Using Dynamic Analysis to Generate Disjunctive Invariants

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

*University of New Mexico, †University of Virginia

ICSE 2014
Program Invariants

- “Invariants are asserted properties, such as relations among variables, at certain locations in a program”
  - Assertions
  - Pre/Post conditions
  - Loop invariants
Program Invariants

- “Invariants are asserted properties, such as relations among variables, at certain locations in a program”
  - Assertions
  - Pre/Post conditions
  - Loop invariants

- Uses of Invariants
  - Understand and verify programs
  - Debug (locate errors)
  - Formal proofs
  - Documentation
Program Invariants

• “Invariants are asserted properties, such as relations among variables, at certain locations in a program”
  • Assertions
  • Pre/Post conditions
  • Loop invariants

• Uses of Invariants
  • Understand and verify programs
  • Debug (locate errors)
  • Formal proofs
  • Documentations

• Invariants can generated using static or dynamic analysis
  • Static analysis examines program source code
  • Dynamic learns from program execution traces
Polynomial Invariants

- A **polynomial** invariant is a relation among numerical program variables, e.g., \( x = 2y + 3, |arr| \geq x \geq 0 \)

- A conjunctive polynomial invariant is a set (logical and) of polynomial invariants, e.g., \( x = 2y + 3 \land |arr| \geq x \geq 0 \)

- A disjunctive polynomial invariant is a disjunct (logical or) of polynomial invariants, e.g., \( x = 2y + 3 \lor |arr| \geq x \geq 0 \)

- Represent semantics of branching
  The invariant after
  \[
  \text{if (p) \{ a=1; \} else \{ a=2; \}}
  \]
  is \( (p \land a = 1) \lor (\neg p \land a = 2) \)

- Existing approaches focus mostly on conjunctive relations

- Existing approaches have trade-offs among soundness, efficiency, and expressive power

- Static analyzers, e.g., Interproc, Astrée, support conjunctive relations

- Dynamic techniques, e.g., Daikon, also have limited support for disjunctive invariants and can produce spurious results
A polynomial invariant is a relation among numerical program variables, e.g., \( x = 2y + 3, |arr| \geq x \geq 0 \)

A conjunctive polynomial invariant is a set (logical and) of polynomial invariants, e.g., \( x = 2y + 3 \land |arr| \geq x \geq 0 \)
Polynomial Invariants

- A polynomial invariant is a relation among numerical program variables, e.g., \( x = 2y + 3, |arr| \geq x \geq 0 \)
- A conjunctive polynomial invariant is a set (logical and) of polynomial invariants, e.g., \( x = 2y + 3 \land |arr| \geq x \geq 0 \)
- A disjunctive polynomial invariant is a disjunct (logical or) of polynomial invariants, e.g., \( x = 2y + 3 \lor |arr| \geq x \geq 0 \)
  - Represent semantics of branching
    The invariant after \( \text{if } (p) \{a=1;\} \text{ else } \{a=2;\} \) is \( (p \land a = 1) \lor (\neg p \land a = 2) \)
  - Existing approaches focus mostly on conjunctive relations
Polynomial Invariants

- A **polynomial** invariant is a relation among numerical program variables, e.g., \( x = 2y + 3, |arr| \geq x \geq 0 \)

- A **conjunctive** polynomial invariant is a set (logical and) of polynomial invariants, e.g., \( x = 2y + 3 \land |arr| \geq x \geq 0 \)

- A **disjunctive polynomial** invariant is a disjunct (logical or) of polynomial invariants, e.g., \( x = 2y + 3 \lor |arr| \geq x \geq 0 \)
  - Represent semantics of branching
    - The invariant after if (p) {a=1;} else {a=2;} is \((p \land a = 1) \lor (\neg p \land a = 2)\)
  - Existing approaches focus mostly on *conjunctive* relations

- Existing approaches have *trade-offs* among soundness, efficiency, and expressive power
  - Static analyzers, e.g., Interproc, Astrée, support conjunctive relations
  - Dynamic techniques, e.g., Daikon, also have limited support for disjunctive invariants and can produce spurious results
Hybrid Invariant Generation

- **DIG** (Dynamic Invariant Generator)
  - Find invariants directly from program traces
  - Build *nonconvex polyhedra* over trace points and extract facets representing disjunctive relations

- **KIP** (K-Inductive Prover)
  - Verify candidate invariants statically from program code
  - Based on k-induction and SMT solving
Geometric Invariant Inference

- Treat trace values as points in multi-dimensional space
- Build a convex hull (polyhedron) over the points
- Representation of a polyhedron: a conjunction of inequalities
Geometric Invariant Inference

- Treat trace values as points in multi-dimensional space
- Build a **convex hull** (polyhedron) over the points
- Representation of a polyhedron: a **conjunction** of inequalities

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>-2</td>
<td>1</td>
</tr>
<tr>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>1</td>
<td>-3</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>-2</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
</tr>
</tbody>
</table>

program traces

\[
\begin{align*}
\text{trace pts in 2D} & \\
\text{polygon} & \quad c_1 x + c_2 y \geq c
\end{align*}
\]
Geometric Invariant Inference

- Treat trace values as points in multi-dimensional space
- Build a convex hull (polyhedron) over the points
- Representation of a polyhedron: a conjunction of inequalities

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$x$</td>
<td>$y$</td>
</tr>
<tr>
<td>-2</td>
<td>1</td>
</tr>
<tr>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>1</td>
<td>-3</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>-2</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
</tr>
</tbody>
</table>

Program traces

- Consider simpler shapes (decreasing precision, increasing efficiency)
Example

def ex(x):
    y = 5
    if x > y: x = y
    while [L] x ≤ 10:
        if x ≥ 5:
            y = y + 1
        x = x + 1
    assert y ≡ 11

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>11</td>
<td>11</td>
</tr>
</tbody>
</table>

Disjunction of 2 cases:
1. if $x < 5$ then $y = 5$
2. if $x ≥ 5$ then $x = y$

$\max(0, x - 5) = y - 5$

A linear relation .. in max-plus algebra
Example

```python
def ex(x):
    y = 5
    if x > y: x = y
    while [L] x ≤ 10:
        if x ≥ 5:
            y = y + 1
        x = x + 1
    assert y ≡ 11
```

<table>
<thead>
<tr>
<th></th>
<th>x</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>:</td>
<td>:</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>11</td>
<td></td>
</tr>
</tbody>
</table>

$L : (x < 5 \land 5 = y) \lor (x ≥ 5 \land x = y), 11 ≥ x$
Example

```python
def ex(x):
y = 5
if x > y: x = y
while x <= 10:
    if x >= 5:
        y = y + 1
    x = x + 1
assert y == 11
```

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>11</td>
<td>11</td>
</tr>
</tbody>
</table>

$L : (x < 5 \land 5 = y) \lor (x \geq 5 \land x = y), 11 \geq x$

Disjunction of 2 cases:

1. if $x < 5$ then $y = 5$
2. if $x \geq 5$ then $x = y$
Example

```python
def ex(x):
    y = 5
    if x > y: x = y
    while [L] x ≤ 10:
        if x ≥ 5:
            y = y + 1
            x = x + 1
    assert y ≡ 11
```

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>11</td>
<td>11</td>
</tr>
</tbody>
</table>

$L : (x < 5 \land 5 = y) \lor (x ≥ 5 \land x = y), 11 ≥ x$

Disjunction of 2 cases:
1. if $x < 5$ then $y = 5$
2. if $x ≥ 5$ then $x = y$
Example

def ex(x):
y = 5
if x > y: x = y
while [L] x ≤ 10:
  if x ≥ 5:
    y = y + 1
  x = x + 1
assert y ≡ 11

\[
\begin{array}{|c|c|}
\hline
x & y \\
\hline
-1 & 5 \\
\vdots & \vdots \\
5 & 5 \\
6 & 6 \\
\vdots & \vdots \\
11 & 11 \\
\hline
\end{array}
\]

\[
L : (x < 5 \land 5 = y) \lor (x \geq 5 \land x = y), 11 \geq x
\]

Disjunction of 2 cases:

1. if \(x < 5\) then \(y = 5\)

2. if \(x \geq 5\) then \(x = y\)

\[
\leftrightarrow \quad \text{if } 0 > x - 5 \text{ then } 0 = y - 5 \text{ else } x - 5 = y - 5
\]

\[
\max(0, x - 5) = y - 5
\]

a linear relation .. in max-plus algebra
Max-plus Algebra
Max-plus Algebra

<table>
<thead>
<tr>
<th>Linear</th>
<th>Max-plus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Domain</strong></td>
<td>$\mathbb{R}$</td>
</tr>
<tr>
<td><strong>Addition</strong></td>
<td>$+$</td>
</tr>
<tr>
<td><strong>Multiplication</strong></td>
<td>$\times$</td>
</tr>
<tr>
<td><strong>Zero elem</strong></td>
<td>$0$</td>
</tr>
<tr>
<td><strong>Unit elem</strong></td>
<td>$1$</td>
</tr>
<tr>
<td><strong>Relation form</strong></td>
<td>$c_0 + c_1 t_1 + \cdots + c_n t_n \geq 0$</td>
</tr>
</tbody>
</table>
# Max-plus Algebra

## Linear vs. Max-plus

<table>
<thead>
<tr>
<th></th>
<th>Linear</th>
<th>Max-plus</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain</td>
<td>$\mathbb{R}$</td>
<td>$\mathbb{R} \cup {-\infty}$</td>
</tr>
<tr>
<td>Addition</td>
<td>$+$</td>
<td>max</td>
</tr>
<tr>
<td>Multiplication</td>
<td>$\times$</td>
<td>$+$</td>
</tr>
<tr>
<td>Zero elem</td>
<td>0</td>
<td>$-\infty$</td>
</tr>
<tr>
<td>Unit elem</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Relation form</td>
<td>$c_0 + c_1 t_1 + \cdots + c_n t_n \geq 0$</td>
<td>$\max(c_0, c_1 + t_1, \ldots, c_n + t_n) \geq \max(d_0, d_1 + t_1, \ldots, d_n + t_n)$</td>
</tr>
</tbody>
</table>

## Line shapes

- Convex hull
- Not convex in classical sense!
### Max-plus Algebra

<table>
<thead>
<tr>
<th>Linear</th>
<th>Max-plus</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Domain</strong></td>
<td>$\mathbb{R}$</td>
</tr>
<tr>
<td><strong>Addition</strong></td>
<td>$+$</td>
</tr>
<tr>
<td><strong>Multiplication</strong></td>
<td>$\times$</td>
</tr>
<tr>
<td><strong>Zero elem</strong></td>
<td>$0$</td>
</tr>
<tr>
<td><strong>Unit elem</strong></td>
<td>$1$</td>
</tr>
<tr>
<td><strong>Relation form</strong></td>
<td>$c_0 + c_1 t_1 + \cdots + c_n t_n \geq 0$</td>
</tr>
</tbody>
</table>

#### Line shapes
- Convex hull
- Not convex in classical sense!
Max-plus Algebra

<table>
<thead>
<tr>
<th>Linear</th>
<th>Max-plus</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain</td>
<td>( \mathbb{R} )</td>
</tr>
<tr>
<td>Addition</td>
<td>+</td>
</tr>
<tr>
<td>Multiplication</td>
<td>( \times )</td>
</tr>
<tr>
<td>Zero elem</td>
<td>0</td>
</tr>
<tr>
<td>Unit elem</td>
<td>1</td>
</tr>
<tr>
<td>Relation form</td>
<td>( c_0 + c_1 t_1 + \cdots + c_n t_n \geq 0 )</td>
</tr>
</tbody>
</table>

Line shapes

Convex hull

Not convex in classical sense!
Dynamically Inferring Max-plus Invariants

Examples

\[ z = \max(x, y) \equiv (x < y \land z = x) \lor (x \geq y \land z = y) \]
Dynamically Inferring Max-plus Invariants

Examples

\[ z = \max(x, y) \equiv (x < y \land z = x) \lor (x \geq y \land z = y) \]

\[ \text{strncpy}(s, d, n) \equiv (n \geq |s| \land |d| = |s|) \lor (n < |s| \land |d| \geq n) \]
Dynamically Inferring Max-plus Invariants

Examples

\[ z = \max(x, y) \equiv (x < y \land z = x) \lor (x \geq y \land z = y) \]
\[ \text{strncpy}(s, d, n) \equiv (n \geq |s| \land |d| = |s|) \lor (n < |s| \land |d| \geq n) \]

DIG discovers disjunctive relations of the **max-plus** form

\[ \max(c_0, c_1 + t_1, \ldots, c_n + t_n) \geq \max(d_0, d_1 + t_1, \ldots, d_n + t_n) \]
Dynamically Inferring Max-plus Invariants

Examples

\[ z = \max(x, y) \equiv (x < y \land z = x) \lor (x \geq y \land z = y) \]

\[ \operatorname{strncpy}(s, d, n) \equiv (n \geq |s| \land |d| = |s|) \lor (n < |s| \land |d| \geq n) \]

DIG discovers disjunctive relations of the \textit{max-plus} form

\[ \max(c_0, c_1 + t_1, \ldots, c_n + t_n) \geq \max(d_0, d_1 + t_1, \ldots, d_n + t_n) \]

Method

- Represent trace values as points
- Build a max-plus convex polyhedron
- Extract facets represented by max-plus relations
Example

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>11</td>
</tr>
</tbody>
</table>
Example

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>11</td>
</tr>
</tbody>
</table>

• DIG builds a max-plus polygon and finds candidate invs:

\[
11 \geq x \geq -1, \quad 11 \geq y \geq 5, \quad 0 \geq x - y \geq -6
\]

\[
\max(0, x - 5) \geq y - 5 \equiv (x < 5 \land 5 \geq y) \lor (x \geq 5 \land y \leq x)
\]
Example

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>11</td>
<td>11</td>
</tr>
</tbody>
</table>

- DIG builds a max-plus polygon and finds candidate invs:

\[
11 \geq x \geq -1, \quad 11 \geq y \geq 5, \quad 0 \geq x - y \geq -6
\]

\[
\max(0, x - 5) \geq y - 5 \equiv (x < 5 \land 5 \geq y) \lor (x \geq 5 \land y \leq x)
\]

- KIP removes the spurious invariants \( x \geq -1 \) and \( x - y \geq -6 \)

- Remaining invariants are true and equivalent to

\[
(x < 5 \land 5 = y) \lor (11 \geq x \geq 5 \land x = y)
\]
Weak Max-plus Invariants

\[
\max(c_0, c_1 + t_1, \ldots, c_k + t_k) \geq t_j + d
\]

\[
t_j + d \geq \max(c_0, c_1 + t_1, \ldots, c_k + t_k)
\]

- Restrict values of coefficients: \(c_i\) to \(\{0, -\infty\}\)
- Fix the number of variables \(t_i\), e.g., \(k = 2\)
- Allow only one unknown parameter \(d\)
Weak Max-plus Invariants

DIG introduces weak max-plus relations of the form

$$\max(c_0, c_1 + t_1, \ldots, c_k + t_k) \geq t_j + d$$

where $d \in \mathbb{R}$ and $t_j \in \{t_1, \ldots, t_k\}$

- Restrict values of coeffs $c_i$ to $\{0, -\infty\}$
- Fix the number $k$ of variables $t_i$, e.g., $k = 2$
- Allow only one unknown param $d$
• DIG introduces weak max-plus relations of the form

\[
\max(c_0, c_1 + t_1, \ldots, c_k + t_k) \geq t_j + d, \\
 t_j + d \geq \max(c_0, c_1 + t_1, \ldots, c_k + t_k),
\]

where \(d \in \mathbb{R}\) and \(t_j \in \{t_1, \ldots, t_k\}\).
• DIG introduces weak max-plus relations of the form

\[
\max(c_0, c_1 + t_1, \ldots, c_k + t_k) \geq t_j + d, \\
t_j + d \geq \max(c_0, c_1 + t_1, \ldots, c_k + t_k),
\]

where \(d \in \mathbb{R}\) and \(t_j \in \{t_1, \ldots, t_k\}\)

• Restrict values of coefs \(c_i\) to \(\{0, -\infty\}\)

• Fix the number \(k\) of variables \(t_i\), e.g., \(k = 2\)

• Allow only one unknown param \(d\)
Algorithm for Finding Weak Max-plus Invs

Given 2D points \( \{(x_1, y_1), \ldots, (x_n, y_n)\} \), build a weak max-plus polygon, i.e., a conjunction of inequalities of the form

\[
\begin{align*}
\max(c_0, c_1 + x, c_2 + y) & \geq x + d, & x + d & \geq \max(c_0, c_1 + x, c_k + y), \\
\max(c_0, c_1 + x, c_2 + y) & \geq y + d, & y + d & \geq \max(c_0, c_1 + x, c_k + y)
\end{align*}
\]
Algorithm for Finding Weak Max-plus Invs

Given 2D points \( \{(x_1, y_1), \ldots, (x_n, y_n)\} \), build a weak max-plus polygon, i.e., a conjunction of inequalities of the form

\[
\begin{align*}
\max(c_0, c_1 + x, c_2 + y) &\geq x + d, \\
x + d &\geq \max(c_0, c_1 + x, c_k + y), \\
\max(c_0, c_1 + x, c_2 + y) &\geq y + d, \\
y + d &\geq \max(c_0, c_1 + x, c_k + y)
\end{align*}
\]

1. **Enumerate weak relations by instantiating** \( c_i \) **over** \( \{0, -\infty\} \)
   - Each weak max-plus form yields at most 8 relations, e.g., \( \max(c_0, c_1 + x, c_2 + y) \geq x + d \) produces
     \[
     \begin{align*}
     \max(0, x, y) &\geq x + d, \\
     \max(0, x) &\geq x + d, \\
     \max(0, y) &\geq x + d, \\
     0 &\geq x + d, \ldots
     \end{align*}
     \]
   - Obtain at most 32 relations from 4 weak max-plus forms
Algorithm for Finding Weak Max-plus Invs

Given 2D points \( \{(x_1, y_1), \ldots, (x_n, y_n)\} \), build a weak max-plus polygon, i.e., a conjunction of inequalities of the form

\[
\begin{align*}
\max(c_0, c_1 + x, c_2 + y) & \geq x + d, & \quad x + d & \geq \max(c_0, c_1 + x, c_k + y), \\
\max(c_0, c_1 + x, c_2 + y) & \geq y + d, & \quad y + d & \geq \max(c_0, c_1 + x, c_k + y)
\end{align*}
\]

1. **Enumerate** weak relations by instantiating \( c_i \) over \( \{0, -\infty\} \)
   - Each weak max-plus form yields at most 8 relations, e.g., \( \max(c_0, c_1 + x, c_2 + y) \geq x + d \) produces
     \[
     \begin{align*}
     \max(0, x, y) & \geq x + d, & \quad \max(0, x) & \geq x + d, \\
     \max(0, y) & \geq x + d, & \quad 0 & \geq x + d, \ldots
     \end{align*}
     \]
   - Obtain at most 32 relations from 4 weak max-plus forms

2. **Solve** for \( d \) in the obtained relations using given points, e.g.,
   \[
   \begin{align*}
   \max(0, y) & \geq x + d \quad \rightarrow \quad d = \min(\max(0, y_i) - x_i) \\
   x + d & \geq \max(0, y) \quad \rightarrow \quad d = \max(\max(0, y_i) - x_i)
   \end{align*}
   \]
Min-plus Invariants

- DIG discovers disjunctive relations of the \textit{min-plus} form

\[
\min(c_0, c_1 + t_1, \ldots, c_n + t_n) \geq \min(d_0, d_1 + t_1, \ldots, d_n + t_n)
\]
Min-plus Invariants

- DIG discovers disjunctive relations of the \textit{min-plus} form

\[
\min(c_0, c_1 + t_1, \ldots, c_n + t_n) \geq \min(d_0, d_1 + t_1, \ldots, d_n + t_n)
\]

Max-plus

Min-plus
Min-plus Invariants

- DIG discovers disjunctive relations of the min-plus form

\[
\min(c_0, c_1 + t_1, \ldots, c_n + t_n) \geq \min(d_0, d_1 + t_1, \ldots, d_n + t_n)
\]

- Also support weak min-plus relations of the form

\[
\min(c_0, c_1 + t_1, \ldots, c_k + t_k) \geq t_j + d_i, \\
t_j + d_i \geq \min(c_0, c_1 + t_1, \ldots, c_k + t_k)
\]
Min-plus Invariants

- DIG discovers disjunctive relations of the **min-plus** form

\[
\min(c_0, c_1 + t_1, \ldots, c_n + t_n) \geq \min(d_0, d_1 + t_1, \ldots, d_n + t_n)
\]

- Also support **weak** min-plus relations of the form

\[
\begin{align*}
\min(c_0, c_1 + t_1, \ldots, c_k + t_k) & \geq t_j + d_i, \\
t_j + d_i & \geq \min(c_0, c_1 + t_1, \ldots, c_k + t_k)
\end{align*}
\]

- Combine max and min-plus invariants for more expressive power, e.g., can capture iff behavior
## Algorithmic Analysis

<table>
<thead>
<tr>
<th>Complexity</th>
<th>Linear</th>
<th>Max/Min-plus</th>
<th>Weak Max/Min-plus</th>
</tr>
</thead>
<tbody>
<tr>
<td>Complexity</td>
<td>$O(p^{\frac{n}{2}})$</td>
<td>$O(pn^2(p + n)^n)$</td>
<td>$O(p2^k)$</td>
</tr>
</tbody>
</table>

$p = \# \text{ of trace pts}, \ n = \# \text{ of variables}, \ k = \text{a predefined constant, e.g.,} \ k = 2$
Algorithmic Analysis

<table>
<thead>
<tr>
<th></th>
<th>Linear</th>
<th>Max/Min-plus</th>
<th>Weak Max/Min-plus</th>
</tr>
</thead>
<tbody>
<tr>
<td>Complexity</td>
<td>( O(p^{\frac{n}{2}}) )</td>
<td>( O(pn^2(p + n)^n) )</td>
<td>( O(p2^k) )</td>
</tr>
<tr>
<td>Facets</td>
<td>unbounded</td>
<td>unbounded</td>
<td>( k2^{k+2} )</td>
</tr>
</tbody>
</table>

\( p = \# \) of trace pts, \( n = \# \) of variables, \( k = a \) predefined constant, e.g., \( k = 2 \)
### Algorithmic Analysis

<table>
<thead>
<tr>
<th>Linear</th>
<th>Max/Min-plus</th>
<th>Weak Max/Min-plus</th>
</tr>
</thead>
<tbody>
<tr>
<td>Complexity</td>
<td>$O(p^n)$</td>
<td>$O(pn^2(p + n)^n)$</td>
</tr>
<tr>
<td>Facets</td>
<td>unbounded</td>
<td>unbounded</td>
</tr>
</tbody>
</table>

$p = \# \text{ of trace pts}, n = \# \text{ of variables}, k = \text{a predefined constant, e.g., } k = 2$

### Underapproximation Property

- **Theorem**: If the form of the *true* invariant $f$ is supported, then DIG generates only the candidate invariant $f'$ such that $f' \Rightarrow f$

- **Application**: useful for debugging; if $f'$ strictly overapproximates $f$, then there is a trace representing a counterexample violating $f$
Outline
Proving Program Invariants Using k-Induction

- Represent a program execution as a state transition system

\[ M = (I, T), \]

with the initial state \( I \) and the transition relation \( T \)
Proving Program Invariants Using k-Induction

- Represent a program execution as a state transition system

\[ M = (I, T), \]

with the initial state \( I \) and the transition relation \( T \)

- Use k-induction to prove that \( p \) is an invariant of \( M \)

\[
I \land T_1 \land \cdots \land T_k \implies p_0 \land \cdots \land p_k
\]

\[
p_n \land T_{n+1} \land \cdots \land p_{n+k} \land T_{n+k+1} \implies p_{n+k+1}
\]

- More powerful than standard \((k = 0)\) induction
- Help prove properties cannot be proved using standard induction
Example

```python
def sqrt(x):
    assert (x \geq 0);
    a = 0; s = 1; t = 1
    while s \leq x:
        a += 1; t += 2; s += t
    return a
```

DIG
find candidate invariants at \textcolor{red}{L}

\begin{align*}
4s &= t^2 + 2t + 1 \\
t &= 2a + 1 \\
s &= (a + 1)^2 \\
s \geq t \\
9989 \geq x
\end{align*}
Example

```python
def sqrt(x):
    assert (x ≥ 0);
    a = 0; s = 1; t = 1
    while [L] s ≤ x:
        a += 1; t += 2; s += t
    return a
```

### DIG
find candidate invariants at L

### KIP
distinguish true and spurious invs

\[ 4s = t^2 + 2t + 1 \]
\[ t = 2a + 1 \]
\[ s = (a + 1)^2 \]
\[ s ≥ t \]
\[ 9989 ≥ x \]

...
Example

def sqrt(x):
    assert(x >= 0);
    a = 0; s = 1; t = 1
    while[s] s <= x:
        a += 1; t += 2; s += t
    return a

DIG
find candidate invariants at L

KIP
distinguish true and spurious invs

4s = t^2 + 2t + 1
inductive

2^2 + 2t + 1
inductive

s = (a + 1)^2
1-inductive

s >= t

9989 >= x

...
Example

```python
def sqrt(x):
    assert (x \geq 0);
    a = 0; s = 1; t = 1
    while [L] s \leq x:
        a += 1; t += 2; s += t
    return a
```

**DIG**

find candidate invariants at L

- $4s = t^2 + 2t + 1$
- $t = 2a + 1$
- $s = (a + 1)^2$
- $s \geq t$
- $9989 \geq x$
- ...

**KIP**

distinguish true and spurious invs

- inductive
- inductive
- 1-inductive
- potentially non-inductive
Example

```python
def sqrt(x):
    assert(x \geq 0);
    a = 0; s = 1; t = 1
    while s \leq x:
        a += 1; t += 2; s += t
    return a
```

**DIG**
find candidate invariants at \( L \)

**KIP**
distinguish true and spurious invs

\[
\begin{align*}
4s &= t^2 + 2t + 1 \\
t &= 2a + 1 \\
s &= (a + 1)^2 \\
s \geq t \\
9989 \geq x
\end{align*}
\]

inductive
inductive
1-inductive
potentially non-inductive
spurious
Features of KIP

- Iterative k-induction
- Employ the Z3 SMT solver
- Learn lemmas
- Eliminate redundancy
- Parallelism
Outline
Evaluation

Benchmarks

- Disjunctive testsuite: 14 programs require disjunctive invariants
- Nonlinear test suite: 27 programs require nonlinear invariants

Setup

- Implemented in SAGE/Python (with Z3 backend solver)
- Test machine: 64-core 2.6GHZ CPU, 128GB RAM, Linux OS
- Traces obtained at loop entrances and program exits
Evaluation

Benchmarks

- Disjunctive testsuite: 14 programs require disjunctive invariants
- Nonlinear test suite: 27 programs require nonlinear invariants

Setup

- Implemented in SAGE/Python (with Z3 backend solver)
- Test machine: 64-core 2.6GHZ CPU, 128GB RAM, Linux OS
- Traces obtained at loop entrances and program exits

Results

- All generated equalities are valid and most inequalities are spurious (and removed by KIP)
- Identified invariants are sufficiently strong to explain program behavior
- Current dynamic analysis cannot find any of these invariants
# Results for Disjunctive Invariants

<table>
<thead>
<tr>
<th>Prog</th>
<th>Loc</th>
<th>Var</th>
</tr>
</thead>
<tbody>
<tr>
<td>ex</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>strncpy</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>oddeven3</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>oddeven4</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>oddeven5</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>bubble3</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>bubble4</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>bubble5</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>partd3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>partd4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>partd5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>parti3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>parti4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>parti5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td><strong>total</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>16055</td>
<td>317.6</td>
</tr>
<tr>
<td></td>
<td>264</td>
<td>3647.5</td>
</tr>
</tbody>
</table>
# Results for Disjunctive Invariants

<table>
<thead>
<tr>
<th>Prog</th>
<th>Loc</th>
<th>Var</th>
<th>Gen</th>
<th>$T_{Gen}$ (secs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ex</td>
<td>1</td>
<td>2</td>
<td>15</td>
<td>0.2</td>
</tr>
<tr>
<td>strncpy</td>
<td>1</td>
<td>3</td>
<td>69</td>
<td>1.1</td>
</tr>
<tr>
<td>oddeven3</td>
<td>1</td>
<td>6</td>
<td>286</td>
<td>3.7</td>
</tr>
<tr>
<td>oddeven4</td>
<td>1</td>
<td>8</td>
<td>867</td>
<td>12.7</td>
</tr>
<tr>
<td>oddeven5</td>
<td>1</td>
<td>10</td>
<td>2334</td>
<td>56.8</td>
</tr>
<tr>
<td>bubble3</td>
<td>1</td>
<td>6</td>
<td>249</td>
<td>4.1</td>
</tr>
<tr>
<td>bubble4</td>
<td>1</td>
<td>8</td>
<td>832</td>
<td>11.7</td>
</tr>
<tr>
<td>bubble5</td>
<td>1</td>
<td>10</td>
<td>2198</td>
<td>53.9</td>
</tr>
<tr>
<td>partd3</td>
<td>4</td>
<td>5</td>
<td>479</td>
<td>10.5</td>
</tr>
<tr>
<td>partd4</td>
<td>5</td>
<td>6</td>
<td>1217</td>
<td>23.3</td>
</tr>
<tr>
<td>partd5</td>
<td>6</td>
<td>7</td>
<td>2943</td>
<td>53.3</td>
</tr>
<tr>
<td>parti3</td>
<td>4</td>
<td>5</td>
<td>464</td>
<td>10.3</td>
</tr>
<tr>
<td>parti4</td>
<td>5</td>
<td>6</td>
<td>1148</td>
<td>22.4</td>
</tr>
<tr>
<td>parti5</td>
<td>6</td>
<td>7</td>
<td>2954</td>
<td>53.6</td>
</tr>
<tr>
<td><strong>total</strong></td>
<td></td>
<td></td>
<td>16055</td>
<td>317.6</td>
</tr>
</tbody>
</table>
## Results for Disjunctive Invariants

<table>
<thead>
<tr>
<th>Prog</th>
<th>Loc</th>
<th>Var</th>
<th>Gen</th>
<th>$T_{\text{Gen}}$ (secs)</th>
<th>Val</th>
<th>$T_{\text{Val}}$ (secs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ex</td>
<td>1</td>
<td>2</td>
<td>15</td>
<td>0.2</td>
<td>4</td>
<td>1.5</td>
</tr>
<tr>
<td>strncpy</td>
<td>1</td>
<td>3</td>
<td>69</td>
<td>1.1</td>
<td>4</td>
<td>7.7</td>
</tr>
<tr>
<td>oddeven3</td>
<td>1</td>
<td>6</td>
<td>286</td>
<td>3.7</td>
<td>8</td>
<td>16.0</td>
</tr>
<tr>
<td>oddeven4</td>
<td>1</td>
<td>8</td>
<td>867</td>
<td>12.7</td>
<td>22</td>
<td>46.0</td>
</tr>
<tr>
<td>oddeven5</td>
<td>1</td>
<td>10</td>
<td>2334</td>
<td>56.8</td>
<td>52</td>
<td>1319.4</td>
</tr>
<tr>
<td>bubble3</td>
<td>1</td>
<td>6</td>
<td>249</td>
<td>4.1</td>
<td>8</td>
<td>4.9</td>
</tr>
<tr>
<td>bubble4</td>
<td>1</td>
<td>8</td>
<td>832</td>
<td>11.7</td>
<td>22</td>
<td>47.6</td>
</tr>
<tr>
<td>bubble5</td>
<td>1</td>
<td>10</td>
<td>2198</td>
<td>53.9</td>
<td>52</td>
<td>938.2</td>
</tr>
<tr>
<td>partd3</td>
<td>4</td>
<td>5</td>
<td>479</td>
<td>10.5</td>
<td>10</td>
<td>50.8</td>
</tr>
<tr>
<td>partd4</td>
<td>5</td>
<td>6</td>
<td>1217</td>
<td>23.3</td>
<td>15</td>
<td>181.1</td>
</tr>
<tr>
<td>partd5</td>
<td>6</td>
<td>7</td>
<td>2943</td>
<td>53.3</td>
<td>21</td>
<td>418.1</td>
</tr>
<tr>
<td>parti3</td>
<td>4</td>
<td>5</td>
<td>464</td>
<td>10.3</td>
<td>10</td>
<td>45.5</td>
</tr>
<tr>
<td>parti4</td>
<td>5</td>
<td>6</td>
<td>1148</td>
<td>22.4</td>
<td>15</td>
<td>165.1</td>
</tr>
<tr>
<td>parti5</td>
<td>6</td>
<td>7</td>
<td>2954</td>
<td>53.6</td>
<td>21</td>
<td>405.6</td>
</tr>
<tr>
<td><strong>total</strong></td>
<td></td>
<td></td>
<td>16055</td>
<td>317.6</td>
<td>264</td>
<td>3647.5</td>
</tr>
</tbody>
</table>
## Results for Disjunctive Invariants

<table>
<thead>
<tr>
<th>Prog</th>
<th>Loc</th>
<th>Var</th>
<th>Gen</th>
<th>$T_{Gen}$ (secs)</th>
<th>Val</th>
<th>$T_{Val}$ (secs)</th>
<th>Strength</th>
</tr>
</thead>
<tbody>
<tr>
<td>ex</td>
<td>1</td>
<td>2</td>
<td>15</td>
<td>0.2</td>
<td>4</td>
<td>1.5</td>
<td>✓</td>
</tr>
<tr>
<td>strncpyy</td>
<td>1</td>
<td>3</td>
<td>69</td>
<td>1.1</td>
<td>4</td>
<td>7.7</td>
<td>✓</td>
</tr>
<tr>
<td>oddeven3</td>
<td>1</td>
<td>6</td>
<td>286</td>
<td>3.7</td>
<td>8</td>
<td>16.0</td>
<td>✓</td>
</tr>
<tr>
<td>oddeven4</td>
<td>1</td>
<td>8</td>
<td>867</td>
<td>12.7</td>
<td>22</td>
<td>46.0</td>
<td>✓</td>
</tr>
<tr>
<td>oddeven5</td>
<td>1</td>
<td>10</td>
<td>2334</td>
<td>56.8</td>
<td>52</td>
<td>1319.4</td>
<td>✓</td>
</tr>
<tr>
<td>bubble3</td>
<td>1</td>
<td>6</td>
<td>249</td>
<td>4.1</td>
<td>8</td>
<td>4.9</td>
<td>✓</td>
</tr>
<tr>
<td>bubble4</td>
<td>1</td>
<td>8</td>
<td>832</td>
<td>11.7</td>
<td>22</td>
<td>47.6</td>
<td>✓</td>
</tr>
<tr>
<td>bubble5</td>
<td>1</td>
<td>10</td>
<td>2198</td>
<td>53.9</td>
<td>52</td>
<td>938.2</td>
<td>✓</td>
</tr>
<tr>
<td>partd3</td>
<td>4</td>
<td>5</td>
<td>479</td>
<td>10.5</td>
<td>10</td>
<td>50.8</td>
<td>✓</td>
</tr>
<tr>
<td>partd4</td>
<td>5</td>
<td>6</td>
<td>1217</td>
<td>23.3</td>
<td>15</td>
<td>181.1</td>
<td>✓</td>
</tr>
<tr>
<td>partd5</td>
<td>6</td>
<td>7</td>
<td>2943</td>
<td>53.3</td>
<td>21</td>
<td>418.1</td>
<td>✓</td>
</tr>
<tr>
<td>parti3</td>
<td>4</td>
<td>5</td>
<td>464</td>
<td>10.3</td>
<td>10</td>
<td>45.5</td>
<td>✓</td>
</tr>
<tr>
<td>parti4</td>
<td>5</td>
<td>6</td>
<td>1148</td>
<td>22.4</td>
<td>15</td>
<td>165.1</td>
<td>✓</td>
</tr>
<tr>
<td>parti5</td>
<td>6</td>
<td>7</td>
<td>2954</td>
<td>53.6</td>
<td>21</td>
<td>405.6</td>
<td>✓</td>
</tr>
<tr>
<td><strong>total</strong></td>
<td></td>
<td></td>
<td>16055</td>
<td>317.6</td>
<td>264</td>
<td>3647.5</td>
<td><strong>14/14</strong></td>
</tr>
</tbody>
</table>
## Results for Complex Invariants

<table>
<thead>
<tr>
<th>Prog</th>
<th>Loc</th>
<th>Var</th>
<th>Gen</th>
<th>$T_{Gen}$ (secs)</th>
<th>Val</th>
<th>$T_{Val}$ (secs)</th>
<th>$kl$</th>
<th>Strength</th>
</tr>
</thead>
<tbody>
<tr>
<td>cohendv</td>
<td>2</td>
<td>6</td>
<td>152</td>
<td>26.2</td>
<td>7</td>
<td>8.2</td>
<td>14</td>
<td>✓</td>
</tr>
<tr>
<td>divbin</td>
<td>2</td>
<td>5</td>
<td>96</td>
<td>37.7</td>
<td>8</td>
<td>8.7</td>
<td>15</td>
<td>–</td>
</tr>
<tr>
<td>manna</td>
<td>1</td>
<td>5</td>
<td>49</td>
<td>19.2</td>
<td>3</td>
<td>5.6</td>
<td>2</td>
<td>✓</td>
</tr>
<tr>
<td>hard</td>
<td>2</td>
<td>6</td>
<td>107</td>
<td>14.2</td>
<td>11</td>
<td>9.2</td>
<td>4</td>
<td>–</td>
</tr>
<tr>
<td>sqrt1</td>
<td>1</td>
<td>4</td>
<td>27</td>
<td>25.3</td>
<td>3</td>
<td>4.3</td>
<td>1</td>
<td>✓</td>
</tr>
<tr>
<td>dijkstra</td>
<td>2</td>
<td>5</td>
<td>61</td>
<td>30.7</td>
<td>8</td>
<td>10.9</td>
<td>6</td>
<td>–</td>
</tr>
<tr>
<td>freire1</td>
<td>1</td>
<td>3</td>
<td>25</td>
<td>22.5</td>
<td>2</td>
<td>2.2</td>
<td>0</td>
<td>✓</td>
</tr>
<tr>
<td>freire2</td>
<td>1</td>
<td>4</td>
<td>35</td>
<td>26.0</td>
<td>3</td>
<td>5.1</td>
<td>1</td>
<td>✓</td>
</tr>
<tr>
<td>cohen cb</td>
<td>1</td>
<td>5</td>
<td>31</td>
<td>23.6</td>
<td>4</td>
<td>4.2</td>
<td>1</td>
<td>✓</td>
</tr>
<tr>
<td>egcd1</td>
<td>1</td>
<td>8</td>
<td>108</td>
<td>43.1</td>
<td>1</td>
<td>12.8</td>
<td>8</td>
<td>–</td>
</tr>
<tr>
<td>egcd2</td>
<td>2</td>
<td>10</td>
<td>209</td>
<td>60.8</td>
<td>8</td>
<td>14.6</td>
<td>12</td>
<td>✓</td>
</tr>
<tr>
<td>egcd3</td>
<td>3</td>
<td>12</td>
<td>475</td>
<td>67.0</td>
<td>14</td>
<td>23.4</td>
<td>25</td>
<td>✓</td>
</tr>
<tr>
<td>lcm1</td>
<td>3</td>
<td>6</td>
<td>203</td>
<td>38.9</td>
<td>12</td>
<td>14.2</td>
<td>0</td>
<td>✓</td>
</tr>
<tr>
<td>lcm2</td>
<td>1</td>
<td>6</td>
<td>52</td>
<td>14.9</td>
<td>1</td>
<td>0.9</td>
<td>10</td>
<td>✓</td>
</tr>
<tr>
<td>prodbin</td>
<td>1</td>
<td>5</td>
<td>61</td>
<td>28.3</td>
<td>3</td>
<td>1.1</td>
<td>10</td>
<td>–</td>
</tr>
<tr>
<td>prod4br</td>
<td>1</td>
<td>6</td>
<td>42</td>
<td>9.6</td>
<td>4</td>
<td>8.6</td>
<td>7</td>
<td>✓</td>
</tr>
<tr>
<td>fermat1</td>
<td>3</td>
<td>5</td>
<td>217</td>
<td>75.7</td>
<td>6</td>
<td>6.2</td>
<td>1</td>
<td>✓</td>
</tr>
<tr>
<td>fermat2</td>
<td>1</td>
<td>5</td>
<td>70</td>
<td>25.8</td>
<td>2</td>
<td>5.2</td>
<td>0</td>
<td>✓</td>
</tr>
<tr>
<td>knuth</td>
<td>1</td>
<td>8</td>
<td>113</td>
<td>57.1</td>
<td>4</td>
<td>24.6</td>
<td>6</td>
<td>✓</td>
</tr>
<tr>
<td>geo1</td>
<td>1</td>
<td>4</td>
<td>25</td>
<td>16.7</td>
<td>2</td>
<td>1.5</td>
<td>4</td>
<td>✓</td>
</tr>
<tr>
<td>geo2</td>
<td>1</td>
<td>4</td>
<td>45</td>
<td>24.1</td>
<td>1</td>
<td>2.1</td>
<td>10</td>
<td>✓</td>
</tr>
<tr>
<td>geo3</td>
<td>1</td>
<td>5</td>
<td>65</td>
<td>22.1</td>
<td>1</td>
<td>2.7</td>
<td>12</td>
<td>✓</td>
</tr>
<tr>
<td>ps2</td>
<td>1</td>
<td>3</td>
<td>25</td>
<td>21.1</td>
<td>2</td>
<td>4.0</td>
<td>0</td>
<td>✓</td>
</tr>
<tr>
<td>ps3</td>
<td>1</td>
<td>3</td>
<td>25</td>
<td>21.9</td>
<td>2</td>
<td>4.2</td>
<td>0</td>
<td>✓</td>
</tr>
<tr>
<td>ps4</td>
<td>1</td>
<td>3</td>
<td>25</td>
<td>23.5</td>
<td>2</td>
<td>4.9</td>
<td>0</td>
<td>✓</td>
</tr>
<tr>
<td>ps5</td>
<td>1</td>
<td>3</td>
<td>24</td>
<td>24.9</td>
<td>2</td>
<td>7.4</td>
<td>0</td>
<td>✓</td>
</tr>
<tr>
<td>ps6</td>
<td>1</td>
<td>3</td>
<td>25</td>
<td>25.0</td>
<td>2</td>
<td>69.5</td>
<td>0</td>
<td>✓</td>
</tr>
</tbody>
</table>

**Total** | 2392 | 825.9 | 118 | 149 | 266.3 | 22/27
## Results for Complex Invariants

<table>
<thead>
<tr>
<th>Prog</th>
<th>Loc</th>
<th>Var</th>
<th>Gen</th>
<th>$T_{Gen}$ (secs)</th>
<th>Val</th>
<th>$T_{Val}$ (secs)</th>
<th>$kI$</th>
<th>Strength</th>
</tr>
</thead>
<tbody>
<tr>
<td>cohendv</td>
<td>2</td>
<td>6</td>
<td>152</td>
<td>26.2</td>
<td>7</td>
<td>8.2</td>
<td>14</td>
<td>✓</td>
</tr>
<tr>
<td>divbin</td>
<td>2</td>
<td>5</td>
<td>96</td>
<td>37.7</td>
<td>8</td>
<td>8.7</td>
<td>15</td>
<td>–</td>
</tr>
<tr>
<td>manna</td>
<td>1</td>
<td>5</td>
<td>49</td>
<td>19.2</td>
<td>3</td>
<td>5.6</td>
<td>2</td>
<td>✓</td>
</tr>
<tr>
<td>hard</td>
<td>2</td>
<td>6</td>
<td>107</td>
<td>14.2</td>
<td>11</td>
<td>9.2</td>
<td>4</td>
<td>–</td>
</tr>
<tr>
<td>sqrt1</td>
<td>1</td>
<td>4</td>
<td>27</td>
<td>25.3</td>
<td>3</td>
<td>4.3</td>
<td>1</td>
<td>✓</td>
</tr>
<tr>
<td>dijkstra</td>
<td>2</td>
<td>5</td>
<td>61</td>
<td>30.7</td>
<td>8</td>
<td>10.9</td>
<td>6</td>
<td>–</td>
</tr>
<tr>
<td>freire1</td>
<td>1</td>
<td>3</td>
<td>25</td>
<td>22.5</td>
<td>2</td>
<td>2.2</td>
<td>0</td>
<td>✓</td>
</tr>
<tr>
<td>freire2</td>
<td>1</td>
<td>4</td>
<td>35</td>
<td>26.0</td>
<td>3</td>
<td>5.1</td>
<td>1</td>
<td>✓</td>
</tr>
<tr>
<td>cohen cb</td>
<td>1</td>
<td>5</td>
<td>31</td>
<td>23.6</td>
<td>4</td>
<td>4.2</td>
<td>1</td>
<td>✓</td>
</tr>
<tr>
<td>egcd1</td>
<td>1</td>
<td>8</td>
<td>108</td>
<td>43.1</td>
<td>1</td>
<td>12.8</td>
<td>8</td>
<td>–</td>
</tr>
<tr>
<td>egcd2</td>
<td>2</td>
<td>10</td>
<td>209</td>
<td>60.8</td>
<td>8</td>
<td>14.6</td>
<td>12</td>
<td>✓</td>
</tr>
<tr>
<td>egcd3</td>
<td>3</td>
<td>12</td>
<td>475</td>
<td>67.0</td>
<td>14</td>
<td>23.4</td>
<td>25</td>
<td>✓</td>
</tr>
<tr>
<td>lcm1</td>
<td>3</td>
<td>6</td>
<td>203</td>
<td>38.9</td>
<td>12</td>
<td>14.2</td>
<td>0</td>
<td>✓</td>
</tr>
<tr>
<td>lcm2</td>
<td>1</td>
<td>6</td>
<td>52</td>
<td>14.9</td>
<td>1</td>
<td>0.9</td>
<td>10</td>
<td>✓</td>
</tr>
<tr>
<td>prod bin</td>
<td>1</td>
<td>5</td>
<td>61</td>
<td>28.3</td>
<td>3</td>
<td>1.1</td>
<td>10</td>
<td>–</td>
</tr>
<tr>
<td>prod4br</td>
<td>1</td>
<td>6</td>
<td>42</td>
<td>9.6</td>
<td>4</td>
<td>8.6</td>
<td>7</td>
<td>✓</td>
</tr>
<tr>
<td>fermat1</td>
<td>3</td>
<td>5</td>
<td>217</td>
<td>75.7</td>
<td>6</td>
<td>6.2</td>
<td>1</td>
<td>✓</td>
</tr>
<tr>
<td>fermat2</td>
<td>1</td>
<td>5</td>
<td>70</td>
<td>25.8</td>
<td>2</td>
<td>5.2</td>
<td>0</td>
<td>✓</td>
</tr>
<tr>
<td>knuth</td>
<td>1</td>
<td>8</td>
<td>113</td>
<td>57.1</td>
<td>4</td>
<td>24.6</td>
<td>6</td>
<td>✓</td>
</tr>
<tr>
<td>geo1</td>
<td>1</td>
<td>4</td>
<td>25</td>
<td>16.7</td>
<td>2</td>
<td>1.5</td>
<td>4</td>
<td>✓</td>
</tr>
<tr>
<td>geo2</td>
<td>1</td>
<td>4</td>
<td>45</td>
<td>24.1</td>
<td>1</td>
<td>2.1</td>
<td>10</td>
<td>✓</td>
</tr>
<tr>
<td>geo3</td>
<td>1</td>
<td>5</td>
<td>65</td>
<td>22.1</td>
<td>1</td>
<td>2.7</td>
<td>12</td>
<td>✓</td>
</tr>
<tr>
<td>ps2</td>
<td>1</td>
<td>3</td>
<td>25</td>
<td>21.1</td>
<td>2</td>
<td>4.0</td>
<td>0</td>
<td>✓</td>
</tr>
<tr>
<td>ps3</td>
<td>1</td>
<td>3</td>
<td>25</td>
<td>21.9</td>
<td>2</td>
<td>4.2</td>
<td>0</td>
<td>✓</td>
</tr>
<tr>
<td>ps4</td>
<td>1</td>
<td>3</td>
<td>25</td>
<td>23.5</td>
<td>2</td>
<td>4.9</td>
<td>0</td>
<td>✓</td>
</tr>
<tr>
<td>ps5</td>
<td>1</td>
<td>3</td>
<td>24</td>
<td>24.9</td>
<td>2</td>
<td>7.4</td>
<td>0</td>
<td>✓</td>
</tr>
<tr>
<td>ps6</td>
<td>1</td>
<td>3</td>
<td>25</td>
<td>25.0</td>
<td>2</td>
<td>69.5</td>
<td>0</td>
<td>✓</td>
</tr>
<tr>
<td><strong>total</strong></td>
<td></td>
<td></td>
<td>2392</td>
<td>825.9</td>
<td>118</td>
<td>149</td>
<td>266.3</td>
<td>22/27</td>
</tr>
</tbody>
</table>
Summary

Hybrid invariant generation:

- **DIG** employs geometric concepts for dynamic invariant inference
  - Build *max-plus polyhedra* to find disjunctive polynomial invariants
  - Combine max and min relations for expressive power
  - Introduce new classes of *weak* max/min invariants that retain expressiveness and have polynomial time complexity
- **KIP** verifies invariants using program code (k-induction, SMT solving, lemmas learning, redundancy elimination, parallelism)

Results:

- Identify strong enough invariants to prove correctness of 36/41 programs
- Produce *no spurious* results
- Take 2 *minutes* per program, on avg, to find and prove invs
Thank you for your attention!

Project (DIG + KIP) is open source and available at

http://cs.unm.edu/~tnguyen

Ask me about (or check my webpage for details)

- Dynamic analysis on complex array invariants (identified 60% of array relations in AES)
- Static analysis on octagonal and max-plus invariants using quantifier elimination
- Automatic program repair using test-input generation (equivalence theorem: program reachability $\equiv$ program synthesis)
Combining Max and Min-plus Invariants

\[ \text{def ex2 (x):} \]
\[ \text{if } x \geq 0: \]
\[ y = x + 1 \]
\[ \text{else:} \]
\[ y = x - 1 \]
\[ b = y > 10 \]
\[ \text{return } b \]

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>-50</td>
<td>-51</td>
<td>0</td>
</tr>
<tr>
<td>-33</td>
<td>-34</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>12</td>
<td>13</td>
<td>1</td>
</tr>
<tr>
<td>40</td>
<td>41</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ \text{By building max and min-plus polyhedra in 3D, DIG obtains} \]
\[ 1 \geq b \geq 0, \max(y - 10, 0) \geq b, b + 10 \geq \min(y, 11) \]
\[ \text{i.e.,} \]
\[ 1 \geq b \geq 0 \land \max(y - 10, 0) \geq b \Rightarrow b = 0 \]
\[ \Rightarrow y \leq 10 \]
\[ 1 \geq b \geq 0 \land b + 10 \geq \min(y, 11) \Rightarrow b \neq 0 \]
\[ \Rightarrow y > 10 \]

\[ \text{Logically equivalent to the} \]
\[ b = 0 \iff y \leq 10 \]
Combining Max and Min-plus Invariants

```python
def ex2(x):
    if x ≥ 0:
        y = x + 1
    else:
        y = x - 1
    b = y > 10
    return b
```

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>-50</td>
<td>-51</td>
<td>0</td>
</tr>
<tr>
<td>-33</td>
<td>-34</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>12</td>
<td>13</td>
<td>1</td>
</tr>
<tr>
<td>40</td>
<td>41</td>
<td>1</td>
</tr>
</tbody>
</table>

• By building max and min-plus polyhedra in 3D, DIG obtains

\[
1 \geq b \geq 0, \quad \max(y - 10, 0) \geq b, \quad b + 10 \geq \min(y, 11),
\]

i.e.,

\[
1 \geq b \geq 0 \land \max(y - 10, 0) \geq b \Rightarrow b = 0 \Rightarrow y \leq 10
\]

\[
1 \geq b \geq 0 \land b + 10 \geq \min(y, 11) \Rightarrow b \neq 0 \Rightarrow y > 10
\]

• Logically equivalent to the iff condition

\[
b = 0 \iff y \leq 10
\]
Combining Max and Min-plus Invariants

```python
def ex2(x):
    if x \geq 0:
        y = x + 1
    else:
        y = x - 1
    b = y > 10
    return b
```

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>-50</td>
<td>-51</td>
<td>0</td>
</tr>
<tr>
<td>-33</td>
<td>-34</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>12</td>
<td>13</td>
<td>1</td>
</tr>
<tr>
<td>40</td>
<td>41</td>
<td>1</td>
</tr>
</tbody>
</table>

• By building max and min-plus polyhedra in 3D, DIG obtains

\[
1 \geq b \geq 0, \quad \max(y - 10, 0) \geq b, \quad b + 10 \geq \min(y, 11),
\]
i.e.,

\[
1 \geq b \geq 0 \land \max(y - 10, 0) \geq b \implies b = 0 \implies y \leq 10 \\
1 \geq b \geq 0 \land b + 10 \geq \min(y, 11) \implies b \neq 0 \implies y > 10
\]
Combining Max and Min-plus Invariants

```python
def ex2(x):
    if x ≥ 0:
        y = x + 1
    else:
        y = x - 1
    b = y > 10
    return b
```

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>-50</td>
<td>-51</td>
<td>0</td>
</tr>
<tr>
<td>-33</td>
<td>-34</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>12</td>
<td>13</td>
<td>1</td>
</tr>
<tr>
<td>40</td>
<td>41</td>
<td>1</td>
</tr>
</tbody>
</table>

- By building max and min-plus polyhedra in 3D, DIG obtains
  \[ 1 \geq b \geq 0, \quad \max(y - 10, 0) \geq b, \quad b + 10 \geq \min(y, 11), \]
  i.e.,
  \[ 1 \geq b \geq 0 \land \max(y - 10, 0) \geq b \Rightarrow b = 0 \Rightarrow y \leq 10 \]
  \[ 1 \geq b \geq 0 \land b + 10 \geq \min(y, 11) \Rightarrow b \neq 0 \Rightarrow y > 10 \]
- Logically equivalent to the iff condition
  \[ b = 0 \iff y \leq 10 \]