nonlinear Equations using Equation Solver

- **Terms and degrees**
  
  \[ V = \{r, y, a\}; \quad \text{deg}_{\text{max}} = 2 \quad \Rightarrow \quad T = \{1, r, y, a, ry, ra, ya, r^2, y^2, a^2\} \]

- **Equation template**
  
  \[ c_1 + c_2 r + c_3 y + c_4 a + c_5 ry + c_6 ra + c_7 ya + c_8 r^2 + c_9 y^2 + c_{10} a^2 = 0 \]

- **System of linear equations**

  **trace 1** : \{r = 15, y = 2, a = 1\}

  **eq 1** : \[ c_1 + 15c_2 + 2c_3 + c_4 + 30c_5 + 15c_6 + 2c_7 + 225c_8 + 4c_9 + c_{10} = 0 \]

  :  

- **Solve for coefficients** \(c_i\)

  \[ V = \{x, y, a, b, q, r\}; \quad \text{deg}_{\text{max}} = 2 \quad \Rightarrow \quad \{b = ya, x = qy + r\} \]
Nonlinear Inequalities

Example:

Square root: \( x + \varepsilon \geq y^2 \geq x - \varepsilon \)

Find inequalities of the form

\[ c_0 + c_1 x + c_2 y + c_3 xy + \cdots + c_n x^d y^d \geq 0, \quad c_i \in \mathbb{R} \]

Method

1. Polyhedra
   - Represents trace values as points
   - Builds a bounded convex polyhedron and extracts facets

2. Deduction
   - Deduces invariants when additional information is given
Finding Nonlinear Inequalities using Polyhedra
Finding Nonlinear Inequalities using Polyhedra
Finding Nonlinear Inequalities using Polyhedra
Finding Nonlinear Inequalities using Deduction

- Use inequality tests \( P \) from loop or branch conditions

\[
\text{while } (r \geq 2b) \{ [L] \ldots \}
\]
Finding Nonlinear Inequalities using Deduction

- Use inequality tests $P$ from loop or branch conditions
  
  while ($r \geq 2b$) { [L] ... }

- Obtain equality relations $Q$ at L
  
  \{ $b = ay, qy + r = x$ \}
Finding Nonlinear Inequalities using Deduction

- Use inequality tests $P$ from loop or branch conditions

  \[ \text{while } (r \geq 2b) \ \{ [L] \ldots \} \]

- Obtain equality relations $Q$ at $L$
  
  \[ \{ b = ay, qy + r = x \} \]

- Deduce new, non-trivial inequality relations at $L$ from $P$ and $Q$

  \( (r \geq 2b \land b = ay) \implies r \geq 2ay \)

  \( (r \geq 2b \land qy + r = x) \implies x - qy \geq 2b \)
Results for Polynomial Invariants

<table>
<thead>
<tr>
<th>Program</th>
<th>Desc</th>
<th>Inv Type</th>
<th>Time (secs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>divbin</td>
<td>div</td>
<td>eq</td>
<td>1</td>
</tr>
<tr>
<td>cohendiv</td>
<td>div</td>
<td>eq, ieq</td>
<td>0.5</td>
</tr>
<tr>
<td>mannadiv</td>
<td>int div</td>
<td>eq</td>
<td>1.3</td>
</tr>
<tr>
<td>hard</td>
<td>int div</td>
<td>eq</td>
<td>0.9</td>
</tr>
<tr>
<td>sqrt1</td>
<td>sqr</td>
<td>eq, ieq</td>
<td>0.7</td>
</tr>
<tr>
<td>dijkstra</td>
<td>sqr</td>
<td>eq</td>
<td>0.2</td>
</tr>
<tr>
<td>freire1</td>
<td>sqr</td>
<td>eq</td>
<td>3.2</td>
</tr>
<tr>
<td>freire2</td>
<td>cubic root</td>
<td>eq</td>
<td>12.6</td>
</tr>
<tr>
<td>cohencube</td>
<td>cube</td>
<td>eq</td>
<td>6.5</td>
</tr>
<tr>
<td>euclidex1</td>
<td>gcd</td>
<td>eq</td>
<td>2.5</td>
</tr>
<tr>
<td>euclidex2</td>
<td>gcd</td>
<td>eq</td>
<td>10.1</td>
</tr>
<tr>
<td>euclidex3</td>
<td>gcd</td>
<td>eq</td>
<td>0.5</td>
</tr>
<tr>
<td>lcm1</td>
<td>gcd, lcm</td>
<td>eq</td>
<td>0.6</td>
</tr>
<tr>
<td>lcm2</td>
<td>gcd, lcm</td>
<td>eq</td>
<td>35</td>
</tr>
<tr>
<td>prodbin</td>
<td>product</td>
<td>eq</td>
<td>8.1</td>
</tr>
<tr>
<td>prod4br</td>
<td>product</td>
<td>eq</td>
<td>0.2</td>
</tr>
<tr>
<td>fermat1</td>
<td>divisor</td>
<td>eq</td>
<td>0.8</td>
</tr>
<tr>
<td>fermat2</td>
<td>divisor</td>
<td>eq</td>
<td>71.5</td>
</tr>
<tr>
<td>knuth</td>
<td>divisor</td>
<td>eq</td>
<td>0.4</td>
</tr>
<tr>
<td>geo2</td>
<td>geo series</td>
<td>eq</td>
<td>0.3</td>
</tr>
<tr>
<td>geo3</td>
<td>geo series</td>
<td>eq</td>
<td>0.1</td>
</tr>
<tr>
<td>ps2</td>
<td>pow sum</td>
<td>eq</td>
<td>0.4</td>
</tr>
<tr>
<td>ps3</td>
<td>pow sum</td>
<td>eq</td>
<td>0.3</td>
</tr>
<tr>
<td>ps4</td>
<td>pow sum</td>
<td>eq</td>
<td>0.1</td>
</tr>
</tbody>
</table>

24 programs
### Results for Polynomial Invariants

<table>
<thead>
<tr>
<th>Program</th>
<th>Desc</th>
<th>Inv Type</th>
<th>Vars</th>
<th>Degree</th>
<th>Annotated Invs</th>
</tr>
</thead>
<tbody>
<tr>
<td>divbin</td>
<td>div</td>
<td>eq</td>
<td>5</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>cohendiv</td>
<td>div</td>
<td>eq, ieq</td>
<td>6</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>mannadiv</td>
<td>int div</td>
<td>eq</td>
<td>5</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>hard</td>
<td>int div</td>
<td>eq</td>
<td>6</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>sqrt1</td>
<td>sqr</td>
<td>eq, ieq</td>
<td>4</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>dijkstra</td>
<td>sqr</td>
<td>eq</td>
<td>5</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>freire1</td>
<td>sqr</td>
<td>eq</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>freire2</td>
<td>cubic root</td>
<td>eq</td>
<td>4</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>cohencube</td>
<td>cube</td>
<td>eq</td>
<td>5</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>euclidex1</td>
<td>gcd</td>
<td>eq</td>
<td>10</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>euclidex2</td>
<td>gcd</td>
<td>eq</td>
<td>8</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>euclidex3</td>
<td>gcd</td>
<td>eq</td>
<td>12</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>lcm1</td>
<td>gcd, lcm</td>
<td>eq</td>
<td>6</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>lcm2</td>
<td>gcd, lcm</td>
<td>eq</td>
<td>6</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>prodbin</td>
<td>product</td>
<td>eq</td>
<td>5</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>prod4br</td>
<td>product</td>
<td>eq</td>
<td>6</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>fermat1</td>
<td>divisor</td>
<td>eq</td>
<td>5</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>fermat2</td>
<td>divisor</td>
<td>eq</td>
<td>5</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>knuth</td>
<td>divisor</td>
<td>eq</td>
<td>8</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>geo2</td>
<td>geo series</td>
<td>eq</td>
<td>4</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>geo3</td>
<td>geo series</td>
<td>eq</td>
<td>5</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>ps2</td>
<td>pow sum</td>
<td>eq</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>ps3</td>
<td>pow sum</td>
<td>eq</td>
<td>3</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>ps4</td>
<td>pow sum</td>
<td>eq</td>
<td>3</td>
<td>4</td>
<td>1</td>
</tr>
</tbody>
</table>

24 programs
## Results for Polynomial Invariants

<table>
<thead>
<tr>
<th>Program</th>
<th>Desc</th>
<th>Inv Type</th>
<th></th>
<th>Vars</th>
<th>Degree</th>
<th>Annotated Invs</th>
<th>Discovered Invs</th>
<th>Time (secs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>divbin</td>
<td>div</td>
<td>eq</td>
<td></td>
<td>5</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0.5</td>
</tr>
<tr>
<td>cohendiv</td>
<td>div</td>
<td>eq, ieq</td>
<td></td>
<td>6</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1.3</td>
</tr>
<tr>
<td>mannadiv</td>
<td>int div</td>
<td>eq</td>
<td></td>
<td>5</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0.3</td>
</tr>
<tr>
<td>hard</td>
<td>int div</td>
<td>eq</td>
<td></td>
<td>6</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0.9</td>
</tr>
<tr>
<td>sqrt1</td>
<td>sqr</td>
<td>eq, ieq</td>
<td></td>
<td>4</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>0.7</td>
</tr>
<tr>
<td>dijkstra</td>
<td>sqr</td>
<td>eq</td>
<td></td>
<td>5</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0.5</td>
</tr>
<tr>
<td>freire1</td>
<td>sqr</td>
<td>eq</td>
<td></td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0.2</td>
</tr>
<tr>
<td>freire2</td>
<td>cubic root</td>
<td>eq</td>
<td></td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>3.2</td>
</tr>
<tr>
<td>cohencube</td>
<td>cube</td>
<td>eq</td>
<td></td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>12.6</td>
</tr>
<tr>
<td>euclidex1</td>
<td>gcd</td>
<td>eq</td>
<td></td>
<td>10</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>6.5</td>
</tr>
<tr>
<td>euclidex2</td>
<td>gcd</td>
<td>eq</td>
<td></td>
<td>8</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2.5</td>
</tr>
<tr>
<td>euclidex3</td>
<td>gcd</td>
<td>eq</td>
<td></td>
<td>12</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>10.1</td>
</tr>
<tr>
<td>lcm1</td>
<td>gcd, lcm</td>
<td>eq</td>
<td></td>
<td>6</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0.5</td>
</tr>
<tr>
<td>lcm2</td>
<td>gcd, lcm</td>
<td>eq</td>
<td></td>
<td>6</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0.6</td>
</tr>
<tr>
<td>prodbin</td>
<td>product</td>
<td>eq</td>
<td></td>
<td>5</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0.3</td>
</tr>
<tr>
<td>prod4br</td>
<td>product</td>
<td>eq</td>
<td></td>
<td>6</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>8.1</td>
</tr>
<tr>
<td>fermat1</td>
<td>divisor</td>
<td>eq</td>
<td></td>
<td>5</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0.8</td>
</tr>
<tr>
<td>fermat2</td>
<td>divisor</td>
<td>eq</td>
<td></td>
<td>5</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0.4</td>
</tr>
<tr>
<td>knuth</td>
<td>divisor</td>
<td>eq</td>
<td></td>
<td>8</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>71.5</td>
</tr>
<tr>
<td>geo2</td>
<td>geo series</td>
<td>eq</td>
<td></td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0.2</td>
</tr>
<tr>
<td>geo3</td>
<td>geo series</td>
<td>eq</td>
<td></td>
<td>5</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>3.1</td>
</tr>
<tr>
<td>ps2</td>
<td>pow sum</td>
<td>eq</td>
<td></td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0.1</td>
</tr>
<tr>
<td>ps3</td>
<td>pow sum</td>
<td>eq</td>
<td></td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>0.3</td>
</tr>
<tr>
<td>ps4</td>
<td>pow sum</td>
<td>eq</td>
<td></td>
<td>3</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>0.8</td>
</tr>
</tbody>
</table>

24 programs

|                  | 35 | 35 | avg 3.6 |
Array Invariants
Simple Array Relations

Examples

block2State: \( R[i][j] = t[4i + j] \)
keySetupEnc8: \( R[i][j] = \text{cipherKey}[8i + j] \)

Find simple array relations of the form

\[ A_1 + c_2 A_2 + \cdots + c_n A_n + c_0 = 0, \quad c_i \in \mathbb{R} \]

Method

- Flattens array elements as new variables
- Infers linear equalities among array elements
- Finds relations among array indices
Finding Simple Array Relations

- Represent array elements with new variables

<table>
<thead>
<tr>
<th>Trace</th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trace 1</td>
<td>{-546, -641, 34}, B = {-78, 3, -92, -34, 4}</td>
<td></td>
</tr>
<tr>
<td>Trace 2</td>
<td>{A = [133, -333, -323], B = [19, 96, -48, -80, -47]}</td>
<td></td>
</tr>
<tr>
<td>Trace 3</td>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>

\[
\begin{align*}
\text{trace 1} & : & A_0 & -546 & A_1 & -641 & A_2 & 34 & B_0 & -78 & B_1 & 3 & B_2 & -92 & B_3 & -34 & B_4 & 4 \\
\text{trace 2} & : & A_0 & -133 & A_1 & -333 & A_2 & -323 & B_0 & -19 & B_1 & 96 & B_2 & -48 & B_3 & -80 & B_4 & -47 \\
\text{trace 3} & : & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots
\end{align*}
\]
Finding Simple Array Relations

- Represent array elements with new variables
  
  trace 1: \{A = [-546, -641, 34], B = [-78, 3, -92, -34, 4]\}
  
  trace 2: \{A = [133, -333, -323], B = [19, 96, -48, -80, -47]\}
  
  trace 3: \ldots

\[
\begin{array}{cccccccc}
| & A_0 & A_1 & A_2 & B_0 & B_1 & B_2 & B_3 & B_4 \\
\hline
\text{trace 1} & -546 & -641 & 34 & -78 & 3 & -92 & -34 & 4 \\
\text{trace 2} & -133 & -333 & -323 & -19 & 96 & -48 & -80 & -47 \\
\text{trace 3} & \ldots & & & & & & & \\
\vdots & & & & & & & & \\
\end{array}
\]

- Find linear relations from traces

\begin{align*}
A_0 - 7B_0 &= 0 \\
A_1 - 7B_2 &= 3 \\
A_2 - 7B_4 &= 6
\end{align*}
Finding Simple Array Relations

- Hypothesize

\[ A[i] = lB[j] + k, \quad i \in \{0, 1, 2\} \]
Finding Simple Array Relations

- Hypothesize
  
  \[ A[i] = lB[j] + k, \quad i \in \{0, 1, 2\} \]

- Find \( j \)
Finding Simple Array Relations

- Hypothesize
  \[ A[i] = lB[j] + k, \quad i \in \{0, 1, 2\} \]

- Find \( j \)
  - Express relation between \( A[i] \) and \( B[j] \) as \( j = ip + q \)
Finding Simple Array Relations

- **Hypothesize**
  \[ A[i] = lB[j] + k, \quad i \in \{0, 1, 2\} \]

- **Find** \( j \)
  - Express relation between \( A[i] \) and \( B[j] \) as \( j = ip + q \)
    \[ A_0 - 7B_0 = 0 \quad \Rightarrow \quad 0 = 0p + q \]
Finding Simple Array Relations

- Hypothesize

\[ A[i] = lB[j] + k, \quad i \in \{0, 1, 2\} \]

- Find \( j \)
  - Express relation between \( A[i] \) and \( B[j] \) as \( j = ip + q \)

\[
\begin{align*}
A_0 - 7B_0 &= 0 \quad \Rightarrow \quad 0 = 0p + q \\
A_1 - 7B_2 &= 3 \quad \Rightarrow \quad 2 = 1p + q \\
A_2 - 7B_4 &= 6 \quad \Rightarrow \quad 4 = 2p + q
\end{align*}
\]
Finding Simple Array Relations

- **Hypothesize**
  
  \[ A[i] = lB[j] + k, \quad i \in \{0, 1, 2\} \]

- **Find \( j \)**
  
  - Express relation between \( A[i] \) and \( B[j] \) as \( j = ip + q \)
    
    \[
    \begin{align*}
    A_0 - 7B_0 &= 0 \quad \Rightarrow \quad 0 = 0p + q \\
    A_1 - 7B_2 &= 3 \quad \Rightarrow \quad 2 = 1p + q \\
    A_2 - 7B_4 &= 6 \quad \Rightarrow \quad 4 = 2p + q 
    \end{align*}
    \]

  - Solve for \( p, q \)
    
    \[
    \{ q = 0, p = 2 \} \quad \Rightarrow \quad j = 2i \\
    \Rightarrow \quad A[i] = lB[2i] + k
    \]
Finding Simple Array Relations

- **Hypothesize**
  \[ A[i] = lB[j] + k, \quad i \in \{0, 1, 2\} \]

- **Find** \( j \)
  - Express relation between \( A[i] \) and \( B[j] \) as \( j = ip + q \)
    \[ A_0 - 7B_0 = 0 \quad \Rightarrow \quad 0 = 0p + q \]
    \[ A_1 - 7B_2 = 3 \quad \Rightarrow \quad 2 = 1p + q \]
    \[ A_2 - 7B_4 = 6 \quad \Rightarrow \quad 4 = 2p + q \]

  - Solve for \( p, q \)
    \[ \{q = 0, p = 2\} \quad \Rightarrow \quad j = 2i \]
    \[ \Rightarrow \quad A[i] = lB[2i] + k \]

- **Find** \( l, k \)
  \[ A[i] = 7B[2i] + 3i \]
Nested Array Relations

Examples

\[ \text{invSubBytes} : R[i][j] = S[T[i][j]] \]

Find nested array relations of the grammar

\[ A[i_1] \cdots [i_k] \mapsto e \]
\[ e \mapsto B[e] \cdots [e] \]

E.g. \[ A[i][j] = B[C[j + 3]][D[E[2i + j]]] \]

Method

- Uses reachability analysis to find potential nesting relations
- Reduces to a satisfiability problem and solves with a theorem prover
Finding Nested Array Relations

- **Given**

\[ A = [7, 1, -3], \quad B = [1, -3, 5, 1, 0, 7, 1], \quad C = [8, 5, 6, 6, 2, 1, 4] \]

- **Generate nestings**

\[ A[i] = B[\ldots], \quad A[i] = C[\ldots], \ldots, \quad C[i] = A[\ldots], \quad A[i] = B[C[\ldots]], \ldots \]

- **Validate nestings**

Discard \( B[i] = C[\ldots] \) because \( B[1] \notin C \)
Reachability Analysis on $A[i] = B[C[. . .]]$
Reachability Analysis on $A[i] = B[C[\ldots]]$
Reachability Analysis on \( A[i] = B[C[\ldots]] \)

\[
A[0] = B[C[1]]
\]

\[
A = \begin{bmatrix}
7 & 1 & -3 \\
0 & 1 & 2
\end{bmatrix}
\]

\[
B = \begin{bmatrix}
1 & -3 & 5 & 1 & 0 & 7 & 1 \\
0 & 1 & 2 & 3 & 4 & 5 & 6
\end{bmatrix}
\]

\[
C = \begin{bmatrix}
8 & 5 & 6 & 6 & 2 & 1 & 4 \\
0 & 1 & 2 & 3 & 4 & 5 & 6
\end{bmatrix}
\]

\[
\]
Reachability Analysis on $A[i] = B[C[\ldots]]$
Reachability Analysis on $A[i] = B[C[\ldots]]$

\[
\begin{align*}
A[0] &= B[C[1]] \\
\end{align*}
\]
Reachability Analysis on $A[i] = B[C[\ldots]]$

\begin{align*}
A[0] &= B[C[1]] \\
\end{align*}
Reachability Analysis on $A[i] = B[C[. . .]]$

$A[0] = B[C[1]]$

$A[i] = B[C[j]]$
Equation Solving for $A[i] = B[C[j]]$

\[
A[0] = B[C[1]] \\
\]

- Express relation between $A[i]$ and $B[C[j]]$ as $j = ip + q$
Express relation between $A[i]$ and $B[C[j]]$ as $j = ip + q$

\{1 = 0p + q, 2 = 1p + q, 5 = 2p + q\}
Equation Solving for $A[i] = B[C[j]]$

- $A[0] = B[C[1]]$

Express relation between $A[i]$ and $B[C[j]]$ as $j = ip + q$

$$\{1 = 0p + q, 2 = 1p + q, 5 = 2p + q\}$$

Solve for $p, q$

No Solution
Equation Solving for $A[i] = B[C[j]]$

\[
\begin{align*}
A[0] &= B[C[1]] \\
\end{align*}
\]

Express relation between $A[i]$ and $B[C[j]]$ as $j = ip + q$

\{1 = 0p + q, 3 = 1p + q, 5 = 2p + q\}
Equation Solving for $A[i] = B[C[j]]$

- $A[0] = B[C[1]]$

Express relation between $A[i]$ and $B[C[j]]$ as $j = ip + q$

$\{1 = 0p + q, 3 = 1p + q, 5 = 2p + q\}$

Solve for $p, q$

$\{q = 1, p = 2\} \Rightarrow j = 2i + 1$

$\Rightarrow A[i] = B[C[2i + 1]]$
Supporting Functions

Examples

\[
\text{addRoundKey} : R[i][j] = \text{xor}(T[i][j], H[i][j])
\]
\[
\text{multWord} : R[i] = T[\text{mod}(L[A[i]] + L[B[i]], 255)]
\]

Treat functions as a special type of arrays

\[
m(2, 3) = 6, \quad m(-1, 1) = -1, \quad m(0, 0) = 0, \ldots
\]
\[
\downarrow
\]
\[
M[2][3] = 6, \quad M[-1][1] = -1, \quad M[0][0] = 0, \ldots
\]
Using SMT solver

Apply SMT solving to improve reachability analysis

\[
\begin{align*}
A[0] &= B[C[1]] \\
\downarrow \\
(0p + q = 1) \land (1p + q = 2 \lor 1p + q = 3) \land (2p + q = 5)
\end{align*}
\]
### Results for Array Invariants

<table>
<thead>
<tr>
<th>Function</th>
<th>Desc</th>
<th>Type</th>
<th>Arrays</th>
<th>Dimension</th>
<th>Annotated Invs</th>
</tr>
</thead>
<tbody>
<tr>
<td>multWord</td>
<td>mult</td>
<td>N(4)</td>
<td>7</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>xor2Word</td>
<td>xor</td>
<td>N(1)</td>
<td>4</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>xor3Word</td>
<td>xor</td>
<td>N(1)</td>
<td>5</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>subWord</td>
<td>subs</td>
<td>N(1)</td>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>rotWord</td>
<td>shift</td>
<td>S</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>block2State</td>
<td>convert</td>
<td>S</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>state2Block</td>
<td>convert</td>
<td>S</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>subBytes</td>
<td>subs</td>
<td>N(1)</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>invSubByte</td>
<td>subs</td>
<td>N(1)</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>shiftRows</td>
<td>shift</td>
<td>S</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>invShiftRow</td>
<td>shift</td>
<td>S</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>addKey</td>
<td>add</td>
<td>N(1)</td>
<td>4</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>mixCol</td>
<td>mult</td>
<td>U</td>
<td>4</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>invMixCol</td>
<td>mult</td>
<td>U</td>
<td>4</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>keySetEnc4</td>
<td>driver</td>
<td>S,U</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>keySetEnc6</td>
<td>driver</td>
<td>S,U</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>keySetEnc8</td>
<td>driver</td>
<td>S,U</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>keySetEnc</td>
<td>driver</td>
<td>U</td>
<td>4</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>keySetDec</td>
<td>driver</td>
<td>U</td>
<td>4</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>keySched1</td>
<td>driver</td>
<td>U</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>keySched2</td>
<td>driver</td>
<td>U</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>aesKeyEnc</td>
<td>driver</td>
<td>eq,U</td>
<td>7</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>aesKeyDec</td>
<td>driver</td>
<td>eq,U</td>
<td>7</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>aesEncrypt</td>
<td>driver</td>
<td>U</td>
<td>8</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>aesDecrypt</td>
<td>driver</td>
<td>U</td>
<td>8</td>
<td>4</td>
<td>1</td>
</tr>
</tbody>
</table>

25 functions  N= Nested, S=Simple, U=unsupported  30
## Results for Array Invariants

<table>
<thead>
<tr>
<th>Function</th>
<th>Desc</th>
<th>Inv Type</th>
<th>Arrays</th>
<th>Dimension</th>
<th>Annotated Invs</th>
<th>Discovered Invs</th>
<th>Time (secs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>multWord</td>
<td>mult</td>
<td>N(4)</td>
<td>7</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>3.6</td>
</tr>
<tr>
<td>xor2Word</td>
<td>xor</td>
<td>N(1)</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0.1</td>
</tr>
<tr>
<td>xor3Word</td>
<td>xor</td>
<td>N(1)</td>
<td>5</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>0.1</td>
</tr>
<tr>
<td>subWord</td>
<td>subs</td>
<td>N(1)</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.4</td>
</tr>
<tr>
<td>rotWord</td>
<td>shift</td>
<td>S</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.5</td>
</tr>
<tr>
<td>block2State</td>
<td>convert</td>
<td>S</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2.0</td>
</tr>
<tr>
<td>state2Block</td>
<td>convert</td>
<td>S</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>11.7</td>
</tr>
<tr>
<td>subBytes</td>
<td>subs</td>
<td>N(1)</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0.6</td>
</tr>
<tr>
<td>invSubByte</td>
<td>subs</td>
<td>N(1)</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>3.8</td>
</tr>
<tr>
<td>shiftRows</td>
<td>shift</td>
<td>S</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>12.2</td>
</tr>
<tr>
<td>invShiftRow</td>
<td>shift</td>
<td>S</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>8.3</td>
</tr>
<tr>
<td>addKey</td>
<td>add</td>
<td>N(1)</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0.6</td>
</tr>
<tr>
<td>mixCol</td>
<td>mult</td>
<td>U</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>-</td>
</tr>
<tr>
<td>invMixCol</td>
<td>mult</td>
<td>U</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>-</td>
</tr>
<tr>
<td>keySetEnc4</td>
<td>driver</td>
<td>S,U</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>4.5</td>
</tr>
<tr>
<td>keySetEnc6</td>
<td>driver</td>
<td>S,U</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>6.7</td>
</tr>
<tr>
<td>keySetEnc8</td>
<td>driver</td>
<td>S,U</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>10.6</td>
</tr>
<tr>
<td>keySetEnc</td>
<td>driver</td>
<td>U</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>-</td>
</tr>
<tr>
<td>keySetDec</td>
<td>driver</td>
<td>U</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>-</td>
</tr>
<tr>
<td>keySched1</td>
<td>driver</td>
<td>U</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>-</td>
</tr>
<tr>
<td>keySched2</td>
<td>driver</td>
<td>U</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>-</td>
</tr>
<tr>
<td>aesKeyEnc</td>
<td>driver</td>
<td>eq,U</td>
<td>7</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.1</td>
</tr>
<tr>
<td>aesKeyDec</td>
<td>driver</td>
<td>eq,U</td>
<td>7</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.1</td>
</tr>
<tr>
<td>aesEncrypt</td>
<td>driver</td>
<td>U</td>
<td>8</td>
<td>4</td>
<td>1</td>
<td>0</td>
<td>-</td>
</tr>
<tr>
<td>aesDecrypt</td>
<td>driver</td>
<td>U</td>
<td>8</td>
<td>4</td>
<td>1</td>
<td>0</td>
<td>-</td>
</tr>
</tbody>
</table>

25 functions  N=Nested, S=Simple, U=unsupported  

| 30 | 17 | tot 65.9 |
Summary

Dynamic Analysis

- Polynomial Invariants
  - Equalities: solve equations over (nonlinear) terms
  - Inequalities: construct a polyhedron over trace points
    Use deduction when additional information is available

- Array Invariants
  - Simple relations: find relations among individual elements
  - Nested relations: perform reachability analysis and use SMT solving

Results

- Identify 100% of the nonlinear invariants in 24 arithmetic algorithms
- Find 60% of the array relations of an AES implementation
- Current dynamic analysis work cannot find these invariants
Thank you for your attention!

Project is open source and available at

http://code.google.com/p/invgen/

Ask me about (or read the paper for details)

- refining and dealing with spurious invariants (more theorem proving)
- complexities of the techniques (e.g. finding nested arrays is NP-complete)
- additional invariants (e.g. disjunctive properties)
- using discovered invariants to repair programs (current work)