Automatic Complexity Analysis of Integer Programs via Triangular Weakly Non-Linear Loops

11th International Joint Conference on Automated Reasoning

Nils Lommen, Eleanore Meyer, and Jürgen Giesl

Motivation

Goal: Infer (upper) runtime bounds for “real-world“ programs

$$\mathtt{while}\ (\mathtt{x}_1^2 < \mathtt{x}_2 \land \mathtt{x}_1 > 0)\ \mathtt{do}$$
$$\begin{bmatrix} \mathtt{x}_1 \\ \mathtt{x}_2 \\ \mathtt{x}_3 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot \mathtt{x}_1 + \mathtt{x}_3^2 \\ 3 \cdot \mathtt{x}_2 \\ \mathtt{x}_3 \end{bmatrix}$$
$$\mathtt{end}$$

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Motivation

Goal: Infer (upper) runtime bounds for “real-world“ programs

$$\texttt{while}\ (\mathtt{x}_1^2 < \mathtt{x}_2 \wedge \mathtt{x}_1 > 0)\ \texttt{do}$$
$$\begin{bmatrix} \mathtt{x}_1 \\ \mathtt{x}_2 \\ \mathtt{x}_3 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot \mathtt{x}_1 + \mathtt{x}_3^2 \\ 3 \cdot \mathtt{x}_2 \\ \mathtt{x}_3 \end{bmatrix}$$
$$\texttt{end}$$

- Does this loop terminate?

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Motivation

Goal: Infer (upper) runtime bounds for "real-world" programs

$$\texttt{while}\ (\mathtt{x}_1^2 < \mathtt{x}_2 \land \mathtt{x}_1 > \mathtt{0})\ \texttt{do}$$
$$\begin{bmatrix}\mathtt{x}_1\\ \mathtt{x}_2\\ \mathtt{x}_3\end{bmatrix} \leftarrow \begin{bmatrix}2 \cdot \mathtt{x}_1 + \mathtt{x}_3^2\\ 3 \cdot \mathtt{x}_2\\ \mathtt{x}_3\end{bmatrix}$$
$$\texttt{end}$$

- Does this loop terminate?
- How often do we execute the loop?

IJCAR 2022
Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Motivation

Goal: Infer (upper) runtime bounds for “real-world“ programs

$$\mathtt{while}\ (\mathtt{x}_1^2 < \mathtt{x}_2 \land \mathtt{x}_1 > 0)\ \mathtt{do}$$
$$\begin{bmatrix} \mathtt{x}_1 \\ \mathtt{x}_2 \\ \mathtt{x}_3 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot \mathtt{x}_1 + \mathtt{x}_3^2 \\ 3 \cdot \mathtt{x}_2 \\ \mathtt{x}_3 \end{bmatrix}$$
$$\mathtt{end}$$

- Does this loop terminate?
- How often do we execute the loop?
  - Linear ranking functions fail.

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Motivation

Goal: Infer (upper) runtime bounds for "real-world" programs

$$\texttt{while}\ (x_1^2 < x_2 \land x_1 > 0)\ \texttt{do}$$
$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot x_1 + x_3^2 \\ 3 \cdot x_2 \\ x_3 \end{bmatrix}$$
$$\texttt{end}$$

- Does this loop terminate?
- How often do we execute the loop?
  - Linear ranking functions fail.
  - Existing tools usually fail with non-linear arithmetic

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Motivation

Goal: Infer (upper) runtime bounds for “real-world“ programs

$$\texttt{while } (\mathtt{x}_1^2 < \mathtt{x}_2 \land \mathtt{x}_1 > 0) \texttt{ do}$$
$$\begin{bmatrix} \mathtt{x}_1 \\ \mathtt{x}_2 \\ \mathtt{x}_3 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot \mathtt{x}_1 + \mathtt{x}_3^2 \\ 3 \cdot \mathtt{x}_2 \\ \mathtt{x}_3 \end{bmatrix}$$
$$\texttt{end}$$

- Does this loop terminate?
- How often do we execute the loop?
  - Linear ranking functions fail.
  - Existing tools usually fail with non-linear arithmetic
  - Can compute non-linear runtime bounds for twn-loops.

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Motivation

Goal: Infer (upper) runtime bounds for “real-world“ programs

$$\texttt{while}\ (x_1^2 < x_2 \land x_1 > 0)\ \texttt{do}$$
$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot x_1 + x_3^2 \\ 3 \cdot x_2 \\ x_3 \end{bmatrix}$$
$$\texttt{end}$$

- Does this loop terminate?
- How often do we execute the loop?
  - Linear ranking functions fail.
  - Existing tools usually fail with non-linear arithmetic
  - Can compute non-linear runtime bounds for twn-loops.

- Combine [TOPLAS ’16] and [SAS ’20; LPAR ’20] in automatic complexity analysis tool KoAT

Motivation

Goal: Infer (upper) runtime bounds for “real-world“ programs

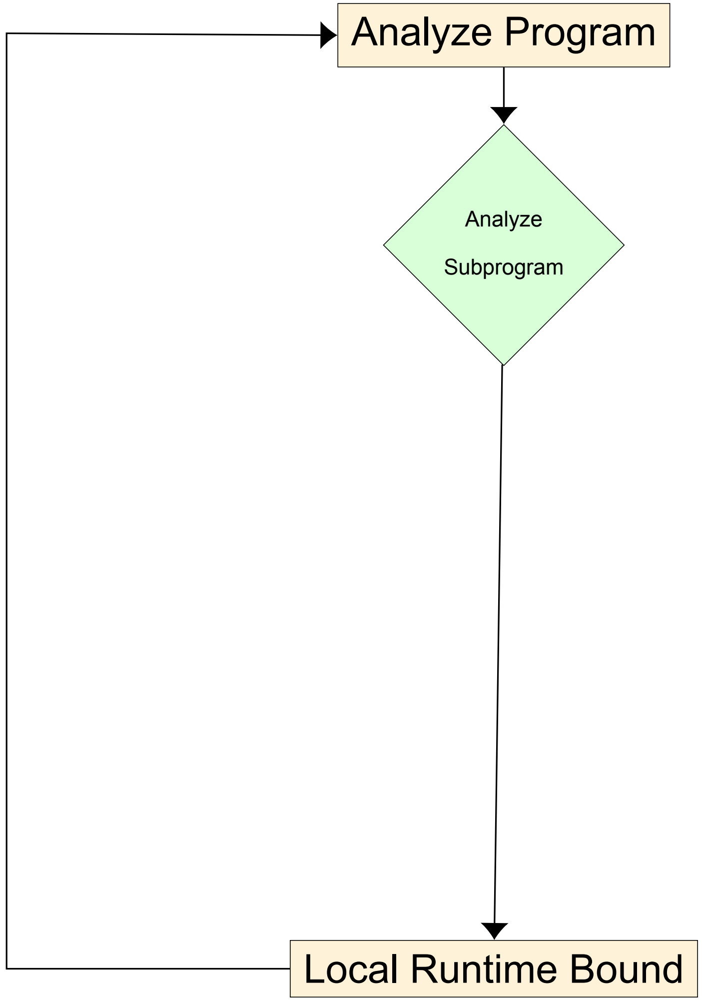
$$
\begin{array}{l}
\mathtt{while}\ (\mathtt{x}_1^2 < \mathtt{x}_2 \land \mathtt{x}_1 > 0)\ \mathtt{do} \\
\qquad \begin{bmatrix} \mathtt{x}_1 \\ \mathtt{x}_2 \\ \mathtt{x}_3 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot \mathtt{x}_1 + \mathtt{x}_3^2 \\ 3 \cdot \mathtt{x}_2 \\ \mathtt{x}_3 \end{bmatrix} \\
\mathtt{end}
\end{array}
$$

- Does this loop terminate?
- How often do we execute the loop?
  - Linear ranking functions fail.
  - Existing tools usually fail with non-linear arithmetic
  - Can compute non-linear runtime bounds for twn-loops.

- Combine [TOPLAS ’16] and [SAS ’20; LPAR ’20] in automatic complexity analysis tool `KoAT`
- Approach is complete for all terminating twn-loops

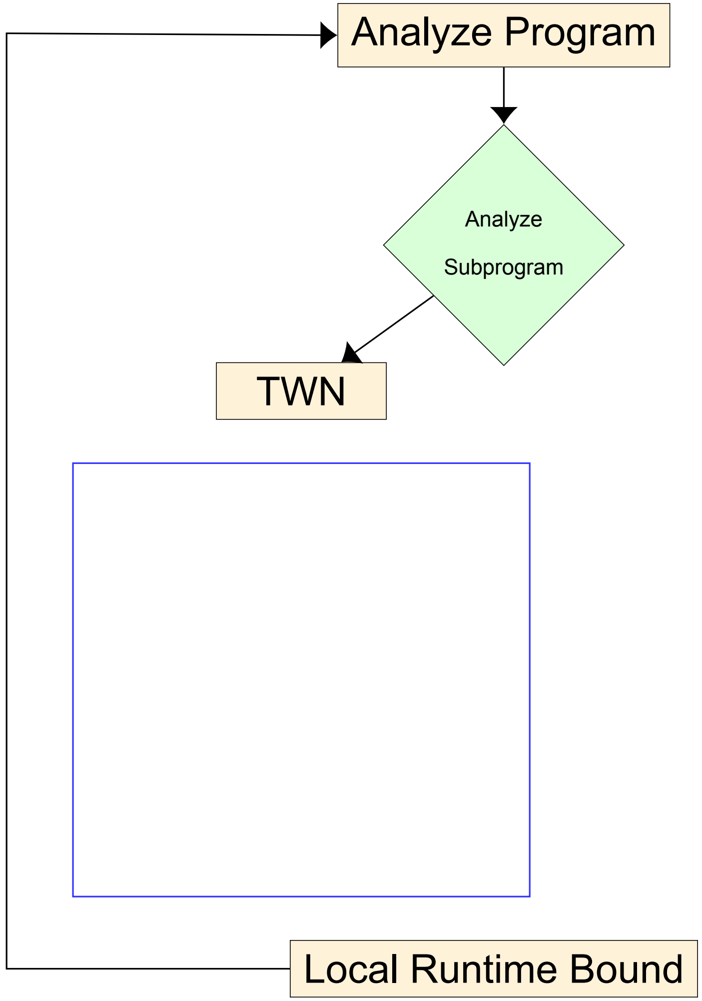
Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Overview



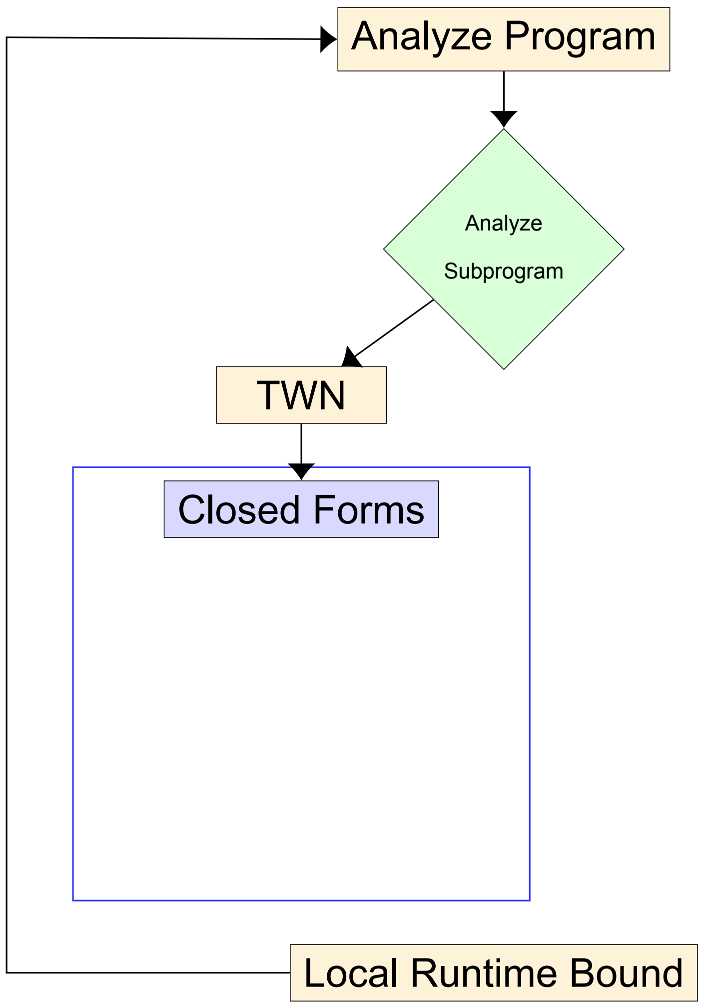
Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Overview



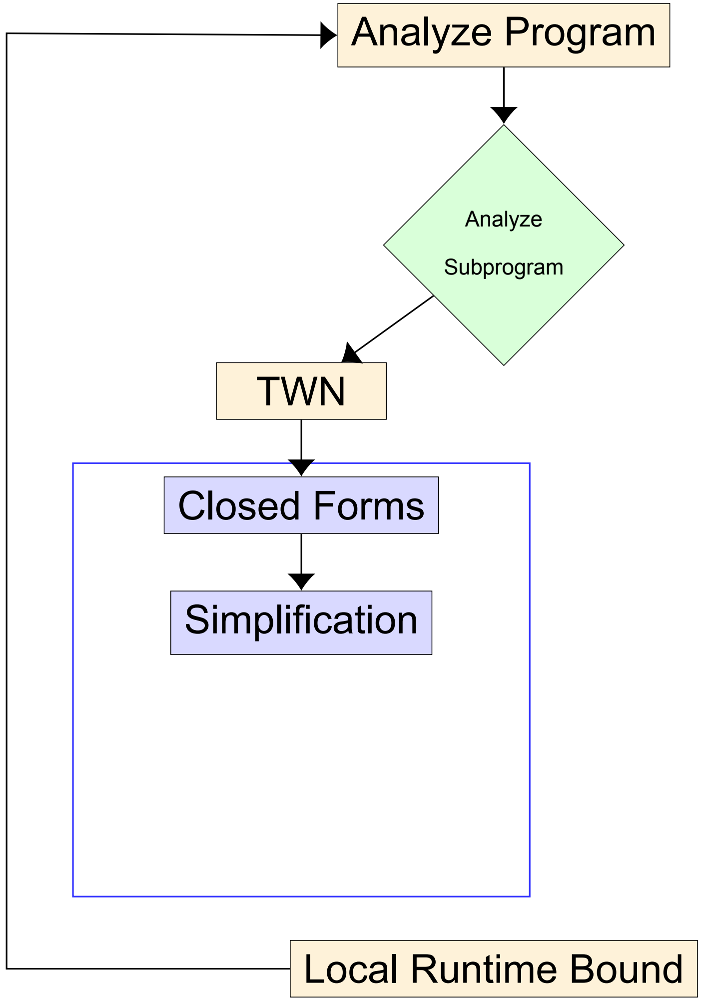
Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Overview



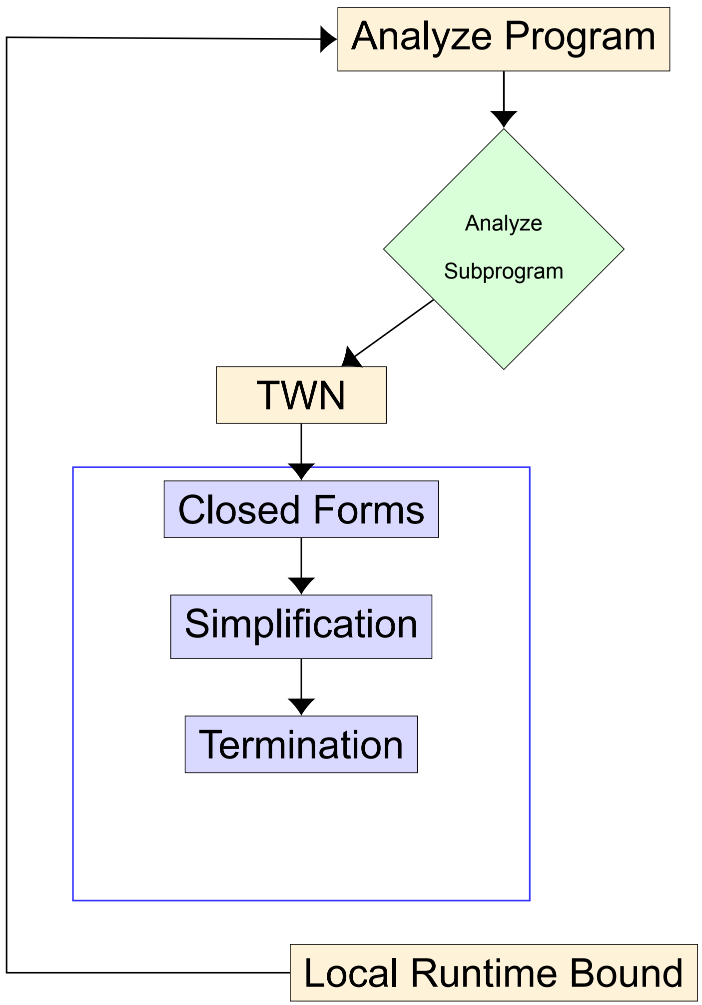
Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Overview



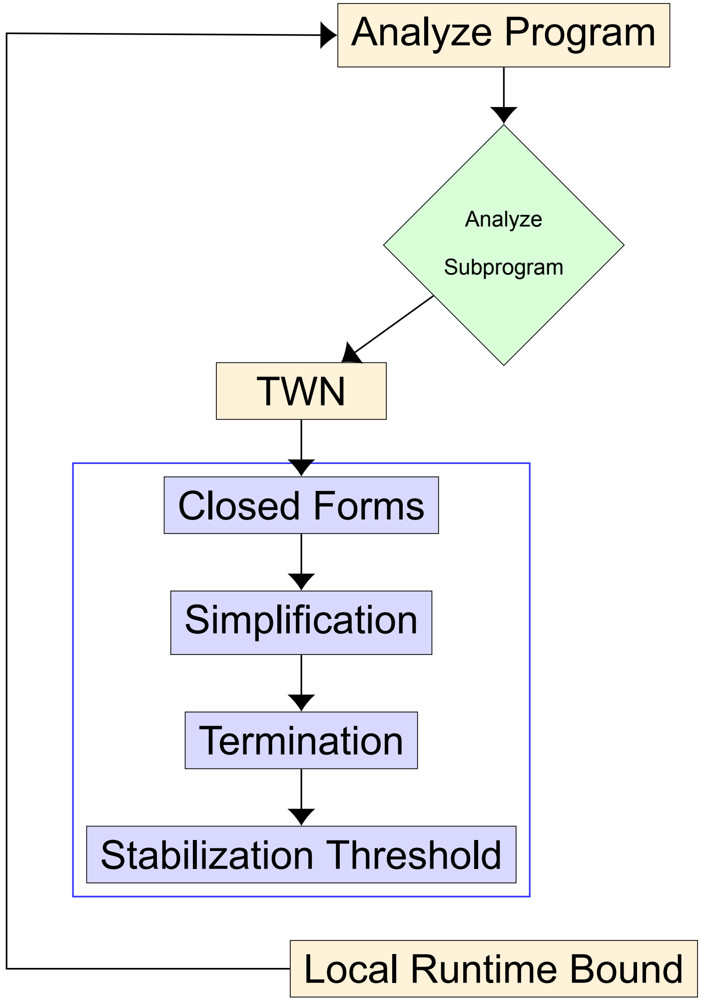
Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Overview



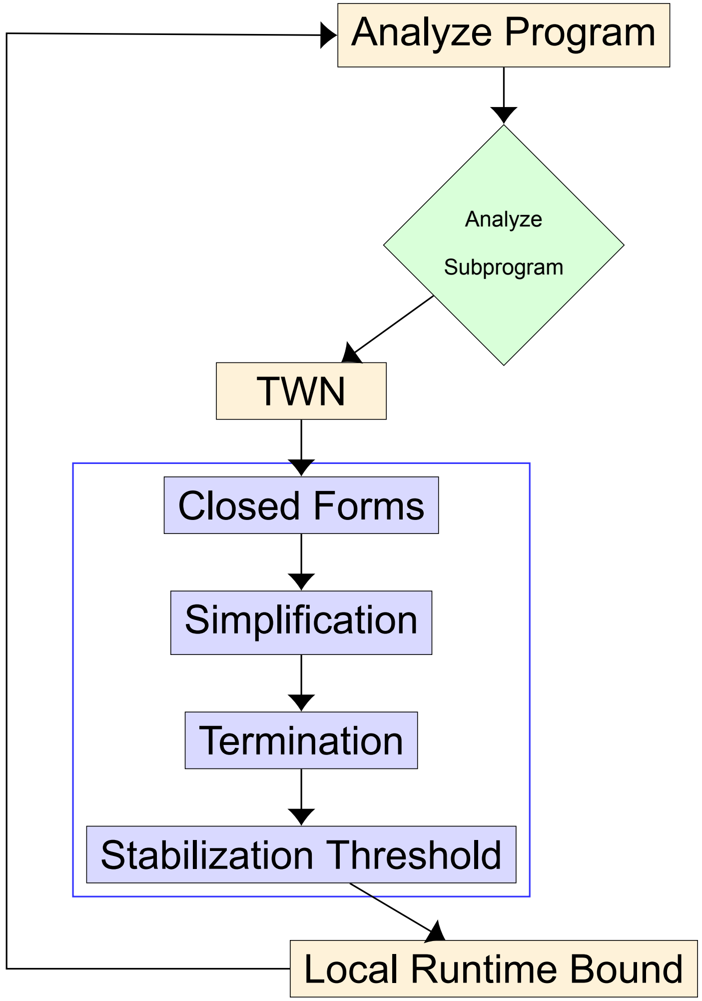
Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Overview



Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Overview



Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

TWN-Loops

```
while (τ) do
```

$$\begin{bmatrix} x_1 \\ \dots \\ x_d \end{bmatrix} \leftarrow \begin{bmatrix} c_1 \cdot x_1 + p_1 \\ \dots \\ c_d \cdot x_d + p_d \end{bmatrix}$$

```
end
```

- τ built from $\wedge$, $\vee$, ($\neg$, ...) and polynomial inequations over $\mathbb{Z}$

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

TWN-Loops

$$\texttt{while } (\tau) \texttt{ do} \quad \begin{bmatrix} x_1 \\ \dots \\ x_d \end{bmatrix} \leftarrow \begin{bmatrix} c_1 \cdot x_1 + p_1 \\ \dots \\ c_d \cdot x_d + p_d \end{bmatrix} \quad \texttt{end}$$

- τ built from $\wedge$, $\vee$, ($\neg$, …) and polynomial inequations over $\mathbb{Z}$
- $c_1, \dots, c_d \in \mathbb{Z}$

IJCAR 2022
Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

TWN-Loops

```
while (τ) do
```

$$\begin{bmatrix} x_1 \\ \dots \\ x_d \end{bmatrix} \leftarrow \begin{bmatrix} c_1 \cdot x_1 + p_1 \\ \dots \\ c_d \cdot x_d + p_d \end{bmatrix}$$

```
end
```

- τ built from $\wedge$, $\vee$, ($\neg$, …) and polynomial inequations over $\mathbb{Z}$
- $c_1, \dots, c_d \in \mathbb{Z}$

- Variable value depends at most linearly on its previous value.

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

TWN-Loops

```
while (τ) do
```
$$\begin{bmatrix} x_1 \\ \dots \\ x_d \end{bmatrix} \leftarrow \begin{bmatrix} c_1 \cdot x_1 + p_1 \\ \dots \\ c_d \cdot x_d + p_d \end{bmatrix}$$
```
end
```

- τ built from $\land$, $\lor$, ($\neg$, ...) and polynomial inequations over $\mathbb{Z}$
- $c_1, \dots, c_d \in \mathbb{Z}$

- Variable value depends at most linearly on its previous value.
  - Prevent super-exponential growth: $\mathtt{x} \leftarrow \mathtt{x}^2$ (so the value is $2^{(2^i)} \cdot e$)

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

TWN-Loops

```
while (τ) do
```

$$\begin{bmatrix} x_1 \\ \dots \\ x_d \end{bmatrix} \leftarrow \begin{bmatrix} c_1 \cdot x_1 + p_1 \\ \dots \\ c_d \cdot x_d + p_d \end{bmatrix}$$

```
end
```

- τ built from $\wedge$, $\vee$, ($\neg$, ...) and polynomial inequations over $\mathbb{Z}$
- $c_1, \dots, c_d \in \mathbb{Z}$
- $p_i \in \mathbb{Z}[x_{i+1}, \dots, x_d]$ non-linear

- Variable value depends at most linearly on its previous value.
  - Prevent super-exponential growth: $\mathtt{x} \leftarrow \mathtt{x}^2$ (so the value is $2^{(2^i)} \cdot e$)

IJCAR 2022
Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

TWN-Loops

```
while (τ) do
```

$$\begin{bmatrix} x_1 \\ \dots \\ x_d \end{bmatrix} \leftarrow \begin{bmatrix} c_1 \cdot x_1 + p_1 \\ \dots \\ c_d \cdot x_d + p_d \end{bmatrix}$$

```
end
```

- τ built from $\wedge$, $\vee$, ($\neg$, ...) and polynomial inequations over $\mathbb{Z}$
- $c_1, \dots, c_d \in \mathbb{Z}$
- $p_i \in \mathbb{Z}[x_{i+1}, \dots, x_d]$ non-linear

- Variable value depends at most linearly on its previous value.
  - Prevent super-exponential growth: $\mathtt{x} \leftarrow \mathtt{x}^2$ (so the value is $2^{(2^i)} \cdot e$)
- Polynomial dependencies only of variables with higher index

IJCAR 2022
Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

TWN-Loops

```
while (τ) do
```
$$\begin{bmatrix} x_1 \\ \dots \\ x_d \end{bmatrix} \leftarrow \begin{bmatrix} c_1 \cdot x_1 + p_1 \\ \dots \\ c_d \cdot x_d + p_d \end{bmatrix}$$
```
end
```

- τ built from $\wedge$, $\vee$, ($\neg$, ...) and polynomial inequations over $\mathbb{Z}$
- $c_1, \dots, c_d \in \mathbb{Z}$
- $p_i \in \mathbb{Z}[x_{i+1}, \dots, x_d]$ non-linear

- Variable value depends at most linearly on its previous value.
  - Prevent super-exponential growth: $\mathtt{x} \leftarrow \mathtt{x}^2$ (so the value is $2^{(2^i)} \cdot e$)
- Polynomial dependencies only of variables with higher index
  - No cyclic dependencies: $\mathtt{x}_1 \leftarrow \mathtt{x}_2$ and $\mathtt{x}_2 \leftarrow \mathtt{x}_1$

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

TWN-Loops

$$\texttt{while}\ (x_1^2 < x_2 \wedge x_1 > 0)\ \texttt{do}$$
$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot x_1 + x_3^2 \\ 3 \cdot x_2 \\ x_3 \end{bmatrix}$$
$$\texttt{end}$$

- τ built from $\wedge$, $\vee$, ($\neg$, ...) and polynomial inequations over $\mathbb{Z}$
- $c_1, \ldots, c_d \in \mathbb{Z}$
- $p_i \in \mathbb{Z}[x_{i+1}, \ldots, x_d]$ non-linear

- Variable value depends at most linearly on its previous value.
  - Prevent super-exponential growth: $x \leftarrow x^2$ (so the value is $2^{(2^i)} \cdot e$)
- Polynomial dependencies only of variables with higher index
  - No cyclic dependencies: $x_1 \leftarrow x_2$ and $x_2 \leftarrow x_1$

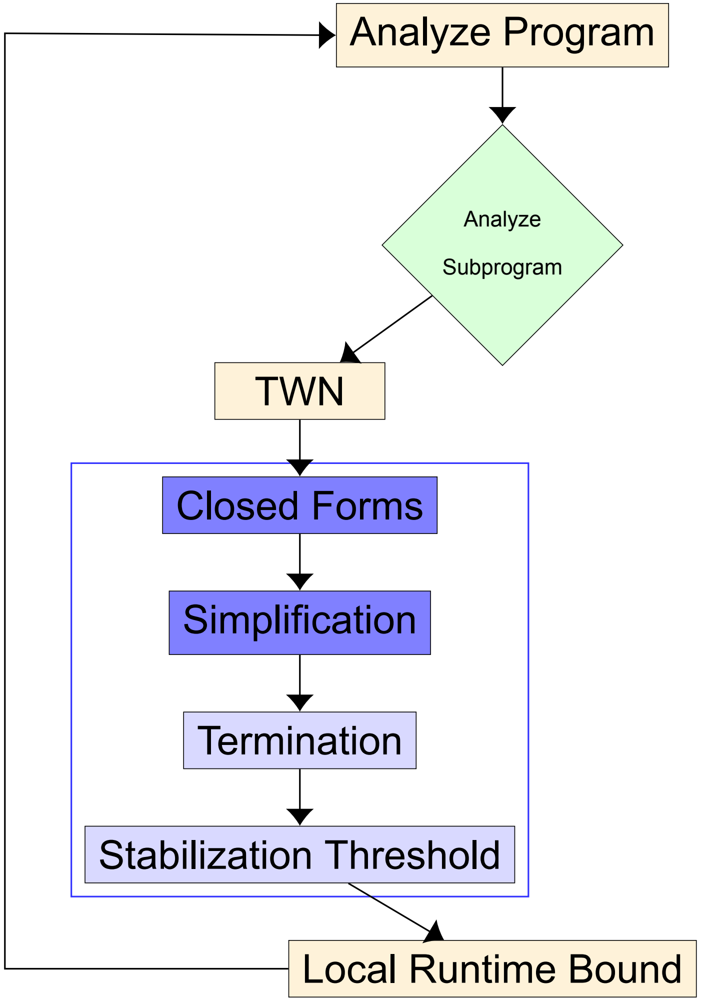
TWN-Loops

$$\texttt{while } (x_1^2 < x_2 \wedge x_1 > 0) \texttt{ do}$$
$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot x_1 \\ 3 \cdot x_2 \\ 1 \cdot x_3 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3^2 \end{bmatrix}$$
$$\texttt{end}$$

- τ built from $\wedge$, $\vee$, ($\neg$, …) and polynomial inequations over $\mathbb{Z}$
- $c_1, \ldots, c_d \in \mathbb{Z}$
- $p_i \in \mathbb{Z}[x_{i+1}, \ldots, x_d]$ non-linear

- Variable value depends at most linearly on its previous value.
  - Prevent super-exponential growth: $x \leftarrow x^2$ (so the value is $2^{(2^i)} \cdot e$)
- Polynomial dependencies only of variables with higher index
  - No cyclic dependencies: $x_1 \leftarrow x_2$ and $x_2 \leftarrow x_1$

Overview



Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Closed Forms & Simplification

Goal: Infer closed forms

```
while (x_1^2 < x_2 ∧ x_1 > 0) do
```

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot x_1 + x_3^2 \\ 3 \cdot x_2 \\ x_3 \end{bmatrix}$$

```
end
```


Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Closed Forms & Simplification

Goal: Infer closed forms

$$\texttt{while}\ (\mathtt{x}_1^2 < \mathtt{x}_2 \land \mathtt{x}_1 > 0)\ \texttt{do}$$
$$\begin{bmatrix} \mathtt{x}_1 \\ \mathtt{x}_2 \\ \mathtt{x}_3 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot \mathtt{x}_1 + \mathtt{x}_3^2 \\ 3 \cdot \mathtt{x}_2 \\ \mathtt{x}_3 \end{bmatrix}$$
$$\texttt{end}$$

- Closed forms after i iterations w.r.t. initial values e_1, e_2 and e_3 :

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Closed Forms & Simplification

Goal: Infer closed forms

$$\texttt{while}\ (\mathtt{x}_1^2 < \mathtt{x}_2 \land \mathtt{x}_1 > 0)\ \texttt{do}$$
$$\begin{bmatrix}\mathtt{x}_1\\ \mathtt{x}_2\\ \mathtt{x}_3\end{bmatrix} \leftarrow \begin{bmatrix}2\cdot \mathtt{x}_1 + \mathtt{x}_3^2\\ 3\cdot \mathtt{x}_2\\ \mathtt{x}_3\end{bmatrix}$$
$$\texttt{end}$$

- Closed forms after i iterations w.r.t. initial values e_1, e_2 and e_3 :
  - Value of $\mathtt{x}_1$: $2^i \cdot (e_1 + e_3^2) - e_3^2$

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Closed Forms & Simplification

Goal: Infer closed forms

```
while (x_1^2 < x_2 ∧ x_1 > 0) do
    [x_1, x_2, x_3] ← [2·x_1 + x_3^2, 3·x_2, x_3]
end
```

$$\texttt{while}\ (\mathtt{x}_1^2 < \mathtt{x}_2 \land \mathtt{x}_1 > 0)\ \texttt{do}$$
$$\begin{bmatrix} \mathtt{x}_1 \\ \mathtt{x}_2 \\ \mathtt{x}_3 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot \mathtt{x}_1 + \mathtt{x}_3^2 \\ 3 \cdot \mathtt{x}_2 \\ \mathtt{x}_3 \end{bmatrix}$$
$$\texttt{end}$$

- Closed forms after i iterations w.r.t. initial values e_1, e_2 and e_3 :
  - Value of $\mathtt{x}_1^2$: $(2^i \cdot (e_1 + e_3^2) - e_3^2)^2$

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Closed Forms & Simplification

Goal: Infer closed forms and remove non-linear parts to reduce degree

$$\mathtt{while}\ (\mathtt{x}_1^2 < \mathtt{x}_2 \land \mathtt{x}_1 > 0)\ \mathtt{do}$$
$$\begin{bmatrix} \mathtt{x}_1 \\ \mathtt{x}_2 \\ \mathtt{x}_3 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot \mathtt{x}_1 + \mathtt{x}_3^2 \\ 3 \cdot \mathtt{x}_2 \\ \mathtt{x}_3 \end{bmatrix}$$
$$\mathtt{end}$$

- Closed forms after i iterations w.r.t. initial values e_1, e_2 and e_3 :
  - Value of $\mathtt{x}_1^2$: $(2^i \cdot (e_1 + e_3^2) - e_3^2)^2$

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Closed Forms & Simplification

Goal: Infer closed forms and remove non-linear parts to reduce degree

$$\texttt{while } (x_1^2 < x_2 \land x_1 > 0) \texttt{ do}$$
$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot x_1 + x_3^2 \\ 3 \cdot x_2 \\ x_3 \end{bmatrix}$$
$$\texttt{end}$$

- Value of x_3^2 always non-negative

- Closed forms after i iterations w.r.t. initial values e_1, e_2 and e_3 :
  - Value of x_1^2: $(2^i \cdot (e_1 + e_3^2) - e_3^2)^2$

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Closed Forms & Simplification

Goal: Infer closed forms and remove non-linear parts to reduce degree

```
while (x_1^2 < x_2 ∧ x_1 > 0) do
```
$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot x_1 + x_3^2 \\ 3 \cdot x_2 \\ x_3 \end{bmatrix}$$
```
end
```

- Value of x_3^2 always non-negative
  - Removing x_3^2 increases runtime

- Closed forms after i iterations w.r.t. initial values e_1, e_2 and e_3 :
  - Value of x_1^2: $(2^i \cdot (e_1 + e_3^2) - e_3^2)^2$

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Closed Forms & Simplification

Goal: Infer closed forms and remove non-linear parts to reduce degree

$$\texttt{while } (\mathtt{x}_1^2 < \mathtt{x}_2 \wedge \mathtt{x}_1 > 0) \texttt{ do}$$
$$\begin{bmatrix} \mathtt{x}_1 \\ \mathtt{x}_2 \\ \mathtt{x}_3 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot \mathtt{x}_1 \\ 3 \cdot \mathtt{x}_2 \\ \mathtt{x}_3 \end{bmatrix}$$
$$\texttt{end}$$

- Value of $\mathtt{x}_3^2$ always non-negative
  - Removing $\mathtt{x}_3^2$ increases runtime

- Closed forms after i iterations w.r.t. initial values e_1, e_2 and e_3 :
  - Value of $\mathtt{x}_1^2$: $(2^i \cdot (e_1 + e_3^2) - e_3^2)^2$

IJCAR 2022
Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Closed Forms & Simplification

Goal: Infer closed forms and remove non-linear parts to reduce degree

```
while (x_1^2 < x_2 ∧ x_1 > 0) do
    [x_1; x_2] ← [2 · x_1; 3 · x_2]
end
```

$$\texttt{while}\ (\mathtt{x}_1^2 < \mathtt{x}_2 \land \mathtt{x}_1 > 0)\ \texttt{do} \quad \begin{bmatrix}\mathtt{x}_1\\ \mathtt{x}_2\end{bmatrix} \leftarrow \begin{bmatrix}2 \cdot \mathtt{x}_1\\ 3 \cdot \mathtt{x}_2\end{bmatrix} \quad \texttt{end}$$

- Value of $\mathtt{x}_3^2$ always non-negative
  - Removing $\mathtt{x}_3^2$ increases runtime

- Closed forms after i iterations w.r.t. initial values e_1, e_2 and e_3 :
  - Value of $\mathtt{x}_1^2$: $(2^i \cdot (e_1 + e_3^2) - e_3^2)^2$

Closed Forms & Simplification

Goal: Infer closed forms and remove non-linear parts to reduce degree

$$\texttt{while}\ (x_1^2 < x_2 \land x_1 > 0)\ \texttt{do}$$
$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot x_1 \\ 3 \cdot x_2 \end{bmatrix}$$
$$\texttt{end}$$

- Value of x_3^2 always non-negative
  - Removing x_3^2 increases runtime

- Closed forms after i iterations w.r.t. initial values e_1 and e_2:
  - Value of x_1: $2^i \cdot e_1$

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Closed Forms & Simplification

Goal: Infer closed forms and remove non-linear parts to reduce degree

$$\texttt{while}\ (\mathtt{x}_1^2 < \mathtt{x}_2 \land \mathtt{x}_1 > 0)\ \texttt{do}$$
$$\begin{bmatrix}\mathtt{x}_1\\ \mathtt{x}_2\end{bmatrix} \leftarrow \begin{bmatrix}2\cdot \mathtt{x}_1\\ 3\cdot \mathtt{x}_2\end{bmatrix}$$
$$\texttt{end}$$

- Value of $\mathtt{x}_3^2$ always non-negative
  - Removing $\mathtt{x}_3^2$ increases runtime

- Closed forms after i iterations w.r.t. initial values e_1 and e_2:
  - Value of $\mathtt{x}_1^2$: $(2^i \cdot e_1)^2 = 4^i \cdot e_1^2$

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Closed Forms & Simplification

Goal: Infer closed forms and remove non-linear parts to reduce degree

```
while (x₁² < x₂ ∧ x₁ > 0) do
    [x₁]    [2 · x₁]
    [x₂] ← [3 · x₂]
end
```

- Value of $\mathtt{x}_3^2$ always non-negative
  - Removing $\mathtt{x}_3^2$ increases runtime

- Closed forms after i iterations w.r.t. initial values e_1 and e_2:
  - Value of $\mathtt{x}_1^2$: $(2^i \cdot e_1)^2 = 4^i \cdot e_1^2$
  - Value of $\mathtt{x}_2$: $3^i \cdot e_2$

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Closed Forms & Simplification

Goal: Infer closed forms and remove non-linear parts to reduce degree

```
while (x_1^2 < x_2 ∧ x_1 > 0) do
    [x_1; x_2] ← [2 · x_1; 3 · x_2]
end
```

$$\texttt{while } (\mathtt{x}_1^2 < \mathtt{x}_2 \land \mathtt{x}_1 > 0) \texttt{ do} \quad \begin{bmatrix} \mathtt{x}_1 \\ \mathtt{x}_2 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot \mathtt{x}_1 \\ 3 \cdot \mathtt{x}_2 \end{bmatrix} \quad \texttt{end}$$

- Value of $\mathtt{x}_3^2$ always non-negative
  - Removing $\mathtt{x}_3^2$ increases runtime
- Eliminated non-linear occurrence of $\mathtt{x}_3$ in closed forms

- Closed forms after i iterations w.r.t. initial values e_1 and e_2:
  - Value of $\mathtt{x}_1^2$: $(2^i \cdot e_1)^2 = 4^i \cdot e_1^2$
  - Value of $\mathtt{x}_2$: $3^i \cdot e_2$

IJCAR 2022
Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Closed Forms & Simplification

Goal: Infer closed forms and remove non-linear parts to reduce degree

```
while (x_1^2 < x_2 ∧ x_1 > 0) do
    [x_1]    [2 · x_1]
    [x_2] ← [3 · x_2]
end
```

$$\texttt{while}\ (x_1^2 < x_2 \land x_1 > 0)\ \texttt{do}\quad \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot x_1 \\ 3 \cdot x_2 \end{bmatrix}\quad \texttt{end}$$

- Value of x_3^2 always non-negative
  - Removing x_3^2 increases runtime
- Eliminated non-linear occurrence of x_3 in closed forms
- Novel approach infers tighter bounds than [LPAR '20]

- Closed forms after i iterations w.r.t. initial values e_1 and e_2:
  - Value of x_1^2: $(2^i \cdot e_1)^2 = 4^i \cdot e_1^2$
  - Value of x_2: $3^i \cdot e_2$

Closed Forms & Simplification

Goal: Infer closed forms and remove non-linear parts to reduce degree

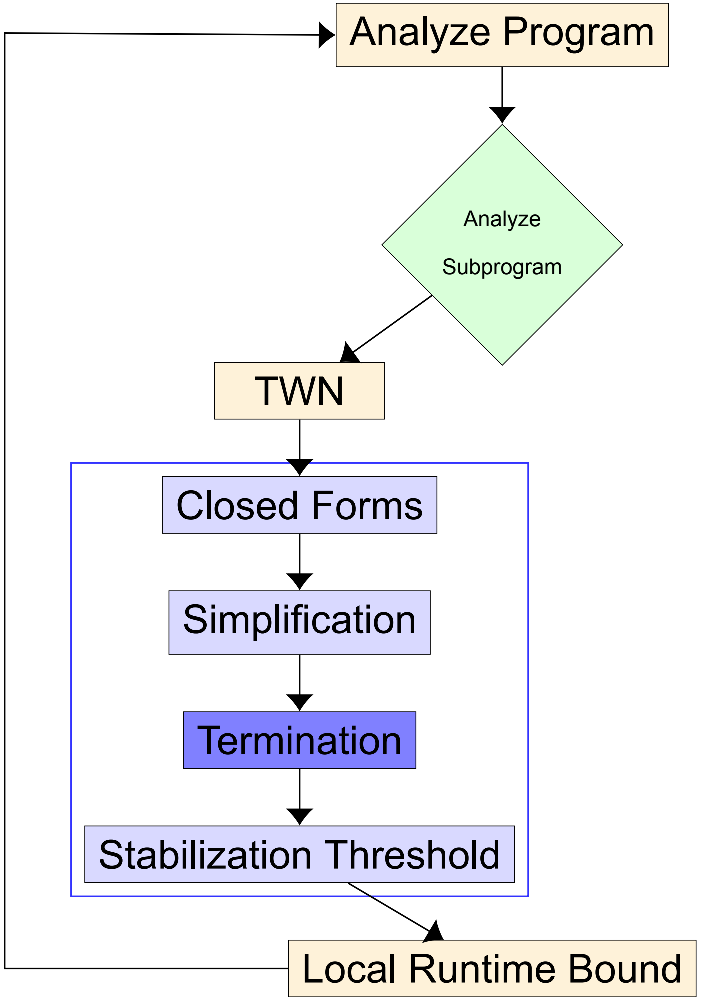
$$\texttt{while}\ (\mathtt{x}_1^2 < \mathtt{x}_2 \land \mathtt{x}_1 > 0)\ \texttt{do}$$
$$\begin{bmatrix}\mathtt{x}_1\\ \mathtt{x}_2\end{bmatrix} \leftarrow \begin{bmatrix}2\cdot \mathtt{x}_1\\ 3\cdot \mathtt{x}_2\end{bmatrix}$$
$$\texttt{end}$$

- Value of $\mathtt{x}_3^2$ always non-negative
  - Removing $\mathtt{x}_3^2$ increases runtime
- Eliminated non-linear occurrence of $\mathtt{x}_3$ in closed forms
- Novel approach infers tighter bounds than [LPAR '20]

- Closed forms after i iterations w.r.t. initial values e_1 and e_2:
  - Value of $\mathtt{x}_1^2$: $(2^i \cdot e_1)^2 = 4^i \cdot e_1^2$
  - Value of $\mathtt{x}_2$: $3^i \cdot e_2$
- KoAT automatically infers closed forms [CAV '19] and applies simplification

IJCAR 2022
Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Overview



Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Termination of TWN-Loops

Does the loop terminate?

$$\texttt{while } (\mathtt{x}_1^2 < \mathtt{x}_2 \wedge \mathtt{x}_1 > 0) \texttt{ do}$$
$$\begin{bmatrix} \mathtt{x}_1 \\ \mathtt{x}_2 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot \mathtt{x}_1 \\ 3 \cdot \mathtt{x}_2 \end{bmatrix}$$
$$\texttt{end}$$

- Yes!

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Termination of TWN-Loops

Does the loop terminate?

$$\texttt{while}\ (\texttt{x}_1^2 < \texttt{x}_2 \wedge \texttt{x}_1 > 0)\ \texttt{do}$$
$$\begin{bmatrix} \texttt{x}_1 \\ \texttt{x}_2 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot \texttt{x}_1 \\ 3 \cdot \texttt{x}_2 \end{bmatrix}$$
$$\texttt{end}$$

- Yes!
- Value of $\texttt{x}_1^2$ eventually *outgrows* value of $\texttt{x}_2$

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Termination of TWN-Loops

Does the loop terminate?

$$\texttt{while}\ (\mathtt{x}_1^2 < \mathtt{x}_2 \land \mathtt{x}_1 > 0)\ \texttt{do}$$
$$\begin{bmatrix}\mathtt{x}_1\\ \mathtt{x}_2\end{bmatrix} \leftarrow \begin{bmatrix}2\cdot\mathtt{x}_1\\ 3\cdot\mathtt{x}_2\end{bmatrix}$$
$$\texttt{end}$$

- Yes!
- Value of $\mathtt{x}_1^2$ eventually *outgrows* value of $\mathtt{x}_2$
- At some point we always have

$$4^i \cdot e_1^2 \geq 3^i \cdot e_2.$$

Termination of TWN-Loops

Does the loop terminate?

$$
\begin{array}{l}
\texttt{while}\ (\mathtt{x}_1^2 < \mathtt{x}_2 \wedge \mathtt{x}_1 > 0)\ \texttt{do} \\
\quad \begin{bmatrix} \mathtt{x}_1 \\ \mathtt{x}_2 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot \mathtt{x}_1 \\ 3 \cdot \mathtt{x}_2 \end{bmatrix} \\
\texttt{end}
\end{array}
$$

- Yes!
- Value of $\mathtt{x}_1^2$ eventually *outgrows* value of $\mathtt{x}_2$
- At some point we always have

$$4^i \cdot e_1^2 \geq 3^i \cdot e_2.$$

- Reduce Termination to an existential formula over $\mathbb{Z}$ [SAS '20]

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Termination of TWN-Loops

Does the loop terminate?

$$
\begin{array}{l}
\texttt{while}\ (\mathtt{x}_1^2 < \mathtt{x}_2 \land \mathtt{x}_1 > 0)\ \texttt{do} \\
\quad \begin{bmatrix} \mathtt{x}_1 \\ \mathtt{x}_2 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot \mathtt{x}_1 \\ 3 \cdot \mathtt{x}_2 \end{bmatrix} \\
\texttt{end}
\end{array}
$$

- Yes!
- Value of $\mathtt{x}_1^2$ eventually *outgrows* value of $\mathtt{x}_2$
- At some point we always have

$$4^i \cdot e_1^2 \geq 3^i \cdot e_2.$$

- Reduce Termination to an existential formula over $\mathbb{Z}$ [SAS '20]
  - linear arithmetic: co-NP-complete

IJCAR 2022
Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Termination of TWN-Loops

Does the loop terminate?

```
while (x₁² < x₂ ∧ x₁ > 0) do
    [x₁]    [2 · x₁]
    [x₂] ←  [3 · x₂]
end
```

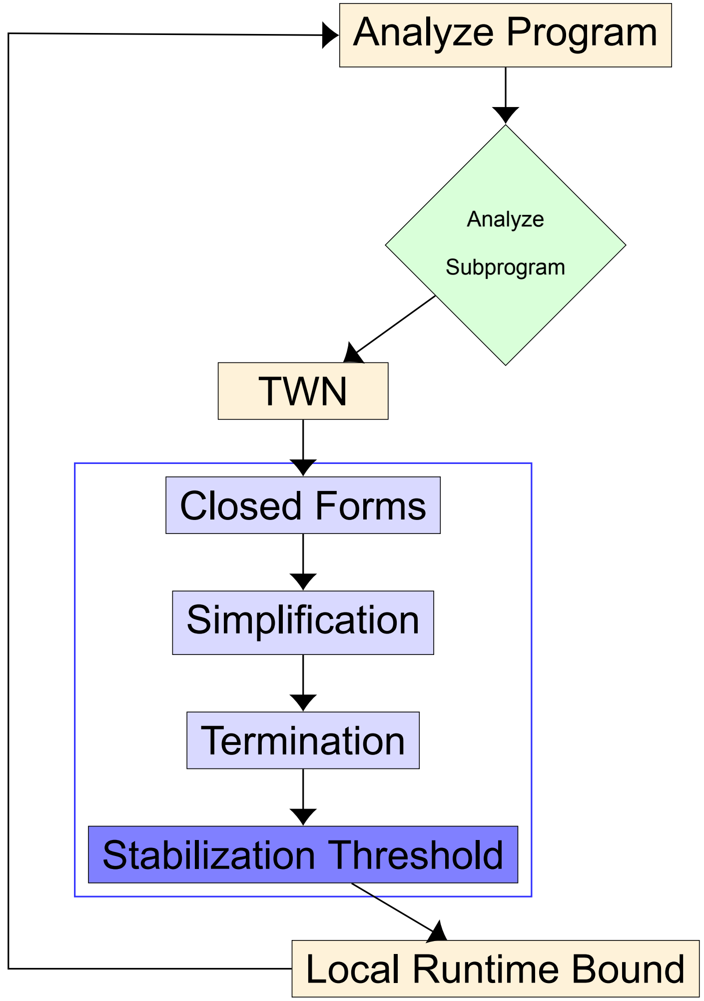
$$\texttt{while}\ (\mathtt{x}_1^2 < \mathtt{x}_2 \land \mathtt{x}_1 > 0)\ \texttt{do} \quad \begin{bmatrix}\mathtt{x}_1\\ \mathtt{x}_2\end{bmatrix} \leftarrow \begin{bmatrix}2\cdot\mathtt{x}_1\\ 3\cdot\mathtt{x}_2\end{bmatrix} \quad \texttt{end}$$

- Yes!
- Value of $\mathtt{x}_1^2$ eventually *outgrows* value of $\mathtt{x}_2$
- At some point we always have

$$4^i \cdot e_1^2 \geq 3^i \cdot e_2.$$

- Reduce Termination to an existential formula over $\mathbb{Z}$ [SAS '20]
  - linear arithmetic: co-NP-complete
  - non-linear arithmetic: non-termination is semi-decidable

Overview



Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Runtime Complexity of TWN-Loops

Goal: Infer (upper) runtime bounds for "real-world" programs

$$\texttt{while}\ (\mathtt{x}_1^2 < \mathtt{x}_2 \land \mathtt{x}_1 > 0)\ \texttt{do}$$
$$\quad \begin{bmatrix} \mathtt{x}_1 \\ \mathtt{x}_2 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot \mathtt{x}_1 \\ 3 \cdot \mathtt{x}_2 \end{bmatrix}$$
$$\texttt{end}$$

Closed forms w.r.t. initial values e_1 and e_2:

- Value of $\mathtt{x}_1^2$: $(2^i \cdot e_1)^2 = 4^i \cdot e_1^2$
- Value of $\mathtt{x}_2$: $3^i \cdot e_2$

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Runtime Complexity of TWN-Loops

Goal: Infer (upper) runtime bounds for “real-world“ programs

$$\texttt{while}\ (\mathtt{x}_1^2 < \mathtt{x}_2 \land \mathtt{x}_1 > 0)\ \texttt{do}\quad \begin{bmatrix}\mathtt{x}_1\\ \mathtt{x}_2\end{bmatrix} \leftarrow \begin{bmatrix}2\cdot\mathtt{x}_1\\ 3\cdot\mathtt{x}_2\end{bmatrix}\quad \texttt{end}$$

Closed forms w.r.t. initial values e_1 and e_2:

- Value of $\mathtt{x}_1^2$: $(2^i \cdot e_1)^2 = 4^i \cdot e_1^2$
- Value of $\mathtt{x}_2$: $3^i \cdot e_2$

- Insert closed forms into guard $x_1^2 < x_2$:

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Runtime Complexity of TWN-Loops

Goal: Infer (upper) runtime bounds for “real-world“ programs

$$\texttt{while}\ (\mathtt{x}_1^2 < \mathtt{x}_2 \wedge \mathtt{x}_1 > 0)\ \texttt{do}$$
$$\quad \begin{bmatrix} \mathtt{x}_1 \\ \mathtt{x}_2 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot \mathtt{x}_1 \\ 3 \cdot \mathtt{x}_2 \end{bmatrix}$$
$$\texttt{end}$$

Closed forms w.r.t. initial values e_1 and e_2:

- Value of $\mathtt{x}_1^2$: $(2^i \cdot e_1)^2 = 4^i \cdot e_1^2$
- Value of $\mathtt{x}_2$: $3^i \cdot e_2$

- Insert closed forms into guard $x_1^2 < x_2$:

$$4^i \cdot e_1^2 < 3^i \cdot e_2$$

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Runtime Complexity of TWN-Loops

Goal: Infer (upper) runtime bounds for “real-world“ programs

$$\texttt{while}\ (\mathtt{x}_1^2 < \mathtt{x}_2 \wedge \mathtt{x}_1 > 0)\ \texttt{do}$$
$$\begin{bmatrix}\mathtt{x}_1\\ \mathtt{x}_2\end{bmatrix} \leftarrow \begin{bmatrix}2 \cdot \mathtt{x}_1\\ 3 \cdot \mathtt{x}_2\end{bmatrix}$$
$$\texttt{end}$$

Closed forms w.r.t. initial values e_1 and e_2:

- Value of $\mathtt{x}_1^2$: $(2^i \cdot e_1)^2 = 4^i \cdot e_1^2$
- Value of $\mathtt{x}_2$: $3^i \cdot e_2$

- Insert closed forms into guard $x_1^2 < x_2$:

$$4^i \cdot e_1^2 - 3^i \cdot e_2 < 0$$

IJCAR 2022
Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Runtime Complexity of TWN-Loops

Goal: Infer (upper) runtime bounds for “real-world“ programs

$$\texttt{while}\ (\mathtt{x}_1^2 < \mathtt{x}_2 \land \mathtt{x}_1 > 0)\ \texttt{do}$$
$$\begin{bmatrix}\mathtt{x}_1 \\ \mathtt{x}_2\end{bmatrix} \leftarrow \begin{bmatrix}2 \cdot \mathtt{x}_1 \\ 3 \cdot \mathtt{x}_2\end{bmatrix}$$
$$\texttt{end}$$

Closed forms w.r.t. initial values e_1 and e_2:

- Value of $\mathtt{x}_1^2$: $(2^i \cdot e_1)^2 = 4^i \cdot e_1^2$
- Value of $\mathtt{x}_2$: $3^i \cdot e_2$

- Insert closed forms into guard $x_1^2 < x_2$:

$$4^i \cdot e_1^2 - 3^i \cdot e_2 < 0$$

- When does the sign of $4^i \cdot e_1^2 - 3^i \cdot e_2$ only depend on e_1^2?

IJCAR 2022
Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Runtime Complexity of TWN-Loops

Goal: Infer (upper) runtime bounds for "real-world" programs

$$\texttt{while } (\mathtt{x}_1^2 < \mathtt{x}_2 \land \mathtt{x}_1 > 0) \texttt{ do} \quad \begin{bmatrix}\mathtt{x}_1 \\ \mathtt{x}_2\end{bmatrix} \leftarrow \begin{bmatrix}2 \cdot \mathtt{x}_1 \\ 3 \cdot \mathtt{x}_2\end{bmatrix} \quad \texttt{end}$$

Closed forms w.r.t. initial values e_1 and e_2:

- Value of $\mathtt{x}_1^2$: $(2^i \cdot e_1)^2 = 4^i \cdot e_1^2$
- Value of $\mathtt{x}_2$: $3^i \cdot e_2$

- Insert closed forms into guard $x_1^2 < x_2$:

$$4^i \cdot e_1^2 - 3^i \cdot e_2 < 0$$

- When does the sign of $4^i \cdot e_1^2 - 3^i \cdot e_2$ only depend on e_1^2?
- When do we have $4^i > -3^i \cdot e_2$?

IJCAR 2022
Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Runtime Complexity of TWN-Loops

Goal: Infer (upper) runtime bounds for “real-world“ programs

```
while (x₁² < x₂ ∧ x₁ > 0) do
    [x₁]    [2 · x₁]
    [x₂] ← [3 · x₂]
end
```

Closed forms w.r.t. initial values e_1 and e_2:

- Value of $\mathtt{x}_1^2$: $(2^i \cdot e_1)^2 = 4^i \cdot e_1^2$
- Value of $\mathtt{x}_2$: $3^i \cdot e_2$

- Insert closed forms into guard $x_1^2 < x_2$:

$$4^i \cdot e_1^2 - 3^i \cdot e_2 < 0$$

- When does the sign of $4^i \cdot e_1^2 - 3^i \cdot e_2$ only depend on e_1^2?
- When do we have $4^i > -3^i \cdot e_2$?
- At this point, the loop terminates or never will.

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Runtime Complexity of TWN-Loops

Goal: Infer (upper) runtime bounds for “real-world“ programs

$$\texttt{while}\ (\texttt{x}_1^2 < \texttt{x}_2 \land \texttt{x}_1 > 0)\ \texttt{do}$$
$$\quad \begin{bmatrix}\texttt{x}_1\\ \texttt{x}_2\end{bmatrix} \leftarrow \begin{bmatrix}2\cdot\texttt{x}_1\\ 3\cdot\texttt{x}_2\end{bmatrix}$$
$$\texttt{end}$$

Closed forms w.r.t. initial values e_1 and e_2:

- Value of $\texttt{x}_1^2$: $(2^i \cdot e_1)^2 = 4^i \cdot e_1^2$
- Value of $\texttt{x}_2$: $3^i \cdot e_2$

- Insert closed forms into guard $x_1^2 < x_2$:

$$4^i \cdot e_1^2 - 3^i \cdot e_2 < 0$$

- When does the sign of $4^i \cdot e_1^2 - 3^i \cdot e_2$ only depend on e_1^2?
- When do we have $4^i > -3^i \cdot e_2$?
- At this point, the loop terminates or never will.
- Bound on stabilization threshold can be computed *automatically*

Runtime Complexity of TWN-Loops

Goal: Infer (upper) runtime bounds for “real-world“ programs

```
while (x_1^2 < x_2 ∧ x_1 > 0) do
    [x_1]    [2 · x_1]
    [x_2] ← [3 · x_2]
end
```

$$\texttt{while}\ (\mathtt{x}_1^2 < \mathtt{x}_2 \land \mathtt{x}_1 > 0)\ \texttt{do}\quad \begin{bmatrix}\mathtt{x}_1\\ \mathtt{x}_2\end{bmatrix} \leftarrow \begin{bmatrix}2\cdot \mathtt{x}_1\\ 3\cdot \mathtt{x}_2\end{bmatrix}\quad \texttt{end}$$

Closed forms w.r.t. initial values e_1 and e_2:

- Value of $\mathtt{x}_1^2$: $(2^i \cdot e_1)^2 = 4^i \cdot e_1^2$
- Value of $\mathtt{x}_2$: $3^i \cdot e_2$

- Insert closed forms into guard $x_1^2 < x_2$:

$$4^i \cdot e_1^2 - 3^i \cdot e_2 < 0$$

- When does the sign of $4^i \cdot e_1^2 - 3^i \cdot e_2$ only depend on e_1^2?
- When do we have $4^i > -3^i \cdot e_2$?
- At this point, the loop terminates or never will.
- Bound on stabilization threshold can be computed *automatically*
- Improve [LPAR ’20] by considering variables individually

Runtime Complexity of TWN-Loops

Goal: Infer (upper) runtime bounds for “real-world“ programs

$$\texttt{while } (\mathtt{x}_1^2 < \mathtt{x}_2 \land \mathtt{x}_1 > 0) \texttt{ do}$$
$$\quad \begin{bmatrix} \mathtt{x}_1 \\ \mathtt{x}_2 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot \mathtt{x}_1 \\ 3 \cdot \mathtt{x}_2 \end{bmatrix}$$
$$\texttt{end}$$

- Bound the point where the truth value of the guard stabilizes.

- When do we have $4^i > -3^i \cdot e_2$?

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Runtime Complexity of TWN-Loops

Goal: Infer (upper) runtime bounds for “real-world“ programs

$$\texttt{while } (\mathtt{x}_1^2 < \mathtt{x}_2 \land \mathtt{x}_1 > 0) \texttt{ do}$$
$$\begin{bmatrix} \mathtt{x}_1 \\ \mathtt{x}_2 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot \mathtt{x}_1 \\ 3 \cdot \mathtt{x}_2 \end{bmatrix}$$
$$\texttt{end}$$

- Bound the point where the truth value of the guard stabilizes.

- When do we have $4^i > -3^i \cdot e_2$?
- Prove: $i > |e_2|$ implies $4^i > -3^i \cdot e_2$

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Runtime Complexity of TWN-Loops

Goal: Infer (upper) runtime bounds for “real-world“ programs

$$\texttt{while}\ (\mathtt{x}_1^2 < \mathtt{x}_2 \land \mathtt{x}_1 > 0)\ \texttt{do}$$
$$\begin{bmatrix} \mathtt{x}_1 \\ \mathtt{x}_2 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot \mathtt{x}_1 \\ 3 \cdot \mathtt{x}_2 \end{bmatrix}$$
$$\texttt{end}$$

- Bound the point where the truth value of the guard stabilizes.

$$4^i > -3^i \cdot e_2$$

- When do we have $4^i > -3^i \cdot e_2$?
- Prove: $i > |e_2|$ implies $4^i > -3^i \cdot e_2$

Runtime Complexity of TWN-Loops

Goal: Infer (upper) runtime bounds for “real-world“ programs

$$
\begin{array}{l}
\texttt{while}\ (\mathtt{x}_1^2 < \mathtt{x}_2 \land \mathtt{x}_1 > 0)\ \texttt{do} \\
\quad \begin{bmatrix} \mathtt{x}_1 \\ \mathtt{x}_2 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot \mathtt{x}_1 \\ 3 \cdot \mathtt{x}_2 \end{bmatrix} \\
\texttt{end}
\end{array}
$$

- Bound the point where the truth value of the guard stabilizes.

$$
\begin{array}{c}
4^i > -3^i \cdot e_2 \\
\Uparrow \\
(4/3)^i > -e_2
\end{array}
$$

- When do we have $4^i > -3^i \cdot e_2$?
- Prove: $i > |e_2|$ implies $4^i > -3^i \cdot e_2$

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Runtime Complexity of TWN-Loops

Goal: Infer (upper) runtime bounds for “real-world“ programs

$$\texttt{while } (x_1^2 < x_2 \wedge x_1 > 0) \texttt{ do} \quad \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot x_1 \\ 3 \cdot x_2 \end{bmatrix} \quad \texttt{end}$$

- Bound the point where the truth value of the guard stabilizes.

$$4^i > -3^i \cdot e_2$$
$$\Uparrow$$
$$(4/3)^i > -e_2$$
$$\Uparrow$$
$$(4/3)^i > |e_2|$$

- When do we have $4^i > -3^i \cdot e_2$?
- Prove: $i > |e_2|$ implies $4^i > -3^i \cdot e_2$

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Runtime Complexity of TWN-Loops

Goal: Infer (upper) runtime bounds for “real-world“ programs

$$\texttt{while}\ (\mathtt{x}_1^2 < \mathtt{x}_2 \land \mathtt{x}_1 > 0)\ \texttt{do}$$
$$\begin{bmatrix}\mathtt{x}_1\\ \mathtt{x}_2\end{bmatrix} \leftarrow \begin{bmatrix}2\cdot\mathtt{x}_1\\ 3\cdot\mathtt{x}_2\end{bmatrix}$$
$$\texttt{end}$$

- Bound the point where the truth value of the guard stabilizes.

- When do we have $4^i > -3^i \cdot e_2$?
- Prove: $i > |e_2|$ implies $4^i > -3^i \cdot e_2$

$$4^i > -3^i \cdot e_2$$
$$\Uparrow$$
$$(4/3)^i > -e_2$$
$$\Uparrow$$
$$(4/3)^i > |e_2|$$
$$\Uparrow$$
$$i > \log(|e_2|)$$

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Runtime Complexity of TWN-Loops

Goal: Infer (upper) runtime bounds for “real-world“ programs

```
while (x_1^2 < x_2 ∧ x_1 > 0) do
    [x_1, x_2] ← [2 · x_1, 3 · x_2]
end
```

$$\texttt{while}\ (\texttt{x}_1^2 < \texttt{x}_2 \land \texttt{x}_1 > 0)\ \texttt{do}\quad \begin{bmatrix}\texttt{x}_1\\ \texttt{x}_2\end{bmatrix} \leftarrow \begin{bmatrix}2\cdot\texttt{x}_1\\ 3\cdot\texttt{x}_2\end{bmatrix}\quad \texttt{end}$$

- Bound the point where the truth value of the guard stabilizes.

$$4^i > -3^i \cdot e_2$$
$$\Uparrow$$
$$(4/3)^i > -e_2$$
$$\Uparrow$$
$$(4/3)^i > |e_2|$$
$$\Uparrow$$
$$i > \log(|e_2|)$$

- When do we have $4^i > -3^i \cdot e_2$?
- Prove: $i > |e_2|$ implies $4^i > -3^i \cdot e_2$
- By Termination: $|e_2| + 1$ is runtime bound

IJCAR 2022
Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Runtime Complexity of TWN-Loops

Goal: Infer (upper) runtime bounds for “real-world“ programs

$$\texttt{while } (\texttt{x}_1^2 < \texttt{x}_2 \land \texttt{x}_1 > 0) \texttt{ do} \quad \begin{bmatrix} \texttt{x}_1 \\ \texttt{x}_2 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot \texttt{x}_1 \\ 3 \cdot \texttt{x}_2 \end{bmatrix} \quad \texttt{end}$$

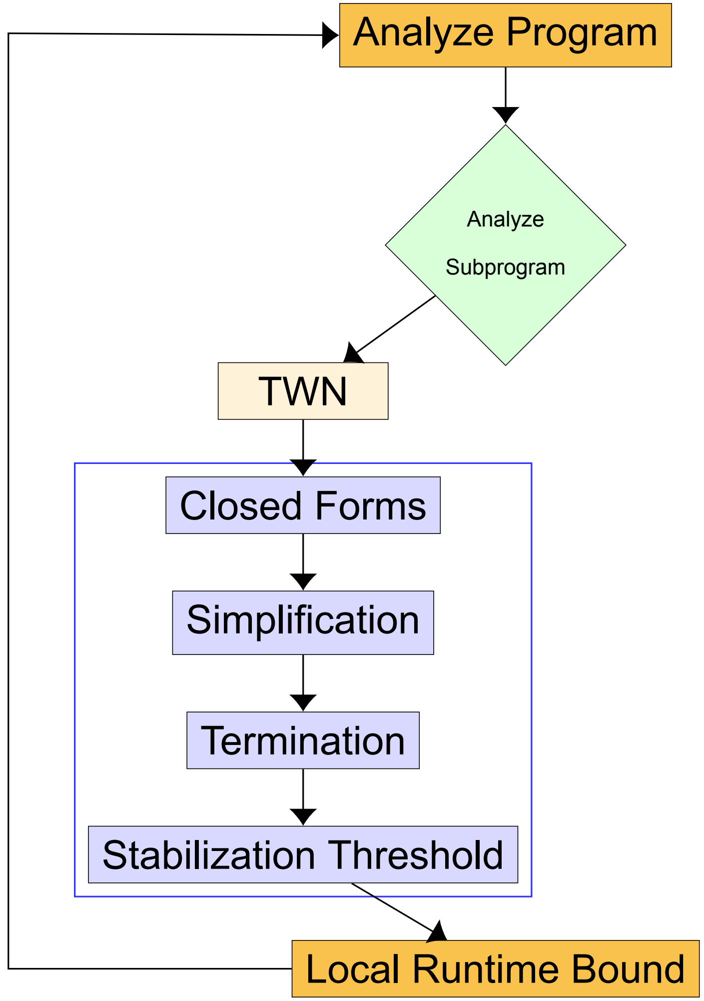
- Bound the point where the truth value of the guard stabilizes.

$$4^i > -3^i \cdot e_2$$
$$\Uparrow$$
$$(4/3)^i > -e_2$$
$$\Uparrow$$
$$(4/3)^i > |e_2|$$
$$\Uparrow$$
$$i > \log(|e_2|)$$

- When do we have $4^i > -3^i \cdot e_2$?
- Prove: $i > |e_2|$ implies $4^i > -3^i \cdot e_2$
- By Termination: $|e_2| + 1$ is runtime bound
- Procedure is complete and implemented in *KoAT*

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Overview



Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Runtime Complexity of Integer Programs

Goal: Infer (upper) runtime bounds for “real-world“ programs

$$\texttt{while}\ (\mathtt{x}_1^2 < \mathtt{x}_2)\ \texttt{do}$$
$$\begin{bmatrix}\mathtt{x}_1\\ \mathtt{x}_2\end{bmatrix} \leftarrow \begin{bmatrix}2\cdot\mathtt{x}_1\\ 3\cdot\mathtt{x}_2\end{bmatrix}$$
$$\texttt{end}$$

Runtime Complexity of Integer Programs

Goal: Infer (upper) runtime bounds for “real-world“ programs

```
while (x_3 > 0) do

  while (x_1^2 < x_2) do
    [x_1; x_2] ← [2 · x_1; 3 · x_2]
  end

end
```

$$\texttt{while } (x_3 > 0) \texttt{ do}$$

$$\quad \texttt{while } (x_1^2 < x_2) \texttt{ do}$$

$$\quad\quad \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot x_1 \\ 3 \cdot x_2 \end{bmatrix}$$

$$\quad \texttt{end}$$

$$\texttt{end}$$

Runtime Complexity of Integer Programs

Goal: Infer (upper) runtime bounds for “real-world“ programs

$$
\begin{array}{l}
\texttt{while } (x_3 > 0) \texttt{ do} \\
\quad \texttt{while } (x_1^2 < x_2) \texttt{ do} \\
\qquad \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot x_1 \\ 3 \cdot x_2 \end{bmatrix} \\
\quad \texttt{end} \\
\quad \begin{bmatrix} x_3 \end{bmatrix} \leftarrow \begin{bmatrix} x_3 - 1 \end{bmatrix} \\
\texttt{end}
\end{array}
$$

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Runtime Complexity of Integer Programs

Goal: Infer (upper) runtime bounds for "real-world" programs

$$
\begin{array}{l}
\texttt{while } (x_3 > 0) \texttt{ do} \\
\quad \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leftarrow \begin{bmatrix} x_4 \\ x_5^2 \end{bmatrix} \\
\quad \texttt{while } (x_1^2 < x_2) \texttt{ do} \\
\qquad \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot x_1 \\ 3 \cdot x_2 \end{bmatrix} \\
\quad \texttt{end} \\
\quad \begin{bmatrix} x_3 \end{bmatrix} \leftarrow \begin{bmatrix} x_3 - 1 \end{bmatrix} \\
\texttt{end}
\end{array}
$$

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Runtime Complexity of Integer Programs

Goal: Infer (upper) runtime bounds for “real-world“ programs

$$
\begin{array}{l}
\texttt{while}\ (x_3 > 0)\ \texttt{do} \\
\quad \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leftarrow \begin{bmatrix} x_4 \\ x_5^2 \end{bmatrix} \\
\quad \texttt{while}\ (x_1^2 < x_2)\ \texttt{do} \\
\quad\quad \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot x_1 \\ 3 \cdot x_2 \end{bmatrix} \\
\quad \texttt{end} \\
\quad \begin{bmatrix} x_3 \end{bmatrix} \leftarrow \begin{bmatrix} x_3 - 1 \end{bmatrix} \\
\texttt{end}
\end{array}
$$

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Runtime Complexity of Integer Programs

Goal: Infer (upper) runtime bounds for “real-world“ programs

$$
\begin{array}{l}
\texttt{while}\ (x_3 > 0)\ \texttt{do} \\
\quad \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leftarrow \begin{bmatrix} x_4 \\ x_5^2 \end{bmatrix} \\
\quad \texttt{while}\ (x_1^2 < x_2)\ \texttt{do} \\
\quad\quad \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot x_1 \\ 3 \cdot x_2 \end{bmatrix} \\
\quad \texttt{end} \\
\quad \begin{bmatrix} x_3 \end{bmatrix} \leftarrow \begin{bmatrix} x_3 - 1 \end{bmatrix} \\
\texttt{end}
\end{array}
$$

- How often do we execute the inner loop?

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Runtime Complexity of Integer Programs

Goal: Infer (upper) runtime bounds for "real-world" programs

$$
\begin{array}{l}
\texttt{while } (x_3 > 0) \texttt{ do} \\
\quad \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leftarrow \begin{bmatrix} x_4 \\ x_5^2 \end{bmatrix} \\
\quad \texttt{while } (x_1^2 < x_2) \texttt{ do} \\
\quad\quad \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leftarrow \begin{bmatrix} 2 \cdot x_1 \\ 3 \cdot x_2 \end{bmatrix} \\
\quad \texttt{end} \\
\quad \begin{bmatrix} x_3 \end{bmatrix} \leftarrow \begin{bmatrix} x_3 - 1 \end{bmatrix} \\
\texttt{end}
\end{array}
$$

- How often do we execute the inner loop?
- Idea: Analyze different subprograms and combine results

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Runtime Complexity of Integer Programs

Goal: Infer (upper) runtime bounds for “real-world“ programs

```
while (x_3 > 0) do
```

$$\begin{bmatrix} \mathtt{x}_1 \\ \mathtt{x}_2 \end{bmatrix} \leftarrow \begin{bmatrix} \mathtt{x}_4 \\ \mathtt{x}_5^2 \end{bmatrix}$$

costs: $|e_2| + 1$

$$\begin{bmatrix} \mathtt{x}_3 \end{bmatrix} \leftarrow \begin{bmatrix} \mathtt{x}_3 - 1 \end{bmatrix}$$

```
end
```

- How often do we execute the inner loop?
- Idea: Analyze different subprograms and combine results

Inner loop executions: $(|e_2| + 1)$

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Runtime Complexity of Integer Programs

Goal: Infer (upper) runtime bounds for "real-world" programs

$$
\begin{array}{l}
\texttt{while}\ (\mathtt{x}_3 > 0)\ \texttt{do} \\
\quad \begin{bmatrix} \mathtt{x}_1 \\ \mathtt{x}_2 \end{bmatrix} \leftarrow \begin{bmatrix} \mathtt{x}_4 \\ \mathtt{x}_5^2 \end{bmatrix} \\
\quad \text{costs: } |e_2| + 1 \\
\quad \begin{bmatrix} \mathtt{x}_3 \end{bmatrix} \leftarrow \begin{bmatrix} \mathtt{x}_3 - 1 \end{bmatrix} \\
\texttt{end}
\end{array}
$$

- How often do we execute the inner loop?
- Idea: Analyze different subprograms and combine results
- Respect size of variables:

Inner loop executions: $(|e_2| + 1)$

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Runtime Complexity of Integer Programs

Goal: Infer (upper) runtime bounds for “real-world“ programs

```
while (x3 > 0) do
    costs: |e5^2| + 1
    [x3] ← [x3 − 1]
end
```

- How often do we execute the inner loop?
- Idea: Analyze different subprograms and combine results
- Respect size of variables:
  - Size of x_2 is bounded by e_5^2 before inner loop

Inner loop executions: $(|e_5|^2 + 1)$

Runtime Complexity of Integer Programs

Goal: Infer (upper) runtime bounds for “real-world“ programs

```
while (x₃ > 0) do
   costs: |e₅²| + 1
   [x₃] ← [x₃ − 1]
end
```

- How often do we execute the inner loop?
- Idea: Analyze different subprograms and combine results
- Respect size of variables:
  - Size of x_2 is bounded by e_5^2 before inner loop
- Use ranking functions (MΦRFs) to analyze outer loop

Inner loop executions: $|e_3| \cdot (|e_5|^2 + 1)$

IJCAR 2022
Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Runtime Complexity of Integer Programs

Goal: Infer (upper) runtime bounds for “real-world“ programs

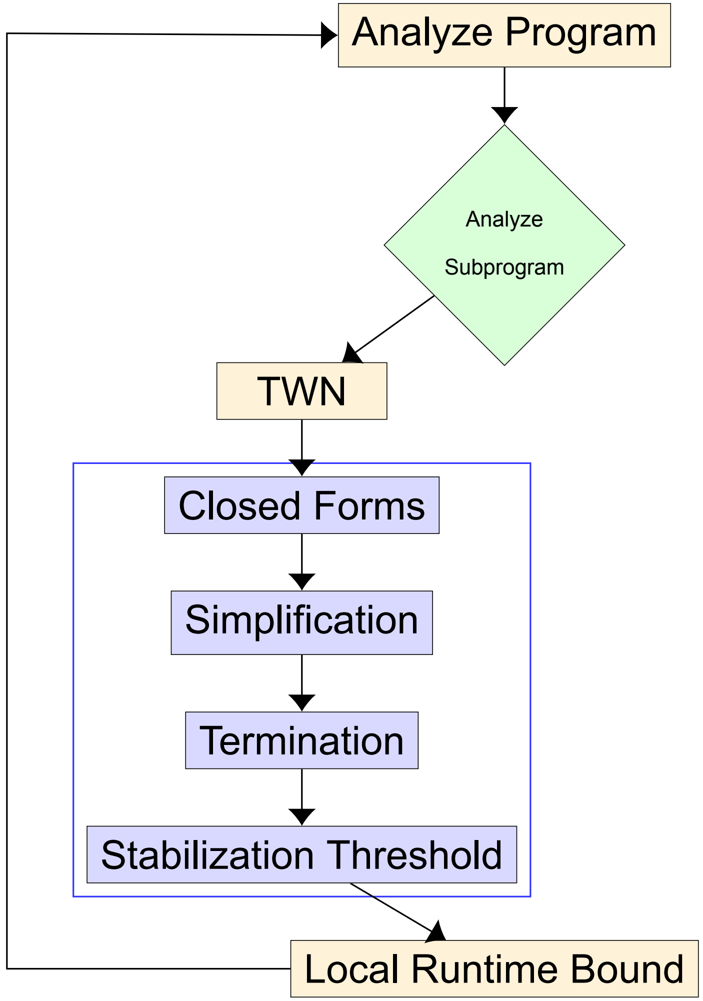
```
while (x_3 > 0) do
    costs: |e_5^2| + 1
    [x_3] ← [x_3 − 1]
end
```

- How often do we execute the inner loop?
- Idea: Analyze different subprograms and combine results
- Respect size of variables:
  - Size of x_2 is bounded by e_5^2 before inner loop
- Use ranking functions (MΦRFs) to analyze outer loop

Inner loop executions: $|e_3| \cdot (|e_5|^2 + 1) \in \mathcal{O}(n^3)$

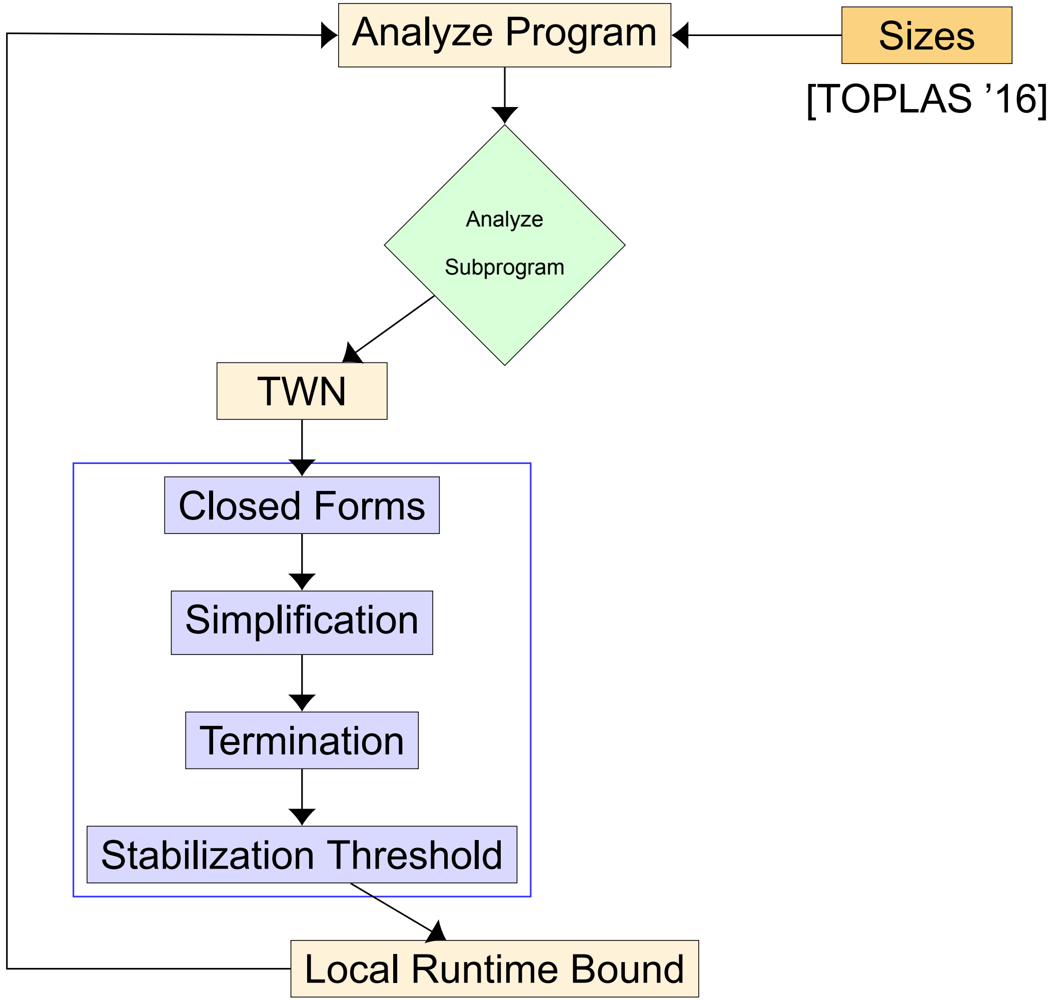
Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Overview



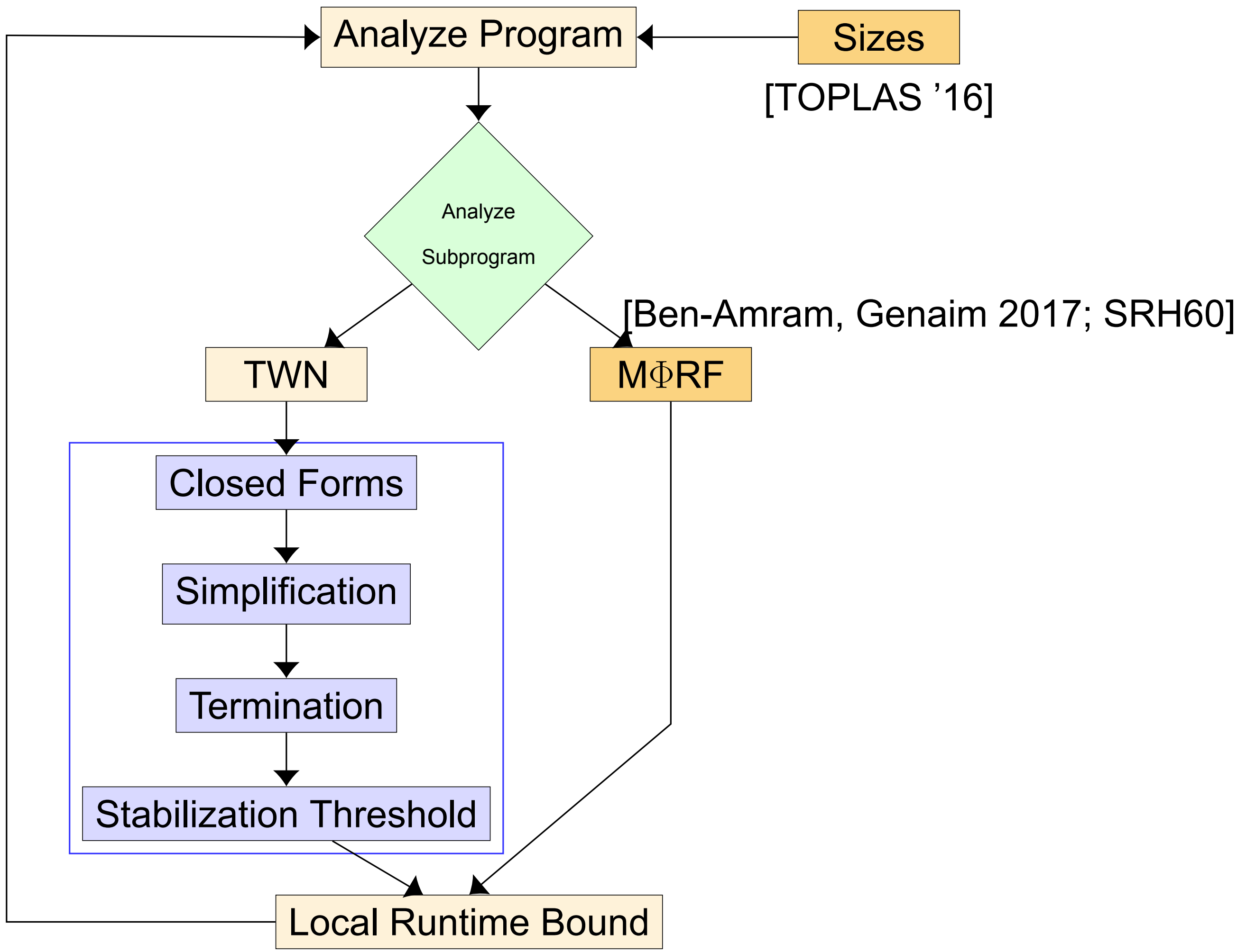
Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Overview



Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Overview



Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Evaluation of our Implementation in KoAT2

- C_Complexity consisting of 504 (mainly linear) benchmarks from TPDB

	$\mathcal{O}(1)$	$\mathcal{O}(n)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n^{>2})$	$< \infty$	AVG(s)
KoAT2 + TWN	20	111	3	2	136	2.54

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Evaluation of our Implementation in KoAT2

- C_Complexity consisting of 504 (mainly linear) benchmarks from TPDB

	$\mathcal{O}(1)$	$\mathcal{O}(n)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n^{>2})$	$< \infty$	AVG(s)
KoAT2 + TWN	20	111	3	2	136	2.54
Loopus	17	170	49	5	241	0.42
KoAT1	25	169	74	12	286	1.77
CoFloCo	22	196	66	5	289	0.62
MaxCore	23	216	66	7	312	2.02

IJCAR 2022
Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Evaluation of our Implementation in KoAT2

- C_Complexity consisting of 504 (mainly linear) benchmarks from TPDB

	$\mathcal{O}(1)$	$\mathcal{O}(n)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n^{>2})$	$< \infty$	AVG(s)
KoAT2 + TWN	20	111	3	2	136	2.54
Loopus	17	170	49	5	241	0.42
KoAT1	25	169	74	12	286	1.77
CoFloCo	22	196	66	5	289	0.62
MaxCore	23	216	66	7	312	2.02
KoAT2 + MΦRF	24	226	68	10	328	8.23

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Evaluation of our Implementation in KoAT2

- C_Complexity consisting of 504 (mainly linear) benchmarks from TPDB

	$\mathcal{O}(1)$	$\mathcal{O}(n)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n^{>2})$	$< \infty$	AVG(s)
KoAT2 + TWN	20	111	3	2	136	2.54
Loopus	17	170	49	5	241	0.42
KoAT1	25	169	74	12	286	1.77
CoFloCo	22	196	66	5	289	0.62
MaxCore	23	216	66	7	312	2.02
KoAT2 + MΦRF	24	226	68	10	328	8.23
KoAT2 + TWN + MΦRF	26	231	73	13	344	8.72

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Evaluation of our Implementation in KoAT2

- C_Complexity consisting of 504 (mainly linear) benchmarks from TPDB

	$\mathcal{O}(1)$	$\mathcal{O}(n)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n^{>2})$	$< \infty$	AVG(s)
KoAT2 + TWN	20	111	3	2	136	2.54
Loopus	17	170	49	5	241	0.42
KoAT1	25	169	74	12	286	1.77
CoFloCo	22	196	66	5	289	0.62
MaxCore	23	216	66	7	312	2.02
KoAT2 + MΦRF	24	226	68	10	328	8.23
KoAT2 + TWN + MΦRF	26	231	73	13	344	8.72

- At most 386 benchmarks might terminate

IJCAR 2022
Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Evaluation of our Implementation in KoAT2

- C_Complexity consisting of 504 (mainly linear) benchmarks from TPDB

	$\mathcal{O}(1)$	$\mathcal{O}(n)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n^{>2})$	$< \infty$	AVG(s)	succ. rate
KoAT2 + TWN	20	111	3	2	136	2.54	35%
Loopus	17	170	49	5	241	0.42	62%
KoAT1	25	169	74	12	286	1.77	74%
CoFloCo	22	196	66	5	289	0.62	75%
MaxCore	23	216	66	7	312	2.02	80%
KoAT2 + MΦRF	24	226	68	10	328	8.23	85%
KoAT2 + TWN + MΦRF	26	231	73	13	344	8.72	89%

- At most 386 benchmarks might terminate
- KoAT2 + TWN + MΦRF solves 89% of benchmarks which might terminate

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Conclusion & Future Work

- Conclusion

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Conclusion & Future Work

- Conclusion
  - Introduced modular approach for complexity analysis combining

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Conclusion & Future Work

- Conclusion
  - Introduced modular approach for complexity analysis combining
    - Procedure to handle twn-loops

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Conclusion & Future Work

- Conclusion
  - Introduced modular approach for complexity analysis combining
    - Procedure to handle twn-loops
    - MΦRFs

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Conclusion & Future Work

► Conclusion

- Introduced modular approach for complexity analysis combining
  - Procedure to handle twn-loops
  - MΦRFs
- Handle loops with non-linear arithmetic

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Conclusion & Future Work

- Conclusion
  - Introduced modular approach for complexity analysis combining
    - Procedure to handle twn-loops
    - MΦRFs
  - Handle loops with non-linear arithmetic
  - Complete for all twn-loops with linear arithmetic

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Conclusion & Future Work

▶ Conclusion

- Introduced modular approach for complexity analysis combining
  - Procedure to handle twn-loops
  - MΦRFs
- Handle loops with non-linear arithmetic
- Complete for all twn-loops with linear arithmetic
- `KoAT2` outperforms other state-of-the-art tools

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Conclusion & Future Work

- Conclusion
  - Introduced modular approach for complexity analysis combining
    - Procedure to handle twn-loops
    - MΦRFs
  - Handle loops with non-linear arithmetic
  - Complete for all twn-loops with linear arithmetic
  - `KoAT2` outperforms other state-of-the-art tools
- Future work

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Conclusion & Future Work

- Conclusion
  - Introduced modular approach for complexity analysis combining
    - Procedure to handle twn-loops
    - MΦRFs
  - Handle loops with non-linear arithmetic
  - Complete for all twn-loops with linear arithmetic
  - `KoAT2` outperforms other state-of-the-art tools
- Future work
  - Extend class of loops by transformations

Conclusion & Future Work

- Conclusion
  - Introduced modular approach for complexity analysis combining
    - Procedure to handle twn-loops
    - MΦRFs
  - Handle loops with non-linear arithmetic
  - Complete for all twn-loops with linear arithmetic
  - `KoAT2` outperforms other state-of-the-art tools
- Future work
  - Extend class of loops by transformations

`https://aprove-developers.github.io/KoAT_TWN/`

Conclusion & Future Work

- Conclusion
  - Introduced modular approach for complexity analysis combining
    - Procedure to handle twn-loops
    - MΦRFs
  - Handle loops with non-linear arithmetic
  - Complete for all twn-loops with linear arithmetic
  - `KoAT2` outperforms other state-of-the-art tools
- Future work
  - Extend class of loops by transformations

`https://aprove-developers.github.io/KoAT_TWN/`

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2

Conclusion & Future Work

- **Conclusion**
  - Introduced modular approach for complexity analysis combining
    - Procedure to handle twn-loops
    - MΦRFs
  - Handle loops with non-linear arithmetic
  - Complete for all twn-loops with linear arithmetic
  - `KoAT2` outperforms other state-of-the-art tools
- **Future work**
  - Extend class of loops by transformations

`https://aprove-developers.github.io/KoAT_TWN/`

Thank You!

Nils Lommen, Eleanore Meyer, and Jürgen Giesl
RWTH Aachen University – LuFGi2