Improved Automatic Complexity Analysis of Integer Programs

Workshop on Termination 2022

Jürgen Giesl, Nils Lommen, Marcel Hark, and Eleanore Meyer

Motivation

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

$$\texttt{while } (\texttt{y} > 0) \texttt{ do} \quad \begin{bmatrix}\texttt{x}\\ \texttt{y}\end{bmatrix} \leftarrow \begin{bmatrix}\texttt{x}\\ \texttt{y}-1\end{bmatrix} \quad \texttt{end}$$

$$\texttt{while } (\texttt{x} > 0) \texttt{ do} \quad \begin{bmatrix}\texttt{x}\\ \texttt{z}\end{bmatrix} \leftarrow \begin{bmatrix}\texttt{x}-1\\ \texttt{z}-1\end{bmatrix} \quad \texttt{end}$$

- runtime complexity:

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

Motivation

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

$$\texttt{while (y > 0) do} \quad \begin{bmatrix} x \\ y \end{bmatrix} \leftarrow \begin{bmatrix} x \\ y - 1 \end{bmatrix} \quad \texttt{end}$$

$$\texttt{while (x > 0) do} \quad \begin{bmatrix} x \\ z \end{bmatrix} \leftarrow \begin{bmatrix} x - 1 \\ z - 1 \end{bmatrix} \quad \texttt{end}$$

- runtime complexity:
  - linear

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

Motivation

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

$$\texttt{while (y > 0) do} \quad \begin{bmatrix} \texttt{x} \\ \texttt{y} \end{bmatrix} \leftarrow \begin{bmatrix} \texttt{x} \\ \texttt{y} - 1 \end{bmatrix} \quad \texttt{end}$$

$$\texttt{while (x > 0) do} \quad \begin{bmatrix} \texttt{x} \\ \texttt{z} \end{bmatrix} \leftarrow \begin{bmatrix} \texttt{x} - 1 \\ \texttt{z} - 1 \end{bmatrix} \quad \texttt{end}$$

- runtime complexity:
  - linear
  - $y_0 + x_0$

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

Motivation

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

$$\texttt{while (y > 0) do} \quad \begin{bmatrix} \mathtt{x} \\ \mathtt{y} \end{bmatrix} \leftarrow \begin{bmatrix} \mathtt{x} + \mathtt{y} \\ \mathtt{y} - 1 \end{bmatrix} \quad \texttt{end}$$

$$\texttt{while (x > 0) do} \quad \begin{bmatrix} \mathtt{x} \\ \mathtt{z} \end{bmatrix} \leftarrow \begin{bmatrix} \mathtt{x} - 1 \\ \mathtt{z} - 1 \end{bmatrix} \quad \texttt{end}$$

- runtime complexity:
  - linear
  - $y_0 + x_0$

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

Motivation

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

$$\texttt{while } (\mathtt{y} > 0) \texttt{ do} \quad \begin{bmatrix} \mathtt{x} \\ \mathtt{y} \end{bmatrix} \leftarrow \begin{bmatrix} \mathtt{x} + \mathtt{y} \\ \mathtt{y} - 1 \end{bmatrix} \quad \texttt{end}$$

$$\texttt{while } (\mathtt{x} > 0) \texttt{ do} \quad \begin{bmatrix} \mathtt{x} \\ \mathtt{z} \end{bmatrix} \leftarrow \begin{bmatrix} \mathtt{x} - 1 \\ \mathtt{z} - 1 \end{bmatrix} \quad \texttt{end}$$

- runtime complexity:
  - quadratic

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

Motivation

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

$$
\begin{aligned}
&\texttt{while (y > 0) do} \\
&\quad \begin{bmatrix} \texttt{x} \\ \texttt{y} \end{bmatrix} \leftarrow \begin{bmatrix} \texttt{x} + \texttt{y} \\ \texttt{y} - 1 \end{bmatrix} \\
&\texttt{end} \\[1em]
&\texttt{while (x > 0) do} \\
&\quad \begin{bmatrix} \texttt{x} \\ \texttt{z} \end{bmatrix} \leftarrow \begin{bmatrix} \texttt{x} - 1 \\ \texttt{z} - 1 \end{bmatrix} \\
&\texttt{end}
\end{aligned}
$$

- runtime complexity:
  - quadratic
  - $y_0 + size(x)$

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

Motivation

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

$$
\begin{array}{l}
\texttt{while (y > 0) do} \\
\quad \begin{bmatrix} \texttt{x} \\ \texttt{y} \end{bmatrix} \leftarrow \begin{bmatrix} \texttt{x} + \texttt{y} \\ \texttt{y} - 1 \end{bmatrix} \\
\texttt{end} \\
\\
\texttt{while (x > 0) do} \\
\quad \begin{bmatrix} \texttt{x} \\ \texttt{z} \end{bmatrix} \leftarrow \begin{bmatrix} \texttt{x} - 1 \\ \texttt{z} - 1 \end{bmatrix} \\
\texttt{end}
\end{array}
$$

- runtime complexity:
  - quadratic
  - $y_0 + size(x) = y_0 + (x_0 + y_0^2)$

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

Motivation

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

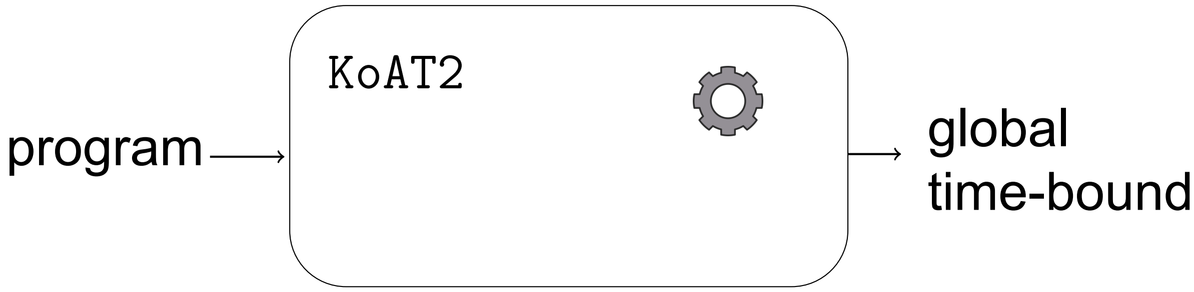
$$
\begin{array}{l}
\texttt{while (y > 0) do} \\
\quad \begin{bmatrix} x \\ y \end{bmatrix} \leftarrow \begin{bmatrix} x + y \\ y - 1 \end{bmatrix} \\
\texttt{end} \\
\\
\texttt{while (x > 0) do} \\
\quad \begin{bmatrix} x \\ z \end{bmatrix} \leftarrow \begin{bmatrix} x + z \\ z - 1 \end{bmatrix} \\
\texttt{end}
\end{array}
$$

- runtime complexity:
  - quadratic?

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

Motivation

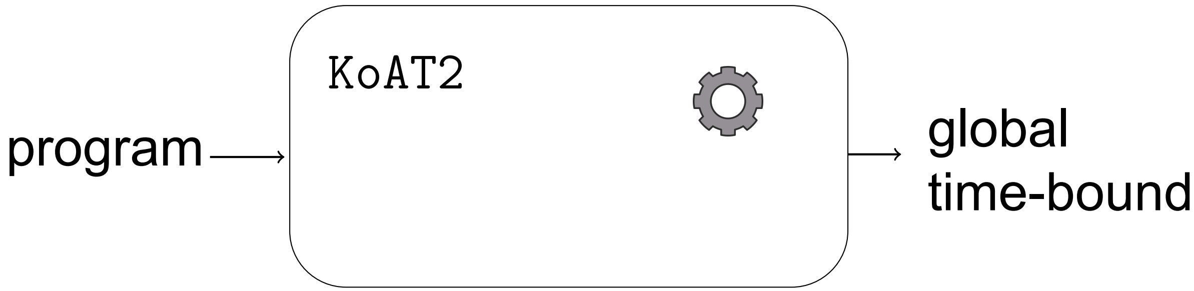
Goal: Infer (upper) runtime bounds for “real-world“ programs



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

Motivation

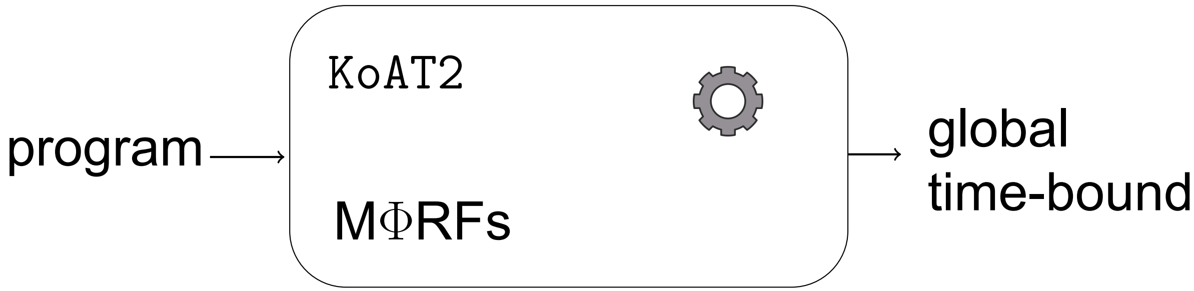
Goal: Infer (upper) runtime bounds for “real-world“ programs



Contributions:

Motivation

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



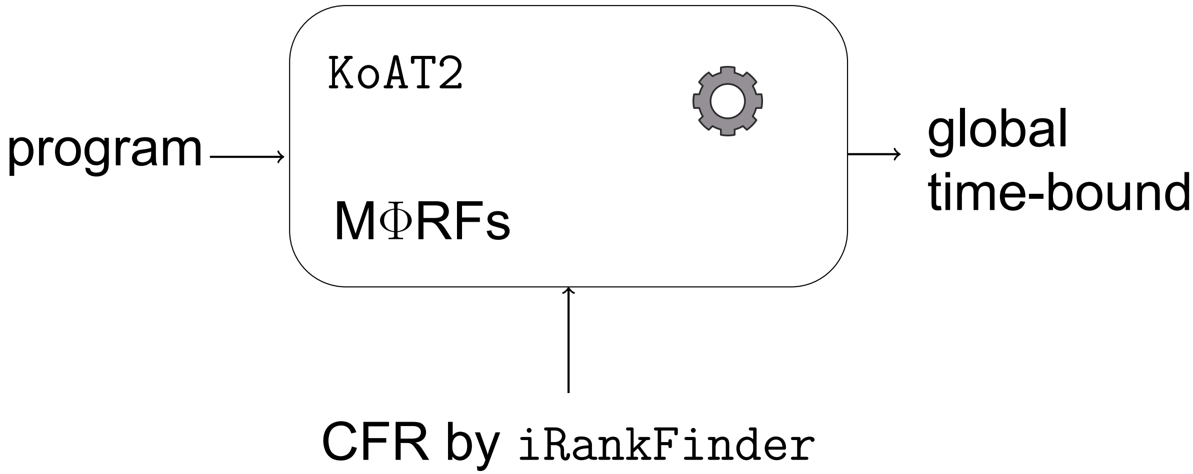
Contributions:

- Integrate MΦRFs in *modular* approach to compute runtime bounds [Ben-Amram, Genaim ’17]

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

Motivation

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



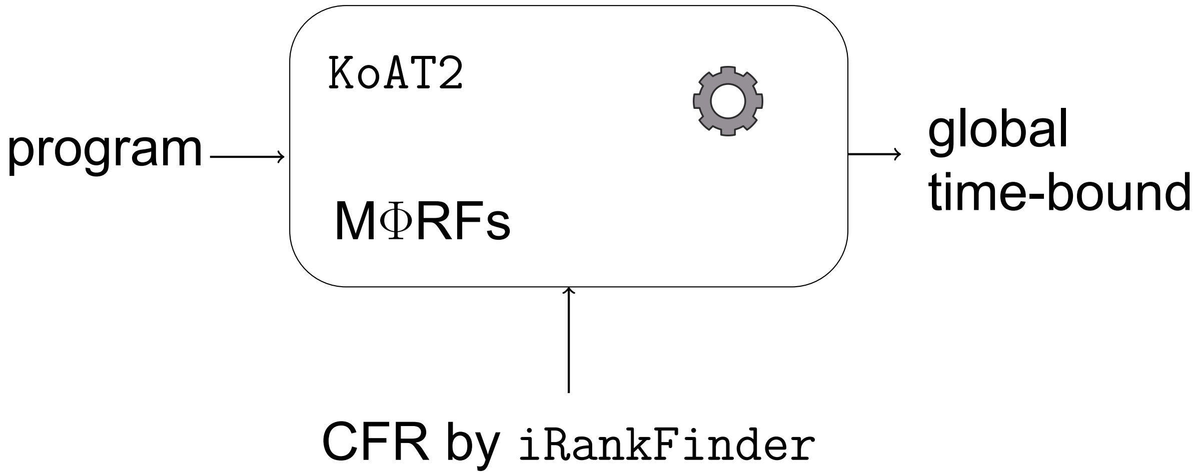
Contributions:

- Integrate MΦRFs in *modular* approach to compute runtime bounds [Ben-Amram, Genaim ’17]
- Incorporate *local* control-flow refinement [Doménech et al. ’19]

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

Motivation

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



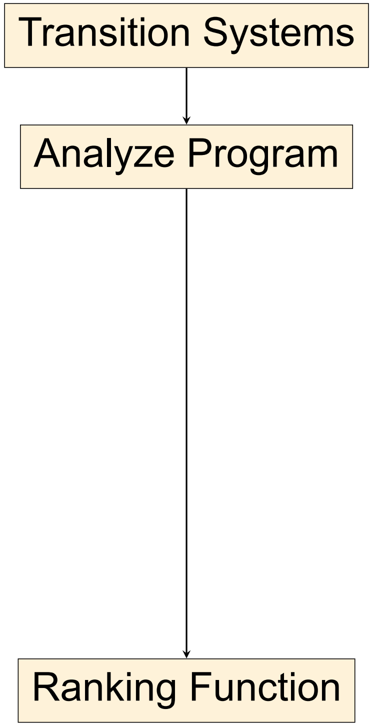
Contributions:

- Integrate MΦRFs in *modular* approach to compute runtime bounds [Ben-Amram, Genaim ’17]
- Incorporate *local* control-flow refinement [Doménech et al. ’19]
- Provide implementation in complexity analysis tool KoAT [TOPLAS ’16]

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

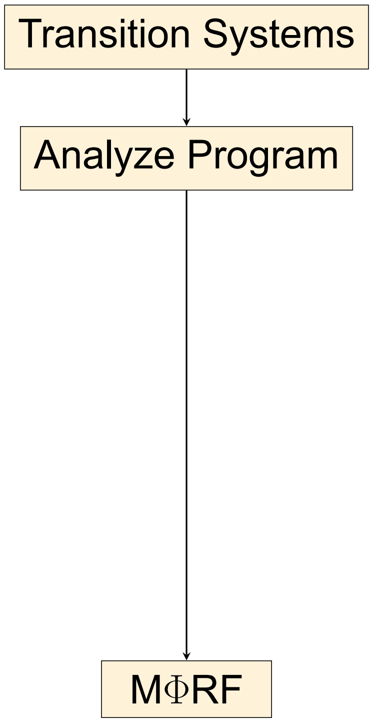
Overview

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



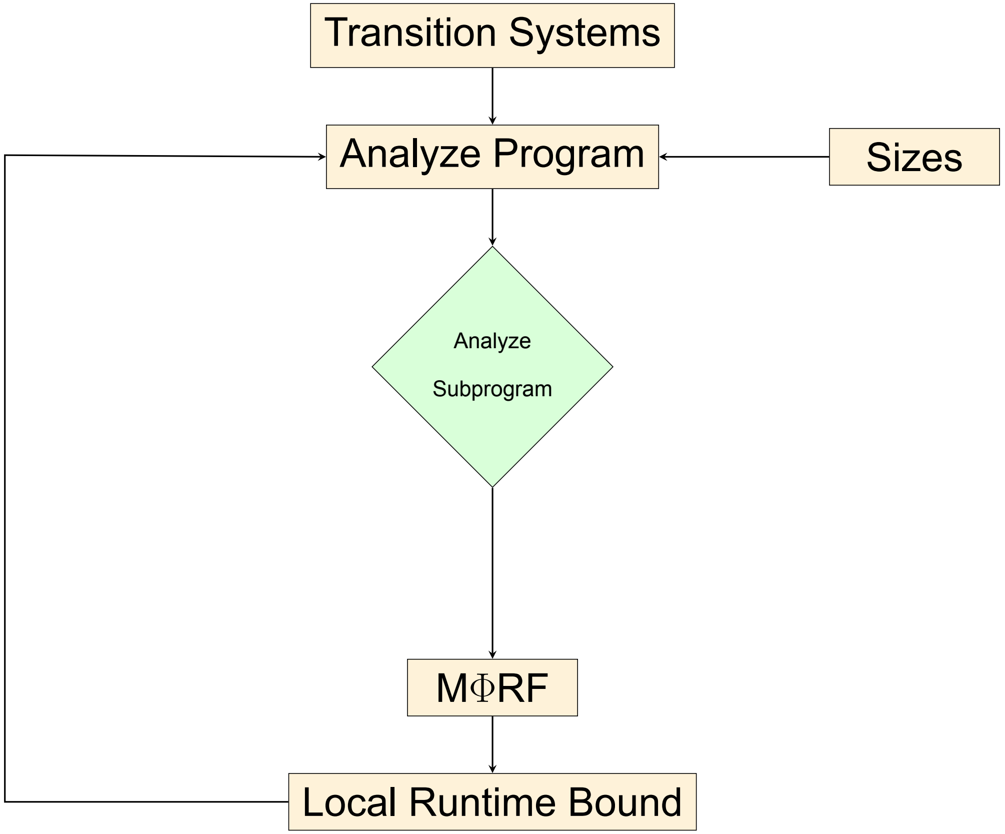
Overview

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



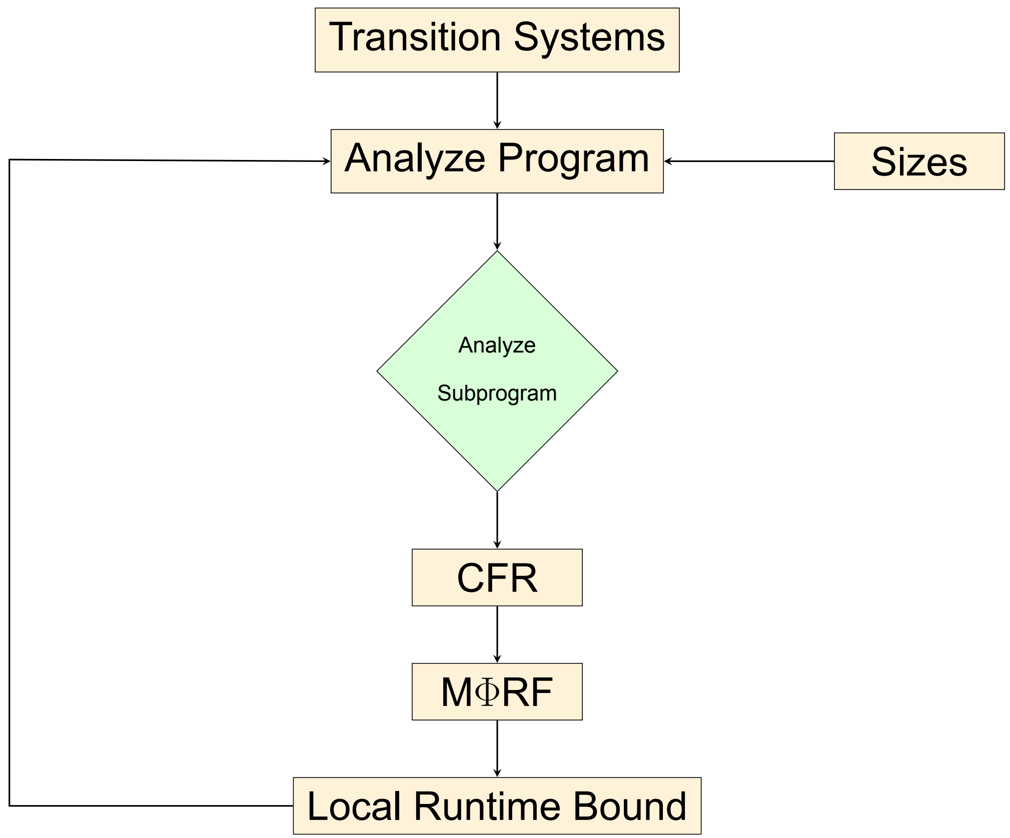
Overview

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



Overview

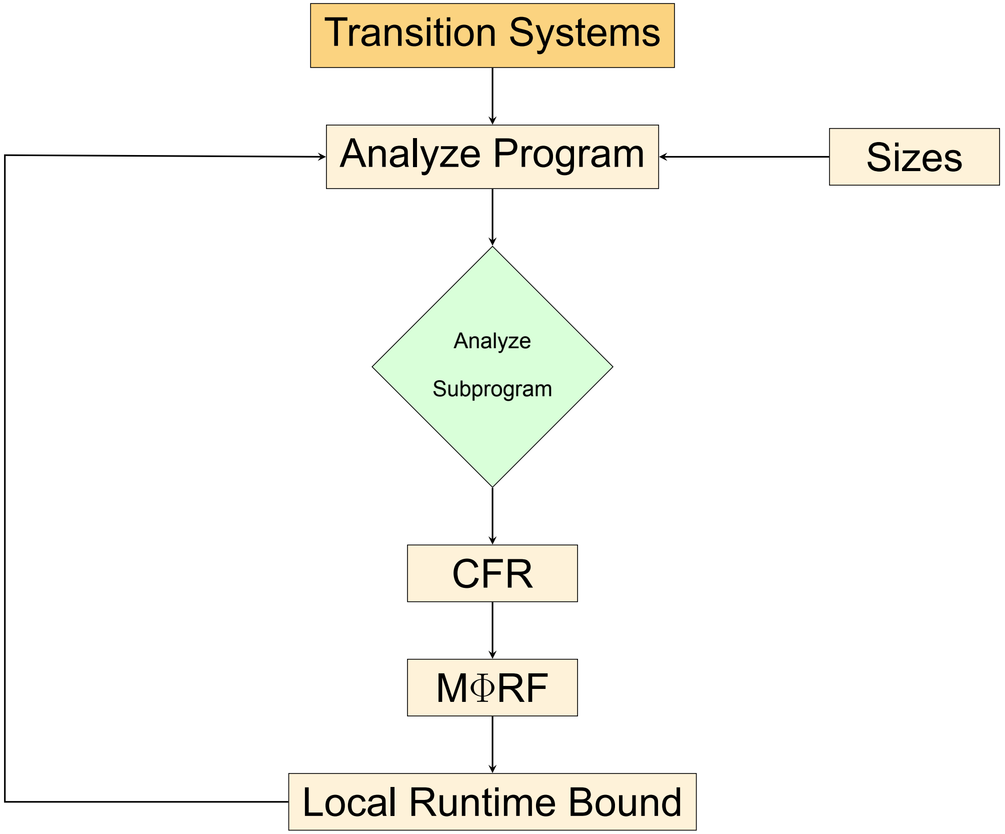
Goal: Infer (upper) runtime bounds for “real-world“ programs



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

Overview

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



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

Complexity Analysis of Integer Programs

Transform “real-world“ programs into *integer program*

$$\texttt{while } (y > 0) \texttt{ do}$$
$$\begin{bmatrix} x \\ y \end{bmatrix} \leftarrow \begin{bmatrix} x + y \\ y - 1 \end{bmatrix}$$
$$\texttt{end}$$

$$\texttt{while } (x > 0) \texttt{ do}$$
$$\begin{bmatrix} x \\ z \end{bmatrix} \leftarrow \begin{bmatrix} x + z \\ z - 1 \end{bmatrix}$$
$$\texttt{end}$$

$\rightsquigarrow$

WST22
Jürgen Giesl, **Nils Lommen**, Marcel Hark, and Eleanore Meyer
RWTH Aachen University – LuFGi2

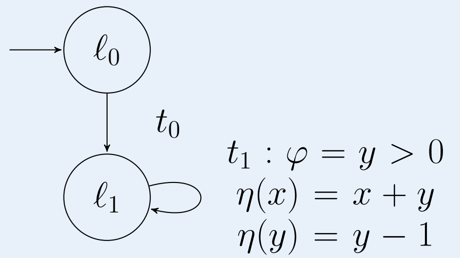
Complexity Analysis of Integer Programs

Transform “real-world“ programs into *integer program*

```
while (y > 0) do
```
$$\begin{bmatrix} x \\ y \end{bmatrix} \leftarrow \begin{bmatrix} x + y \\ y - 1 \end{bmatrix}$$
```
end

while (x > 0) do
```
$$\begin{bmatrix} x \\ z \end{bmatrix} \leftarrow \begin{bmatrix} x + z \\ z - 1 \end{bmatrix}$$
```
end
```

$\rightsquigarrow$



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

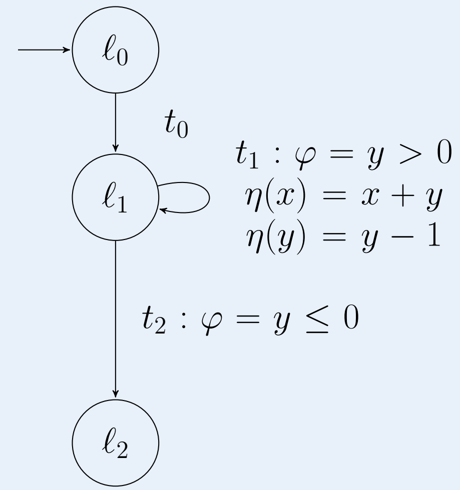
Complexity Analysis of Integer Programs

Transform “real-world“ programs into *integer program*

```
while (y > 0) do
  [x]    [x + y]
  [y] ←  [y − 1]
end

while (x > 0) do
  [x]    [x + z]
  [z] ←  [z − 1]
end
```

$\rightsquigarrow$



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

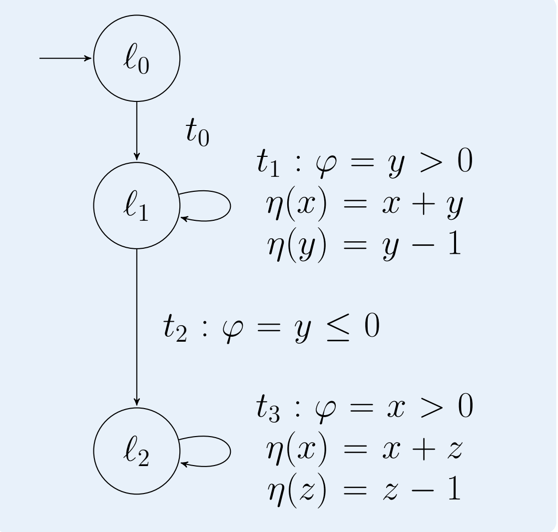
Complexity Analysis of Integer Programs

Transform “real-world“ programs into *integer program*

```
while (y > 0) do
    [x]    [x + y]
    [y] ← [y − 1]
end

while (x > 0) do
    [x]    [x + z]
    [z] ← [z − 1]
end
```

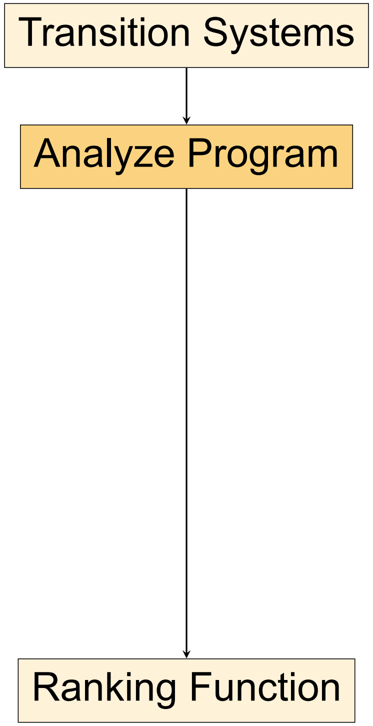
$\leadsto$



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

Overview

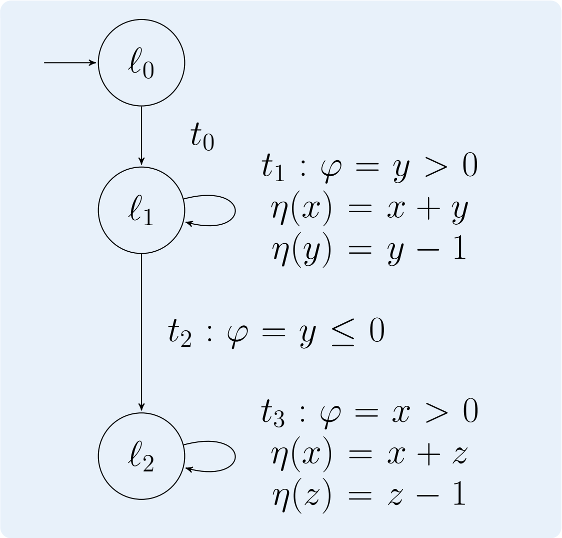
Goal: Infer (upper) runtime bounds for “real-world“ programs



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

Runtime Complexity of Integer Programs

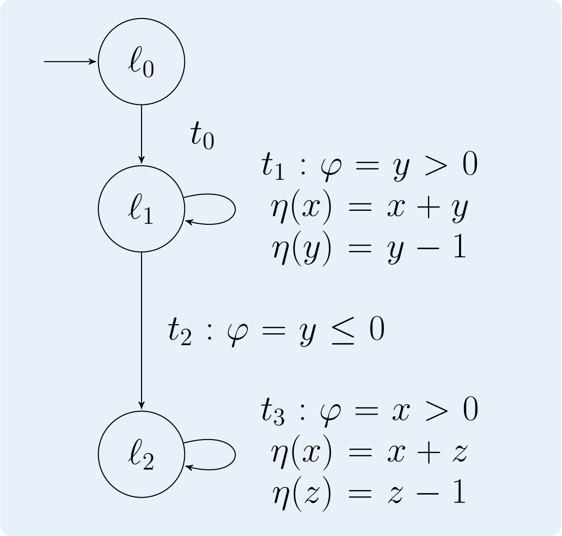
- How often are transitions evaluated in the worst case?



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

Runtime Complexity of Integer Programs

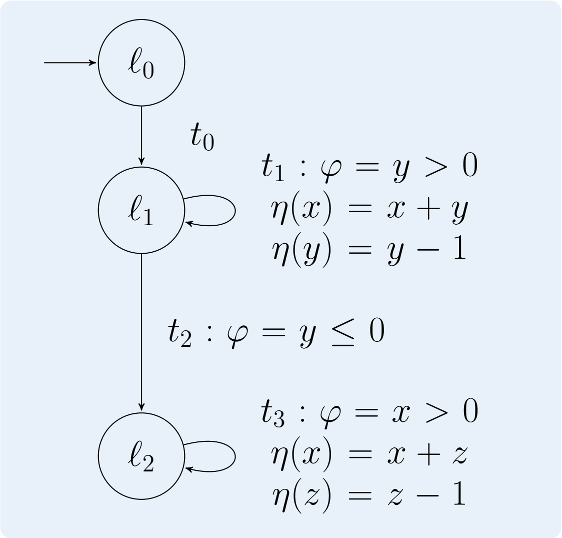
- How often are transitions evaluated in the worst case?
  - **Goal**: Bound runtime complexity rc for a state $\sigma : \{x, y, z\} \to \mathbb{Z}$



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

Runtime Complexity of Integer Programs

- How often are transitions evaluated in the worst case?
  - **Goal**: Bound runtime complexity rc for a state $\sigma : \{x, y, z\} \to \mathbb{Z}$
- Compute runtime bound $\mathcal{RB}(t_i)$ for each transition $t_0, \dots, t_3$

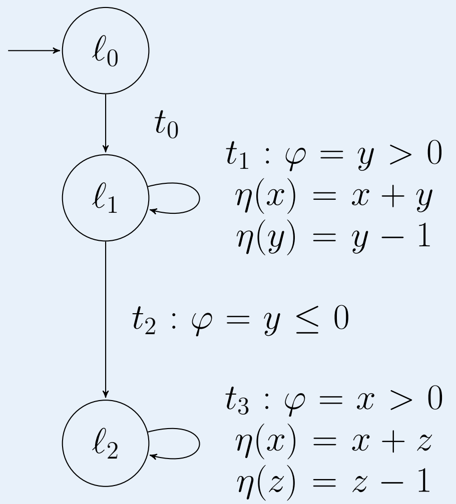


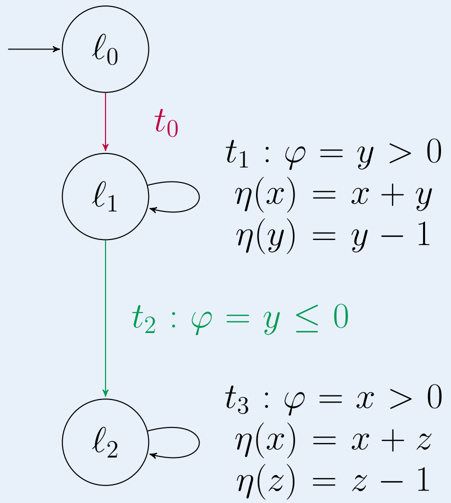
Jürgen Giesl, **Nils Lommen**, Marcel Hark, and Eleanore Meyer
RWTH Aachen University – LuFGi2

Runtime Complexity of Integer Programs

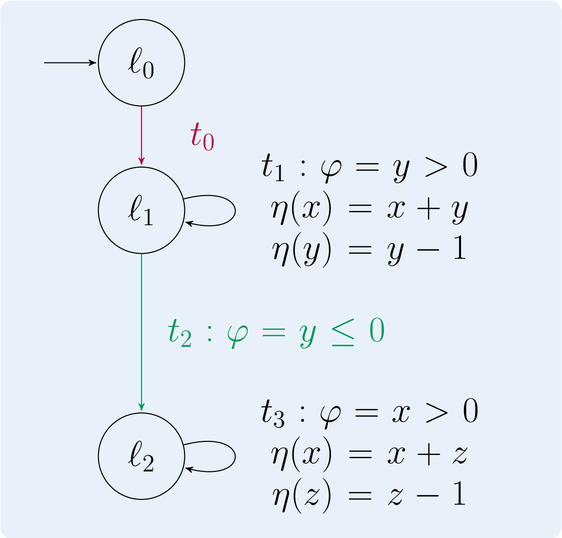
- How often are transitions evaluated in the worst case?
  - **Goal**: Bound runtime complexity rc for a state $\sigma : \{x, y, z\} \to \mathbb{Z}$
- Compute runtime bound $\mathcal{RB}(t_i)$ for each transition $t_0, \ldots, t_3$
- For all states $\sigma : \{x, y, z\} \to \mathbb{Z}$ we have

$$\mathrm{rc}(\sigma) \leq |\sigma| \left(\sum_{i=1}^{4} \mathcal{RB}(t_i) \right).$$




ℓ_0
t_0
$t_1 : \varphi = y > 0$
$\eta(x) = x + y$
$\eta(y) = y - 1$
ℓ_1
$t_2 : \varphi = y \leq 0$
$t_3 : \varphi = x > 0$
$\eta(x) = x + z$
$\eta(z) = z - 1$
ℓ_2

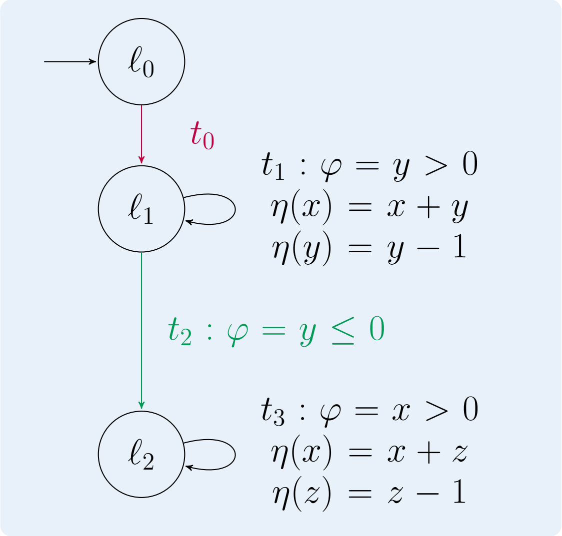
Runtime Complexity of Integer Programs



- t_0 and t_2 are not part of a cycle

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

Runtime Complexity of Integer Programs

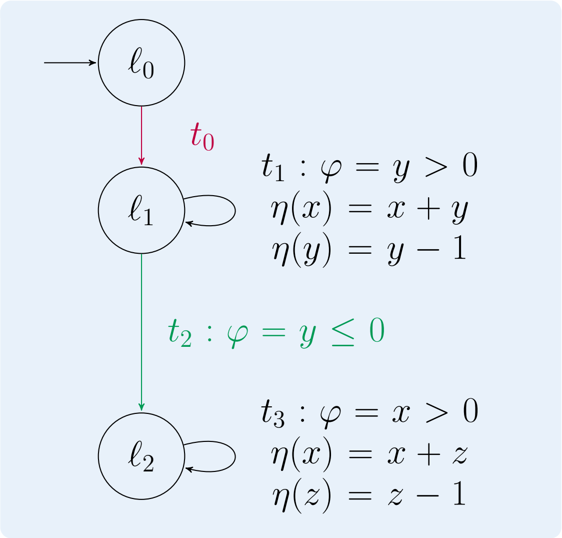


- t_0 and t_2 are not part of a cycle

  $\Rightarrow$ t_0 and t_2 are evaluated at most once

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

Runtime Complexity of Integer Programs



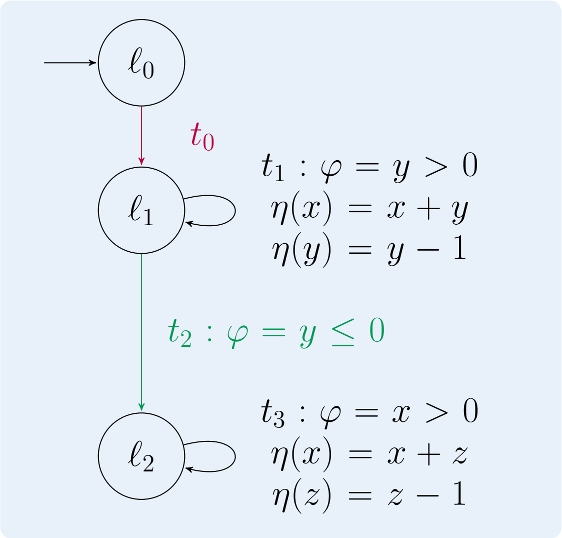
- Runtime bounds:
  - $\mathcal{RB}(t_0) = 1$
  - $\mathcal{RB}(t_1) = ?$
  - $\mathcal{RB}(t_2) = 1$
  - $\mathcal{RB}(t_3) = ?$

- t_0 and t_2 are not part of a cycle

  $\Rightarrow$ t_0 and t_2 are evaluated at most once

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

Runtime Complexity of Integer Programs



- Runtime bounds:
  - $\mathcal{RB}(t_0) = 1$
  - $\mathcal{RB}(t_1) = ?$
  - $\mathcal{RB}(t_2) = 1$
  - $\mathcal{RB}(t_3) = ?$
- $\mathcal{RB}(\mathcal{P}) = 1 + ? + 1 + ?$

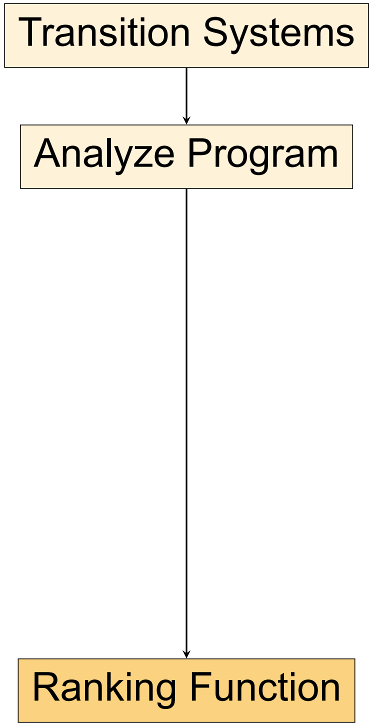
- t_0 and t_2 are not part of a cycle

  $\Rightarrow$ t_0 and t_2 are evaluated at most once

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

Overview

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



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

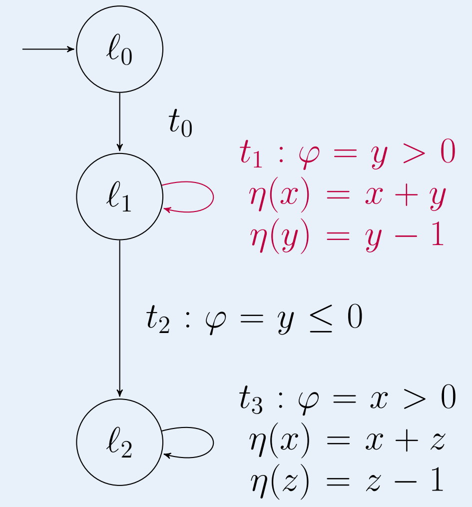
Runtime Bounds from Ranking Functions

Ranking function $\mathfrak{r}$ for program $\mathcal{P}$

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

Runtime Bounds from Ranking Functions

Ranking function $\mathfrak{r}$ for program $\mathcal{P}$

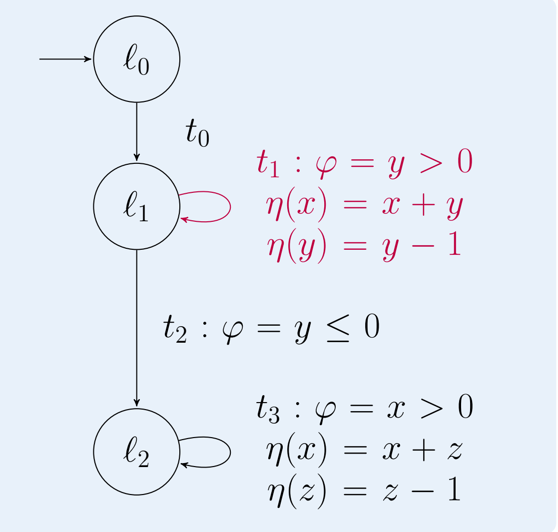


Jürgen Giesl, **Nils Lommen**, Marcel Hark, and Eleanore Meyer
RWTH Aachen University – LuFGi2

Runtime Bounds from Ranking Functions

Ranking function $\mathfrak{r}$ for program $\mathcal{P}$

- $\mathfrak{r}$ maps *locations* to $\mathbb{Z}[v_1, \ldots, v_n]$



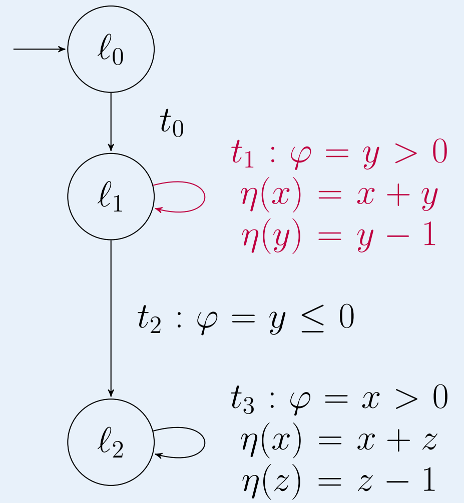
- Ranking Function: $\mathfrak{r}(\ell) = y$ for all locations ℓ

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

Runtime Bounds from Ranking Functions

Ranking function $\mathfrak{r}$ for program $\mathcal{P}$

- $\mathfrak{r}$ maps *locations* to $\mathbb{Z}[v_1, \ldots, v_n]$
- ***Non-Increase:*** no transition in $\mathcal{P}$ increases value of $\mathfrak{r}$



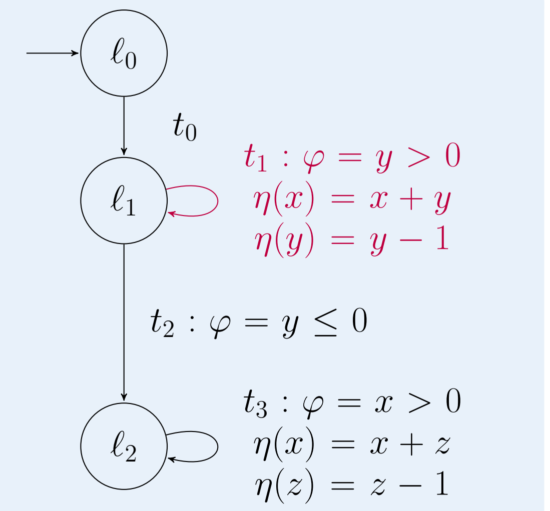
- Ranking Function: $\mathfrak{r}(\ell) = y$ for all locations ℓ

WST22
Jürgen Giesl, **Nils Lommen**, Marcel Hark, and Eleanore Meyer
RWTH Aachen University – LuFGi2

Runtime Bounds from Ranking Functions

Ranking function $\mathfrak{r}$ for program $\mathcal{P}$

- $\mathfrak{r}$ maps *locations* to $\mathbb{Z}[v_1, \ldots, v_n]$
- ***Non-Increase:*** no transition in $\mathcal{P}$ increases value of $\mathfrak{r}$
- ***Decrease:*** value of $\mathfrak{r}$ decreases by at least 1 for $\mathcal{P}_{\succ} \subseteq \mathcal{P}$



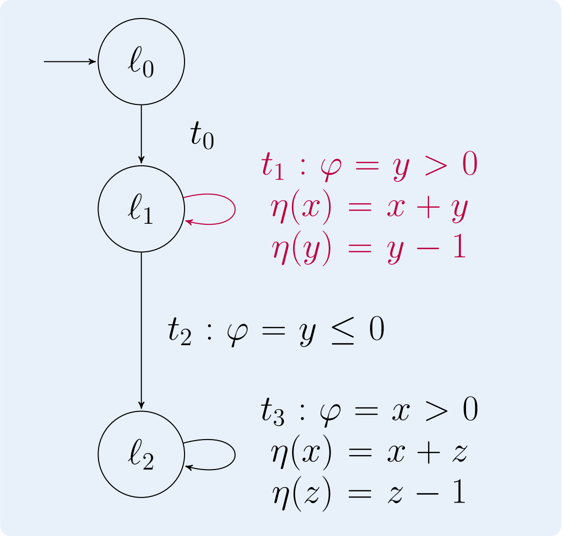
- Ranking Function: $\mathfrak{r}(\ell) = y$ for all locations ℓ

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

Runtime Bounds from Ranking Functions

Ranking function $\mathfrak{r}$ for program $\mathcal{P}$

- $\mathfrak{r}$ maps *locations* to $\mathbb{Z}[v_1, \dots, v_n]$
- ***Non-Increase:*** no transition in $\mathcal{P}$ increases value of $\mathfrak{r}$
- ***Decrease:*** value of $\mathfrak{r}$ decreases by at least 1 for $\mathcal{P}_\succ \subseteq \mathcal{P}$
- ***Boundedness:*** $\mathfrak{r} \geq 0$ after $\mathcal{P}_\succ \subseteq \mathcal{P}$



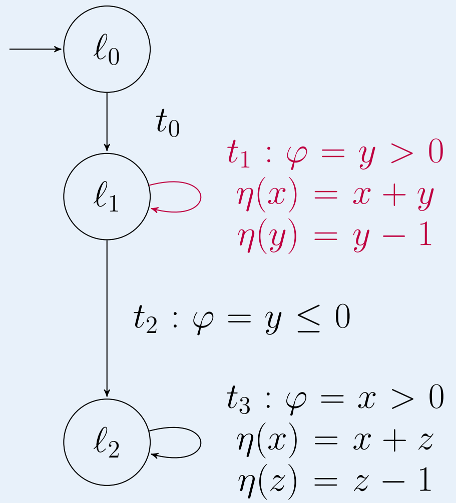
- Ranking Function: $\mathfrak{r}(\ell) = y$ for all locations ℓ

WST22
Jürgen Giesl, **Nils Lommen**, Marcel Hark, and Eleanore Meyer
RWTH Aachen University – LuFGi2

Runtime Bounds from Ranking Functions

Ranking function $\mathfrak{r}$ for program $\mathcal{P}$

- for all $t \in \mathcal{P}_\succ$, set $\mathcal{RB}(t) = \mathfrak{r}(\ell_0)$
- ***Non-Increase:*** no transition in $\mathcal{P}$ increases value of $\mathfrak{r}$
- ***Decrease:*** value of $\mathfrak{r}$ decreases by at least 1 for $\mathcal{P}_\succ \subseteq \mathcal{P}$
- ***Boundedness:*** $\mathfrak{r} \geq 0$ after $\mathcal{P}_\succ \subseteq \mathcal{P}$



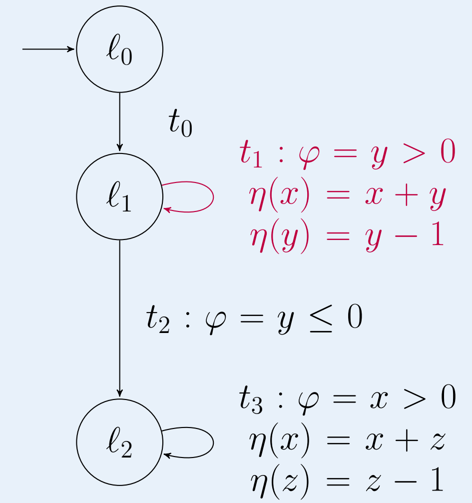
- Ranking Function: $\mathfrak{r}(\ell) = y$ for all locations ℓ

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

Runtime Bounds from Ranking Functions

Ranking function $\mathfrak{r}$ for program $\mathcal{P}$

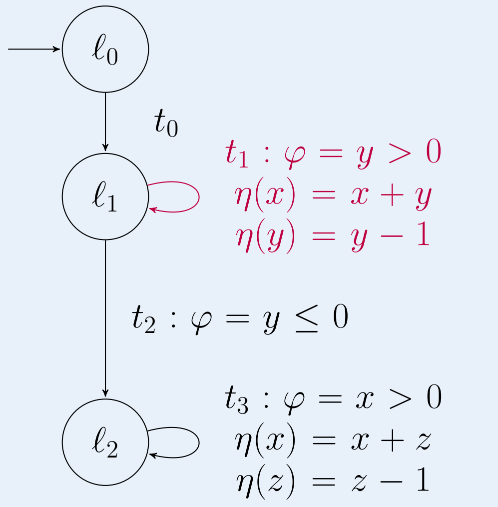
- for all $t \in \mathcal{P}_{\succ}$, set $\mathcal{RB}(t) = \mathfrak{r}(\ell_0)$
- ***Non-Increase:*** no transition in $\mathcal{P}$ increases value of $\mathfrak{r}$
- ***Decrease:*** value of $\mathfrak{r}$ decreases by at least 1 for $\mathcal{P}_{\succ} \subseteq \mathcal{P}$
- ***Boundedness:*** $\mathfrak{r} \geq 0$ after $\mathcal{P}_{\succ} \subseteq \mathcal{P}$



- Ranking Function: $\mathfrak{r}(\ell) = y$ for all locations ℓ
- By $t_1 \in \mathcal{P}_{\succ}$, we have $\mathcal{RB}(t_1) = y$.

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

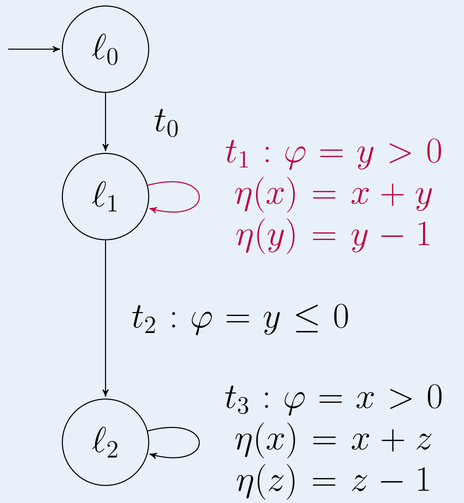
Runtime Complexity of Integer Programs



- Runtime bounds:
  - $\mathcal{RB}(t_0) = 1$
  - $\mathcal{RB}(t_1) = y$
  - $\mathcal{RB}(t_2) = 1$
  - $\mathcal{RB}(t_3) = ?$
- $\mathcal{RB}(\mathcal{P}) = 1 + y + 1 + ?$

WST22
Jürgen Giesl, **Nils Lommen**, Marcel Hark, and Eleanore Meyer
RWTH Aachen University – LuFGi2

Runtime Complexity of Integer Programs



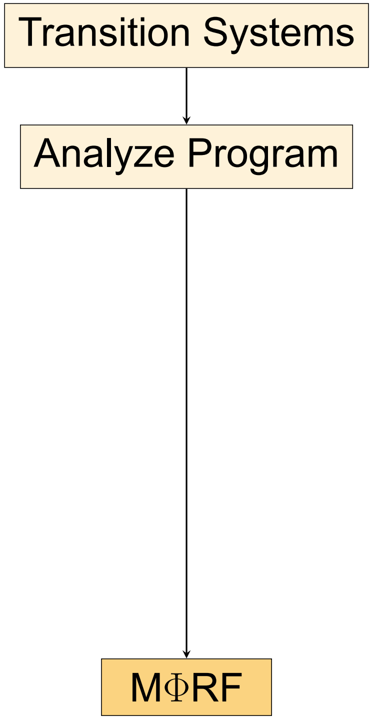
- Runtime bounds:
  - $\mathcal{RB}(t_0) = 1$
  - $\mathcal{RB}(t_1) = y$
  - $\mathcal{RB}(t_2) = 1$
  - $\mathcal{RB}(t_3) = ?$
- $\mathcal{RB}(\mathcal{P}) = 1 + y + 1 + ?$

- **Problem**: No linear ranking function for t_3

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

Overview

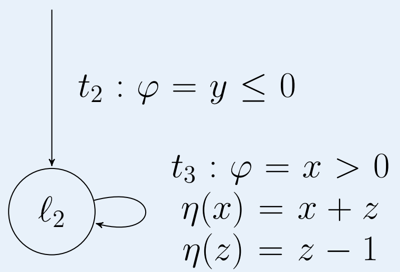
Goal: Infer (upper) runtime bounds for “real-world“ programs



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

Multiphase-Linear Ranking Functions for Loops

Consider program $\mathcal{P}'$

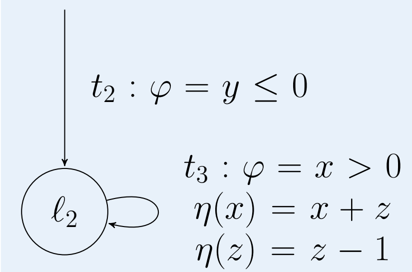


Jürgen Giesl, **Nils Lommen**, Marcel Hark, and Eleanore Meyer
RWTH Aachen University – LuFGi2

Multiphase-Linear Ranking Functions for Loops

Consider program $\mathcal{P}'$

- 2 phases:

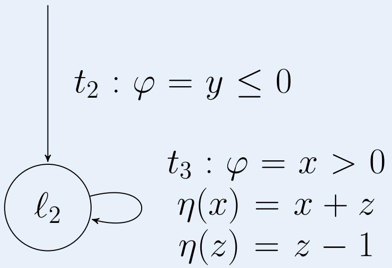


Jürgen Giesl, **Nils Lommen**, Marcel Hark, and Eleanore Meyer
RWTH Aachen University – LuFGi2

Multiphase-Linear Ranking Functions for Loops

Consider program $\mathcal{P}'$

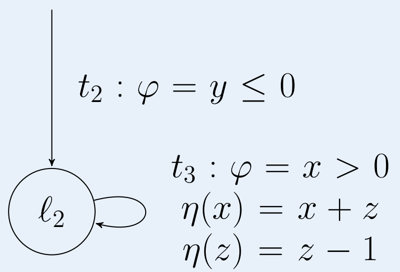
- 2 phases:
  1. z is decremented until $z < 0$



Multiphase-Linear Ranking Functions for Loops

Consider program $\mathcal{P}'$

- 2 phases:
  1. z is decremented until $z < 0$
  2. x is decremented until $x \leq 0$



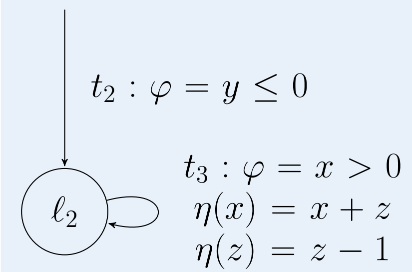
Jürgen Giesl, **Nils Lommen**, Marcel Hark, and Eleanore Meyer
RWTH Aachen University – LuFGi2

Multiphase-Linear Ranking Functions for Loops

Consider program $\mathcal{P}'$

- 2 phases:
  1. z is decremented until $z < 0$
  2. x is decremented until $x \leq 0$

  $\Rightarrow$ runtime is linear



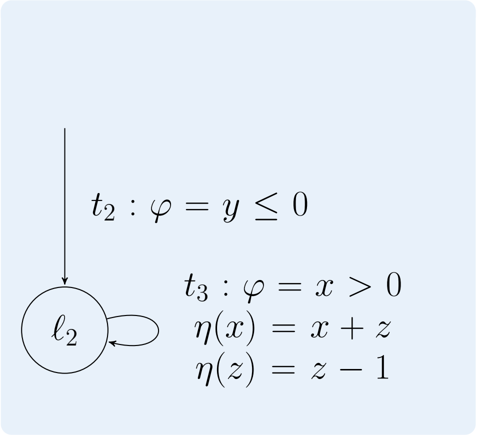
Jürgen Giesl, **Nils Lommen**, Marcel Hark, and Eleanore Meyer
RWTH Aachen University – LuFGi2

Multiphase-Linear Ranking Functions for Loops

Consider program $\mathcal{P}'$

- 2 phases:
  1. z is decremented until $z < 0$
  2. x is decremented until $x \leq 0$

  $\Rightarrow$ runtime is linear



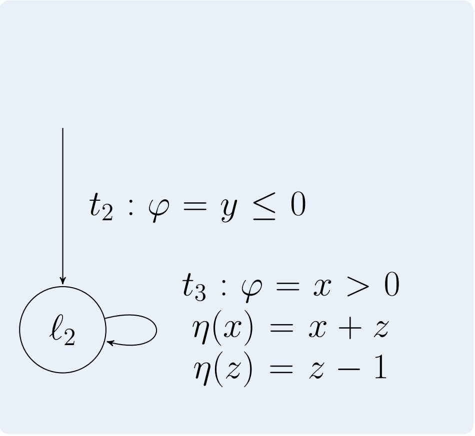
- **Multiphase-Linear Ranking Function (MΦRF)** [Ben-Amram, Genaim '17]

WST22
Jürgen Giesl, **Nils Lommen**, Marcel Hark, and Eleanore Meyer
RWTH Aachen University – LuFGi2

Consider program $\mathcal{P}'$

- 2 phases:
  1. z is decremented until $z < 0$
  2. x is decremented until $x \leq 0$

  $\Rightarrow$ runtime is linear



- **Multiphase-Linear Ranking Function (MΦRF)** [Ben-Amram, Genaim '17]

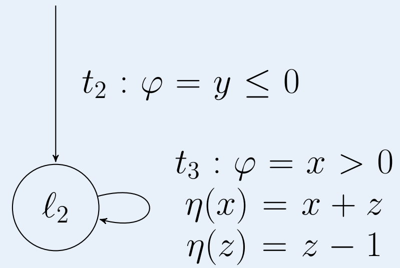
  $\Rightarrow$ every loop which admits MΦRF has **linear** runtime complexity

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

Runtime Bounds from MΦRFs for Loops

Ranking function $\mathfrak{r}$ for program $\mathcal{P}'$

- ***Non-Increase:*** no transition in $\mathcal{P}'$ increases value of $\mathfrak{r}$
- ***Decrease:*** value of $\mathfrak{r}$ decreases by at least 1 for $\mathcal{P}_{\succ} \subseteq \mathcal{P}'$
- ***Boundedness:*** $\mathfrak{r} \geq 0$ after $\mathcal{P}_{\succ} \subseteq \mathcal{P}'$

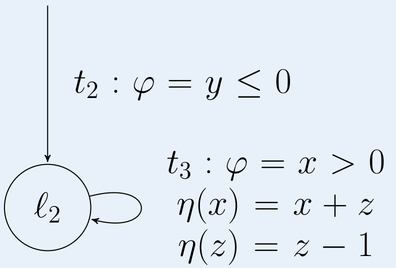


Jürgen Giesl, **Nils Lommen**, Marcel Hark, and Eleanore Meyer
RWTH Aachen University – LuFGi2

Runtime Bounds from MΦRFs for Loops

Ranking function $\mathfrak{r}$ for program $\mathcal{P}'$

- ***Non-Increase:*** no transition in $\mathcal{P}'$ increases value of $\mathfrak{r}$
- ***Decrease:*** value of $\mathfrak{r}$ decreases by at least 1 for $\mathcal{P}_{\succ} \subseteq \mathcal{P}'$
- ***Boundedness:*** $\mathfrak{r} \geq 0$ after $\mathcal{P}_{\succ} \subseteq \mathcal{P}'$



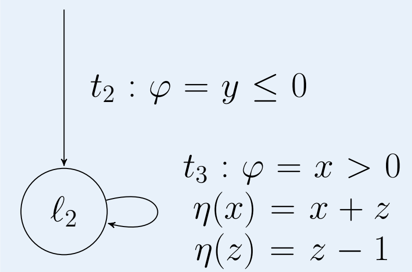
MΦRF $\mathfrak{r} = (\mathfrak{r}_1, \ldots, \mathfrak{r}_d)$ for program $\mathcal{P}'$

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

Runtime Bounds from MΦRFs for Loops

Ranking function $\mathfrak{r}$ for program $\mathcal{P}'$

- ***Non-Increase:*** no transition in $\mathcal{P}'$ increases value of $\mathfrak{r}$
- ***Decrease:*** value of $\mathfrak{r}$ decreases by at least 1 for $\mathcal{P}_{\succ} \subseteq \mathcal{P}'$
- ***Boundedness:*** $\mathfrak{r} \geq 0$ after $\mathcal{P}_{\succ} \subseteq \mathcal{P}'$



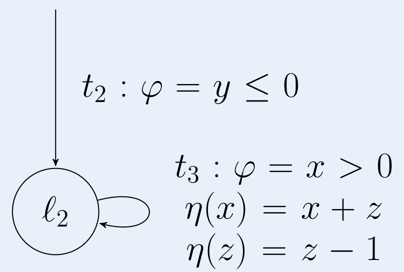
MΦRF $\mathfrak{r} = (\mathfrak{r}_1, \ldots, \mathfrak{r}_d)$ for program $\mathcal{P}'$

- ***Non-Increase:*** no transition in $\mathcal{P}' \setminus \mathcal{P}_{\succ}$ increases value of $\mathfrak{r}_1, \ldots, \mathfrak{r}_d$

WST22
Jürgen Giesl, **Nils Lommen**, Marcel Hark, and Eleanore Meyer
RWTH Aachen University – LuFGi2

Runtime Bounds from MΦRFs for Loops

- MΦRF: $\mathfrak{r}_1(\ell_2) = z + 1$ and $\mathfrak{r}_2(\ell_2) = x$



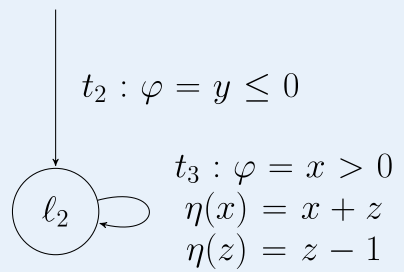
MΦRF $\mathfrak{r} = (\mathfrak{r}_1, \ldots, \mathfrak{r}_d)$ for program $\mathcal{P}'$

- ***Non-Increase:*** no transition in $\mathcal{P}' \setminus \mathcal{P}_\succ$ increases value of $\mathfrak{r}_1, \ldots, \mathfrak{r}_d$

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

Runtime Bounds from MΦRFs for Loops

- MΦRF: $\mathfrak{r}_1(\ell_2) = z + 1$ and $\mathfrak{r}_2(\ell_2) = x$
- $\mathcal{P}' = \{t_2, t_3\}$ and $\mathcal{P}_{\succ} = \{t_3\}$



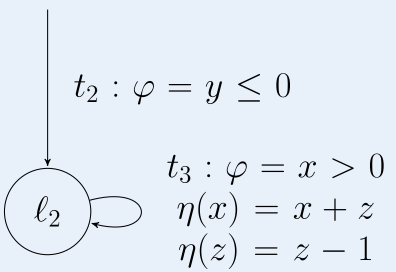
MΦRF $\mathfrak{r} = (\mathfrak{r}_1, \dots, \mathfrak{r}_d)$ for program $\mathcal{P}'$

- ***Non-Increase:*** no transition in $\mathcal{P}' \setminus \mathcal{P}_{\succ}$ increases value of $\mathfrak{r}_1, \dots, \mathfrak{r}_d$

WST22
Jürgen Giesl, **Nils Lommen**, Marcel Hark, and Eleanore Meyer
RWTH Aachen University – LuFGi2

Runtime Bounds from MΦRFs for Loops

- MΦRF: $\mathfrak{r}_1(\ell_2) = z + 1$ and $\mathfrak{r}_2(\ell_2) = x$
- $\mathcal{P}' = \{t_2, t_3\}$ and $\mathcal{P}_\succ = \{t_3\}$
- Non-Increase ✓



MΦRF $\mathfrak{r} = (\mathfrak{r}_1, \ldots, \mathfrak{r}_d)$ for program $\mathcal{P}'$

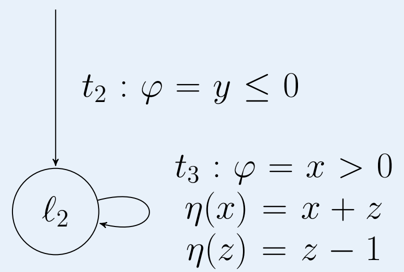
- ***Non-Increase:*** no transition in $\mathcal{P}' \setminus \mathcal{P}_\succ$ increases value of $\mathfrak{r}_1, \ldots, \mathfrak{r}_d$

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

Runtime Bounds from MΦRFs for Loops

Ranking function $\mathfrak{r}$ for program $\mathcal{P}'$

- ***Non-Increase:*** no transition in $\mathcal{P}'$ increases value of $\mathfrak{r}$
- ***Decrease:*** value of $\mathfrak{r}$ decreases by at least 1 for $\mathcal{P}_{\succ} \subseteq \mathcal{P}'$
- ***Boundedness:*** $\mathfrak{r} \geq 0$ after $\mathcal{P}_{\succ} \subseteq \mathcal{P}'$



MΦRF $\mathfrak{r} = (\mathfrak{r}_1, \ldots, \mathfrak{r}_d)$ for program $\mathcal{P}'$

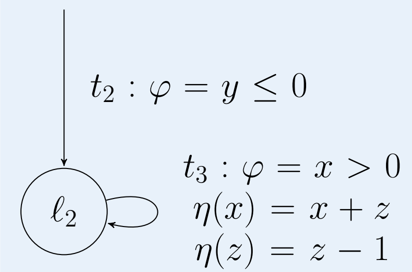
- ***Non-Increase:*** no transition in $\mathcal{P}' \setminus \mathcal{P}_{\succ}$ increases value of $\mathfrak{r}_1, \ldots, \mathfrak{r}_d$

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

Runtime Bounds from MΦRFs for Loops

Ranking function $\mathfrak{r}$ for program $\mathcal{P}'$

- ***Non-Increase:*** no transition in $\mathcal{P}'$ increases value of $\mathfrak{r}$
- ***Decrease*** for $t \in \mathcal{P}_{\succ} \subseteq \mathcal{P}'$:
  $\mathfrak{r}$ before $t \quad \geq \quad 1 + \mathfrak{r}$ after t
- ***Boundedness:*** $\mathfrak{r} \geq 0$ after $\mathcal{P}_{\succ} \subseteq \mathcal{P}'$



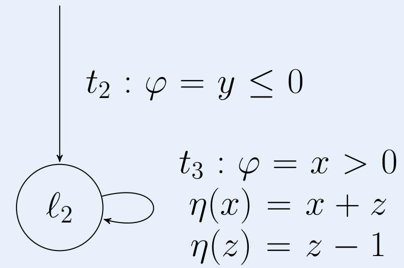
MΦRF $\mathfrak{r} = (\mathfrak{r}_1, \ldots, \mathfrak{r}_d)$ for program $\mathcal{P}'$

- ***Non-Increase:*** no transition in $\mathcal{P}' \setminus \mathcal{P}_{\succ}$ increases value of $\mathfrak{r}_1, \ldots, \mathfrak{r}_d$

Runtime Bounds from MΦRFs for Loops

Ranking function $\mathfrak{r}$ for program $\mathcal{P}'$

- ***Non-Increase:*** no transition in $\mathcal{P}'$ increases value of $\mathfrak{r}$
- ***Decrease*** for $t \in \mathcal{P}_{\succ} \subseteq \mathcal{P}'$:
  $\mathfrak{r}$ before $t \quad \geq \quad 1 + \mathfrak{r}$ after t
- ***Boundedness:*** $\mathfrak{r} \geq 0$ after $\mathcal{P}_{\succ} \subseteq \mathcal{P}'$



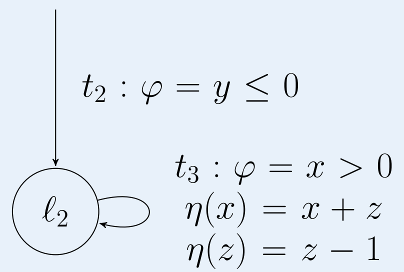
MΦRF $\mathfrak{r} = (\mathfrak{r}_1, \ldots, \mathfrak{r}_d)$ for program $\mathcal{P}'$

- ***Non-Increase:*** no transition in $\mathcal{P}' \setminus \mathcal{P}_{\succ}$ increases value of $\mathfrak{r}_1, \ldots, \mathfrak{r}_d$
- ***Decrease*** for $t \in \mathcal{P}_{\succ} \subseteq \mathcal{P}'$: $\mathfrak{r}_{i-1} + \mathfrak{r}_i$ before $t \quad \geq \quad 1 + \mathfrak{r}_i$ after t, $\mathfrak{r}_0 = 0$

WST22
Jürgen Giesl, **Nils Lommen**, Marcel Hark, and Eleanore Meyer
RWTH Aachen University – LuFGi2

Runtime Bounds from MΦRFs for Loops

- MΦRF: $\mathfrak{r}_1(\ell_2) = z + 1$ and $\mathfrak{r}_2(\ell_2) = x$
- $\mathcal{P}' = \{t_2, t_3\}$ and $\mathcal{P}_\succ = \{t_3\}$
- Non-Increase ✓



MΦRF $\mathfrak{r} = (\mathfrak{r}_1, \dots, \mathfrak{r}_d)$ for program $\mathcal{P}'$

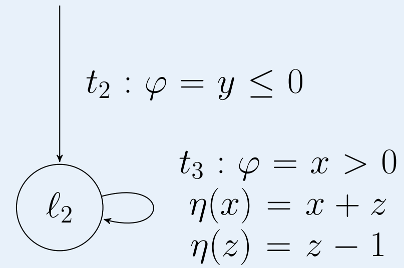
- ***Non-Increase:*** no transition in $\mathcal{P}' \setminus \mathcal{P}_\succ$ increases value of $\mathfrak{r}_1, \dots, \mathfrak{r}_d$
- ***Decrease*** for $t \in \mathcal{P}_\succ \subseteq \mathcal{P}'$: $\mathfrak{r}_{i-1} + \mathfrak{r}_i$ before $t \quad \geq \quad 1 + \mathfrak{r}_i$ after t, $\mathfrak{r}_0 = 0$

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

Runtime Bounds from MΦRFs for Loops

- MΦRF: $\mathfrak{r}_1(\ell_2) = z + 1$ and $\mathfrak{r}_2(\ell_2) = x$
- $\mathcal{P}' = \{t_2, t_3\}$ and $\mathcal{P}_{\succ} = \{t_3\}$
- Non-Increase ✓

$$\mathfrak{r}_0 + \mathfrak{r}_1 \text{ before } t_3 \quad \geq \quad 1 + \mathfrak{r}_1 \text{ after } t_3$$
$$\mathfrak{r}_1 + \mathfrak{r}_2 \text{ before } t_3 \quad \geq \quad 1 + \mathfrak{r}_2 \text{ after } t_3$$



MΦRF $\mathfrak{r} = (\mathfrak{r}_1, \ldots, \mathfrak{r}_d)$ for program $\mathcal{P}'$

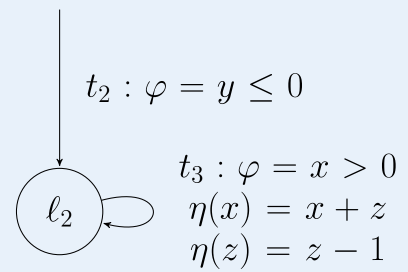
- ***Non-Increase:*** no transition in $\mathcal{P}' \setminus \mathcal{P}_{\succ}$ increases value of $\mathfrak{r}_1, \ldots, \mathfrak{r}_d$
- ***Decrease*** for $t \in \mathcal{P}_{\succ} \subseteq \mathcal{P}'$: $\mathfrak{r}_{i-1} + \mathfrak{r}_i$ before $t \quad \geq \quad 1 + \mathfrak{r}_i$ after t, $\mathfrak{r}_0 = 0$

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

Runtime Bounds from MΦRFs for Loops

- MΦRF: $\mathfrak{r}_1(\ell_2) = z + 1$ and $\mathfrak{r}_2(\ell_2) = x$
- $\mathcal{P}' = \{t_2, t_3\}$ and $\mathcal{P}_{\succ} = \{t_3\}$
- Non-Increase ✓

$$\begin{array}{lcl} \mathtt{z} + 1 & \geq & 1 + \mathfrak{r}_1 \text{ after } t_3 \\ \mathfrak{r}_1 + \mathfrak{r}_2 \text{ before } t_3 & \geq & 1 + \mathfrak{r}_2 \text{ after } t_3 \end{array}$$



MΦRF $\mathfrak{r} = (\mathfrak{r}_1, \dots, \mathfrak{r}_d)$ for program $\mathcal{P}'$

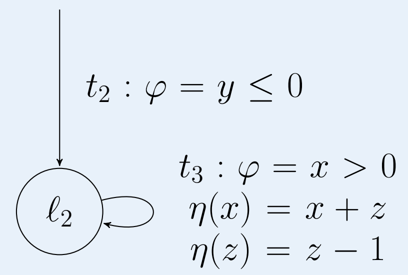
- ***Non-Increase:*** no transition in $\mathcal{P}' \setminus \mathcal{P}_{\succ}$ increases value of $\mathfrak{r}_1, \dots, \mathfrak{r}_d$
- ***Decrease*** for $t \in \mathcal{P}_{\succ} \subseteq \mathcal{P}'$: $\mathfrak{r}_{i-1} + \mathfrak{r}_i$ before $t \quad \geq \quad 1 + \mathfrak{r}_i$ after t, $\mathfrak{r}_0 = 0$

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

Runtime Bounds from MΦRFs for Loops

- MΦRF: $\mathfrak{r}_1(\ell_2) = z + 1$ and $\mathfrak{r}_2(\ell_2) = x$
- $\mathcal{P}' = \{t_2, t_3\}$ and $\mathcal{P}_\succ = \{t_3\}$
- Non-Increase ✓

$$\begin{array}{lcl} \texttt{z} + 1 & \geq & 1 + \texttt{z} - 1 + 1 \\ \mathfrak{r}_1 + \mathfrak{r}_2 \text{ before } t_3 & \geq & 1 + \mathfrak{r}_2 \text{ after } t_3 \end{array}$$



MΦRF $\mathfrak{r} = (\mathfrak{r}_1, \ldots, \mathfrak{r}_d)$ for program $\mathcal{P}'$

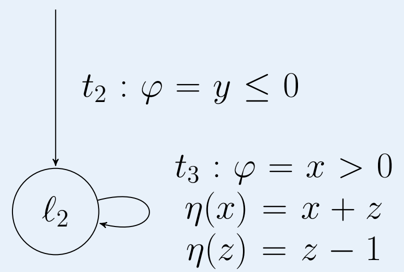
- ***Non-Increase:*** no transition in $\mathcal{P}' \setminus \mathcal{P}_\succ$ increases value of $\mathfrak{r}_1, \ldots, \mathfrak{r}_d$
- ***Decrease*** for $t \in \mathcal{P}_\succ \subseteq \mathcal{P}'$: $\mathfrak{r}_{i-1} + \mathfrak{r}_i$ before $t \quad \geq \quad 1 + \mathfrak{r}_i$ after t, $\mathfrak{r}_0 = 0$

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

Runtime Bounds from MΦRFs for Loops

- MΦRF: $\mathfrak{r}_1(\ell_2) = z + 1$ and $\mathfrak{r}_2(\ell_2) = x$
- $\mathcal{P}' = \{t_2, t_3\}$ and $\mathcal{P}_{\succ} = \{t_3\}$
- Non-Increase ✓

$$\mathtt{z} + 1 \quad \geq \quad 1 + \mathtt{z} - 1 + 1$$
$$\mathtt{z} + 1 + \mathtt{x} \quad \geq \quad 1 + \mathfrak{r}_2 \text{ after } t_3$$



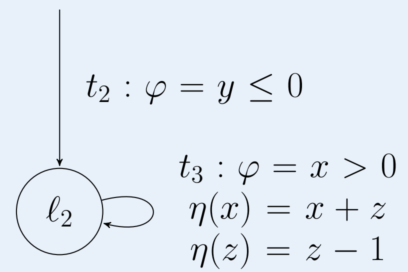
MΦRF $\mathfrak{r} = (\mathfrak{r}_1, \ldots, \mathfrak{r}_d)$ for program $\mathcal{P}'$

- ***Non-Increase:*** no transition in $\mathcal{P}' \setminus \mathcal{P}_{\succ}$ increases value of $\mathfrak{r}_1, \ldots, \mathfrak{r}_d$
- ***Decrease*** for $t \in \mathcal{P}_{\succ} \subseteq \mathcal{P}'$: $\mathfrak{r}_{i-1} + \mathfrak{r}_i$ before $t \quad \geq \quad 1 + \mathfrak{r}_i$ after t, $\mathfrak{r}_0 = 0$

Runtime Bounds from MΦRFs for Loops

- MΦRF: $\mathfrak{r}_1(\ell_2) = z + 1$ and $\mathfrak{r}_2(\ell_2) = x$
- $\mathcal{P}' = \{t_2, t_3\}$ and $\mathcal{P}_{\succ} = \{t_3\}$
- Non-Increase ✓

$$\begin{array}{lcl} \mathtt{z} + 1 & \geq & 1 + \mathtt{z} - 1 + 1 \\ \mathtt{z} + 1 + \mathtt{x} & \geq & 1 + \mathtt{x} + \mathtt{z} \end{array}$$



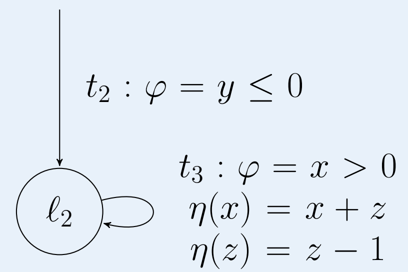
MΦRF $\mathfrak{r} = (\mathfrak{r}_1, \ldots, \mathfrak{r}_d)$ for program $\mathcal{P}'$

- ***Non-Increase:*** no transition in $\mathcal{P}' \setminus \mathcal{P}_{\succ}$ increases value of $\mathfrak{r}_1, \ldots, \mathfrak{r}_d$
- ***Decrease*** for $t \in \mathcal{P}_{\succ} \subseteq \mathcal{P}'$: $\mathfrak{r}_{i-1} + \mathfrak{r}_i$ before $t \quad \geq \quad 1 + \mathfrak{r}_i$ after t, $\mathfrak{r}_0 = 0$

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

Runtime Bounds from MΦRFs for Loops

- MΦRF: $\mathfrak{r}_1(\ell_2) = z + 1$ and $\mathfrak{r}_2(\ell_2) = x$
- $\mathcal{P}' = \{t_2, t_3\}$ and $\mathcal{P}_\succ = \{t_3\}$
- Non-Increase ✓
- Decrease ✓



MΦRF $\mathfrak{r} = (\mathfrak{r}_1, \ldots, \mathfrak{r}_d)$ for program $\mathcal{P}'$

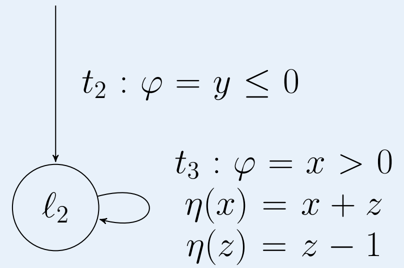
- ***Non-Increase:*** no transition in $\mathcal{P}' \setminus \mathcal{P}_\succ$ increases value of $\mathfrak{r}_1, \ldots, \mathfrak{r}_d$
- ***Decrease*** for $t \in \mathcal{P}_\succ \subseteq \mathcal{P}'$: $\mathfrak{r}_{i-1} + \mathfrak{r}_i$ before $t \quad \geq \quad 1 + \mathfrak{r}_i$ after t, $\mathfrak{r}_0 = 0$

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

Runtime Bounds from MΦRFs for Loops

Ranking function $\mathfrak{r}$ for program $\mathcal{P}'$

- ***Non-Increase:*** no transition in $\mathcal{P}'$ increases value of $\mathfrak{r}$
- ***Decrease*** for $t \in \mathcal{P}_{\succ} \subseteq \mathcal{P}'$:
  $\mathfrak{r}$ before $t \quad \geq \quad 1 + \mathfrak{r}$ after t
- ***Boundedness:*** $\mathfrak{r} \geq 0$ after $\mathcal{P}_{\succ} \subseteq \mathcal{P}'$



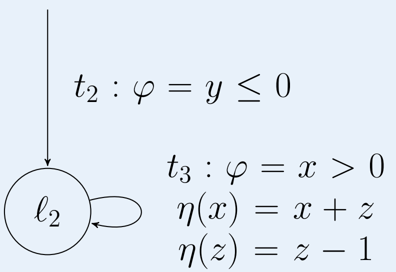
MΦRF $\mathfrak{r} = (\mathfrak{r}_1, \ldots, \mathfrak{r}_d)$ for program $\mathcal{P}'$

- ***Non-Increase:*** no transition in $\mathcal{P}' \setminus \mathcal{P}_{\succ}$ increases value of $\mathfrak{r}_1, \ldots, \mathfrak{r}_d$
- ***Decrease*** for $t \in \mathcal{P}_{\succ} \subseteq \mathcal{P}'$: $\mathfrak{r}_{i-1} + \mathfrak{r}_i$ before $t \quad \geq \quad 1 + \mathfrak{r}_i$ after t, $\mathfrak{r}_0 = 0$

Runtime Bounds from MΦRFs for Loops

Ranking function $\mathfrak{r}$ for program $\mathcal{P}'$

- ***Non-Increase:*** no transition in $\mathcal{P}'$ increases value of $\mathfrak{r}$
- ***Decrease*** for $t \in \mathcal{P}_\succ \subseteq \mathcal{P}'$:
  $\mathfrak{r}$ before $t \quad \geq \quad 1 + \mathfrak{r}$ after t
- ***Boundedness:*** $\mathfrak{r} \geq 0$ after $\mathcal{P}_\succ \subseteq \mathcal{P}'$



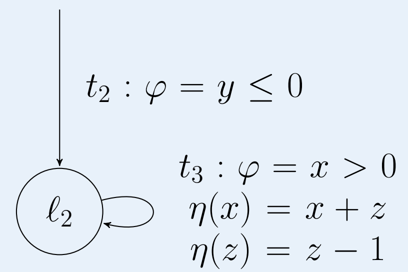
MΦRF $\mathfrak{r} = (\mathfrak{r}_1, \ldots, \mathfrak{r}_d)$ for program $\mathcal{P}'$

- ***Non-Increase:*** no transition in $\mathcal{P}' \setminus \mathcal{P}_\succ$ increases value of $\mathfrak{r}_1, \ldots, \mathfrak{r}_d$
- ***Decrease*** for $t \in \mathcal{P}_\succ \subseteq \mathcal{P}'$: $\mathfrak{r}_{i-1} + \mathfrak{r}_i$ before $t \quad \geq \quad 1 + \mathfrak{r}_i$ after t, $\mathfrak{r}_0 = 0$
- ***Boundedness:*** $\mathfrak{r}_d \geq 0$ before $\mathcal{P}_\succ \subseteq \mathcal{P}'$

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

Runtime Bounds from MΦRFs for Loops

- MΦRF: $\mathfrak{r}_1(\ell_2) = z + 1$ and $\mathfrak{r}_2(\ell_2) = x$
- $\mathcal{P}' = \{t_2, t_3\}$ and $\mathcal{P}_\succ = \{t_3\}$
- Non-Increase ✓
- Decrease ✓
- Boundedness



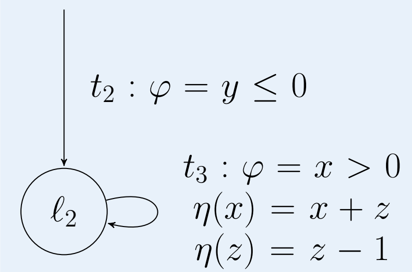
MΦRF $\mathfrak{r} = (\mathfrak{r}_1, \ldots, \mathfrak{r}_d)$ for program $\mathcal{P}'$

- ***Non-Increase:*** no transition in $\mathcal{P}' \setminus \mathcal{P}_\succ$ increases value of $\mathfrak{r}_1, \ldots, \mathfrak{r}_d$
- ***Decrease*** for $t \in \mathcal{P}_\succ \subseteq \mathcal{P}'$: $\mathfrak{r}_{i-1} + \mathfrak{r}_i$ before $t \quad \geq \quad 1 + \mathfrak{r}_i$ after t, $\mathfrak{r}_0 = 0$
- ***Boundedness:*** $\mathfrak{r}_d \geq 0$ before $\mathcal{P}_\succ \subseteq \mathcal{P}'$

WST22
Jürgen Giesl, **Nils Lommen**, Marcel Hark, and Eleanore Meyer
RWTH Aachen University – LuFGi2

Runtime Bounds from MΦRFs for Loops

- MΦRF: $\mathfrak{r}_1(\ell_2) = z + 1$ and $\mathfrak{r}_2(\ell_2) = x$
- $\mathcal{P}' = \{t_2, t_3\}$ and $\mathcal{P}_{\succ} = \{t_3\}$
- Non-Increase ✓
- Decrease ✓
- Boundedness $\mathfrak{r}_2 \geq 0$ before t_3



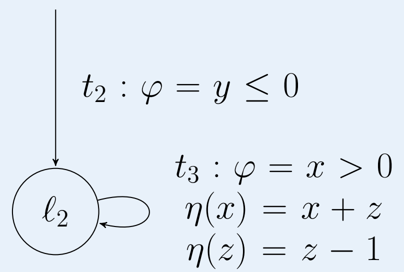
MΦRF $\mathfrak{r} = (\mathfrak{r}_1, \ldots, \mathfrak{r}_d)$ for program $\mathcal{P}'$

- ***Non-Increase:*** no transition in $\mathcal{P}' \setminus \mathcal{P}_{\succ}$ increases value of $\mathfrak{r}_1, \ldots, \mathfrak{r}_d$
- ***Decrease*** for $t \in \mathcal{P}_{\succ} \subseteq \mathcal{P}'$: $\mathfrak{r}_{i-1} + \mathfrak{r}_i$ before $t \quad \geq \quad 1 + \mathfrak{r}_i$ after t, $\mathfrak{r}_0 = 0$
- ***Boundedness:*** $\mathfrak{r}_d \geq 0$ before $\mathcal{P}_{\succ} \subseteq \mathcal{P}'$

WST22
Jürgen Giesl, **Nils Lommen**, Marcel Hark, and Eleanore Meyer
RWTH Aachen University – LuFGi2

Runtime Bounds from MΦRFs for Loops

- MΦRF: $\mathfrak{r}_1(\ell_2) = z + 1$ and $\mathfrak{r}_2(\ell_2) = x$
- $\mathcal{P}' = \{t_2, t_3\}$ and $\mathcal{P}_{\succ} = \{t_3\}$
- Non-Increase ✓
- Decrease ✓
- Boundedness `x` ≥ 0 before t_3



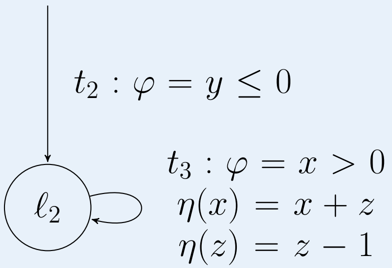
MΦRF $\mathfrak{r} = (\mathfrak{r}_1, \ldots, \mathfrak{r}_d)$ for program $\mathcal{P}'$

- ***Non-Increase:*** no transition in $\mathcal{P}' \setminus \mathcal{P}_{\succ}$ increases value of $\mathfrak{r}_1, \ldots, \mathfrak{r}_d$
- ***Decrease*** for $t \in \mathcal{P}_{\succ} \subseteq \mathcal{P}'$: $\mathfrak{r}_{i-1} + \mathfrak{r}_i$ before $t \quad \geq \quad 1 + \mathfrak{r}_i$ after t, $\mathfrak{r}_0 = 0$
- ***Boundedness:*** $\mathfrak{r}_d \geq 0$ before $\mathcal{P}_{\succ} \subseteq \mathcal{P}'$

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

Runtime Bounds from MΦRFs for Loops

- MΦRF: $\mathfrak{r}_1(\ell_2) = z + 1$ and $\mathfrak{r}_2(\ell_2) = x$
- $\mathcal{P}' = \{t_2, t_3\}$ and $\mathcal{P}_\succ = \{t_3\}$
- Non-Increase ✓
- Decrease ✓
- Boundedness ✓



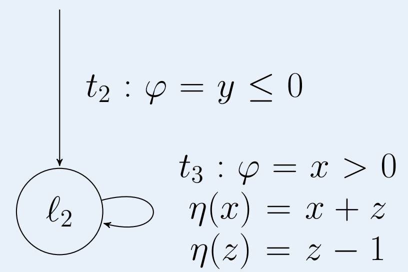
MΦRF $\mathfrak{r} = (\mathfrak{r}_1, \ldots, \mathfrak{r}_d)$ for program $\mathcal{P}'$

- ***Non-Increase:*** no transition in $\mathcal{P}' \setminus \mathcal{P}_\succ$ increases value of $\mathfrak{r}_1, \ldots, \mathfrak{r}_d$
- ***Decrease*** for $t \in \mathcal{P}_\succ \subseteq \mathcal{P}'$: $\mathfrak{r}_{i-1} + \mathfrak{r}_i$ before $t \quad \geq \quad 1 + \mathfrak{r}_i$ after t, $\mathfrak{r}_0 = 0$
- ***Boundedness:*** $\mathfrak{r}_d \geq 0$ before $\mathcal{P}_\succ \subseteq \mathcal{P}'$

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

Runtime Bounds from MΦRFs for Loops

- MΦRF: $\mathfrak{r}_1(\ell_2) = z + 1$ and $\mathfrak{r}_2(\ell_2) = x$
- $\mathcal{P}' = \{t_2, t_3\}$ and $\mathcal{P}_{\succ} = \{t_3\}$
- Non-Increase ✓
- Decrease ✓
- Boundedness ✓



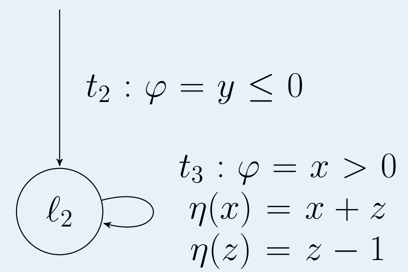
MΦRF $\mathfrak{r} = (\mathfrak{r}_1, \ldots, \mathfrak{r}_d)$ for program $\mathcal{P}'$

- for all $t \in \mathcal{P}_{\succ}$, set $\mathcal{RB}(\mathcal{P}', t) = 1 + c_d \cdot (\mathfrak{r}_1(\ell_2) + \ldots + \mathfrak{r}_d(\ell_2))$
- ***Non-Increase:*** no transition in $\mathcal{P}' \setminus \mathcal{P}_{\succ}$ increases value of $\mathfrak{r}_1, \ldots, \mathfrak{r}_d$
- ***Decrease*** for $t \in \mathcal{P}_{\succ} \subseteq \mathcal{P}'$: $\mathfrak{r}_{i-1} + \mathfrak{r}_i$ before $t \quad \geq \quad 1 + \mathfrak{r}_i$ after t, $\mathfrak{r}_0 = 0$
- ***Boundedness:*** $\mathfrak{r}_d \geq 0$ before $\mathcal{P}_{\succ} \subseteq \mathcal{P}'$

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

Runtime Bounds from MΦRFs for Loops

- MΦRF: $\mathfrak{r}_1(\ell_2) = z + 1$ and $\mathfrak{r}_2(\ell_2) = x$
- $\mathcal{P}' = \{t_2, t_3\}$ and $\mathcal{P}_\succ = \{t_3\}$
- Non-Increase ✓
- Decrease ✓
- Boundedness ✓
- $\mathcal{RB}(\mathcal{P}', t_3) = 1 + 8 \cdot (z + 1 + x)$



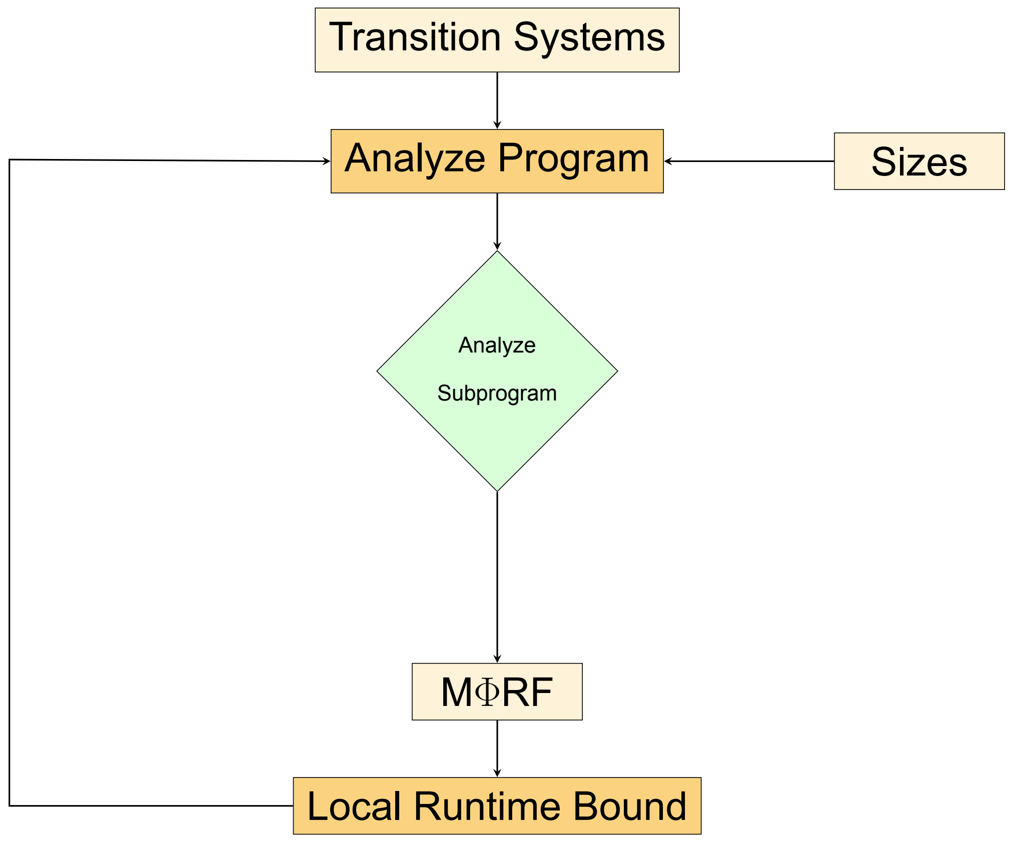
MΦRF $\mathfrak{r} = (\mathfrak{r}_1, \ldots, \mathfrak{r}_d)$ for program $\mathcal{P}'$

- for all $t \in \mathcal{P}_\succ$, set $\mathcal{RB}(\mathcal{P}', t) = 1 + c_d \cdot (\mathfrak{r}_1(\ell_2) + \ldots + \mathfrak{r}_d(\ell_2))$
- ***Non-Increase:*** no transition in $\mathcal{P}' \setminus \mathcal{P}_\succ$ increases value of $\mathfrak{r}_1, \ldots, \mathfrak{r}_d$
- ***Decrease*** for $t \in \mathcal{P}_\succ \subseteq \mathcal{P}'$: $\mathfrak{r}_{i-1} + \mathfrak{r}_i$ before $t \quad \geq \quad 1 + \mathfrak{r}_i$ after t, $\mathfrak{r}_0 = 0$
- ***Boundedness:*** $\mathfrak{r}_d \geq 0$ before $\mathcal{P}_\succ \subseteq \mathcal{P}'$

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

Overview

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



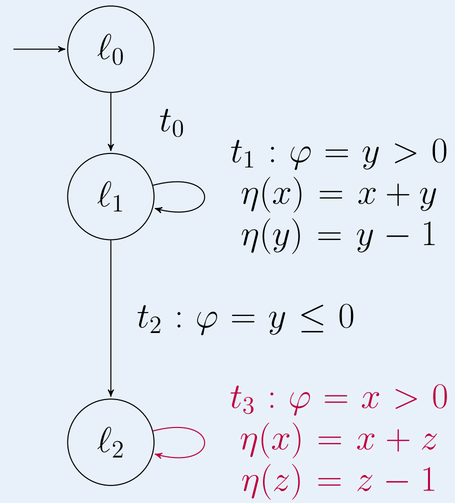
Jürgen Giesl, **Nils Lommen**, Marcel Hark, and Eleanore Meyer
RWTH Aachen University – LuFGi2

Modular Runtime Bounds from MΦRFs

Lift Runtime Bound $\mathcal{RB}(\mathcal{P}', t)$ of $t \in \mathcal{P}'$ to $\mathcal{P}$

Computing runtime bound for $t \in \mathcal{P}'$

$$\mathcal{RB}(t) \ =$$



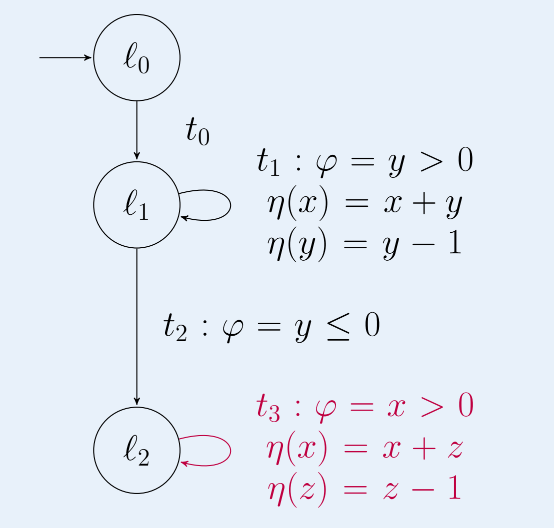
WST22
Jürgen Giesl, **Nils Lommen**, Marcel Hark, and Eleanore Meyer
RWTH Aachen University – LuFGi2

Modular Runtime Bounds from MΦRFs

Lift Runtime Bound $\mathcal{RB}(\mathcal{P}', t)$ of $t \in \mathcal{P}'$ to $\mathcal{P}$

Computing runtime bound for $t \in \mathcal{P}'$

$$\mathcal{RB}(t) \;=$$



- Lift runtime bounds of subprogram $\mathcal{P}' = \{t_3\}$ to bounds for $\mathcal{P}$

$$\mathcal{RB}(t_3) = \qquad 1 + 8 \cdot (z + 1 + x)$$

WST22
Jürgen Giesl, **Nils Lommen**, Marcel Hark, and Eleanore Meyer
RWTH Aachen University – LuFGi2

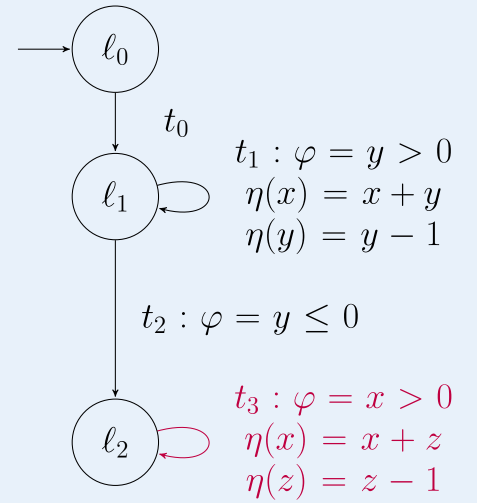
Modular Runtime Bounds from MΦRFs

Lift Runtime Bound $\mathcal{RB}(\mathcal{P}', t)$ of $t \in \mathcal{P}'$ to $\mathcal{P}$

Computing runtime bound for $t \in \mathcal{P}'$

$$\mathcal{RB}(t) \;=\; \mathcal{RB}(\mathcal{P}', t)$$

- Runtime bound $\mathcal{RB}(\mathcal{P}', t)$ of t in $\mathcal{P}'$



- Lift runtime bounds of subprogram $\mathcal{P}' = \{t_3\}$ to bounds for $\mathcal{P}$

$$\mathcal{RB}(t_3) = \quad 1 + 8 \cdot (z + 1 + x)$$

WST22
Jürgen Giesl, **Nils Lommen**, Marcel Hark, and Eleanore Meyer
RWTH Aachen University – LuFGi2

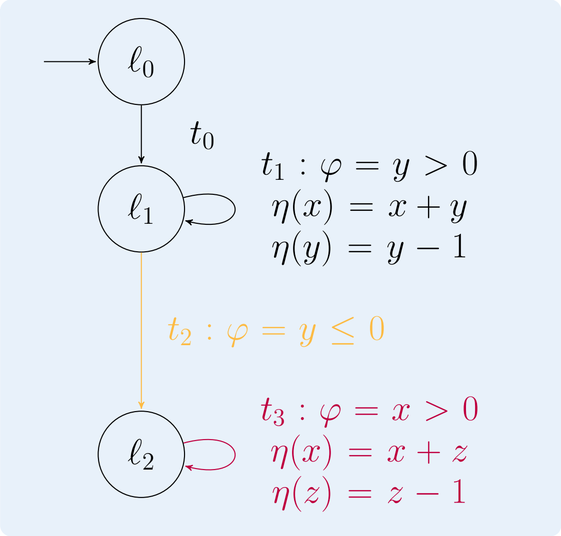
Modular Runtime Bounds from MΦRFs

Lift Runtime Bound $\mathcal{RB}(\mathcal{P}', t)$ of $t \in \mathcal{P}'$ to $\mathcal{P}$

Computing runtime bound for $t \in \mathcal{P}'$

$$\mathcal{RB}(t) \; = \qquad \mathcal{RB}(\mathcal{P}', t)$$

- Runtime bound $\mathcal{RB}(\mathcal{P}', t)$ of t in $\mathcal{P}'$



- Lift runtime bounds of subprogram $\mathcal{P}' = \{t_3\}$ to bounds for $\mathcal{P}$
  - How often is $\mathcal{P}'$ reached (by t_2)?

$$\mathcal{RB}(t_3) = \qquad 1 + 8 \cdot (z + 1 + x)$$

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

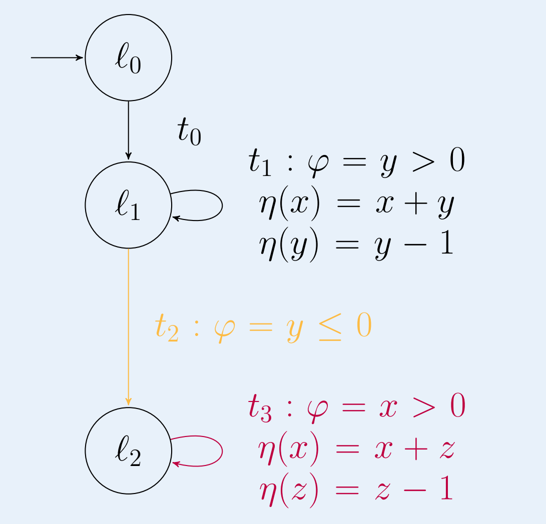
Modular Runtime Bounds from MΦRFs

Lift Runtime Bound $\mathcal{RB}(\mathcal{P}', t)$ of $t \in \mathcal{P}'$ to $\mathcal{P}$

Computing runtime bound for $t \in \mathcal{P}'$

$$\mathcal{RB}(t) \;=\; \mathcal{RB}(\mathcal{P}', t)$$

- Runtime bound $\mathcal{RB}(\mathcal{P}', t)$ of t in $\mathcal{P}'$



- Lift runtime bounds of subprogram $\mathcal{P}' = \{t_3\}$ to bounds for $\mathcal{P}$
  - How often is $\mathcal{P}'$ reached (by t_2)?
    - $\mathcal{RB}(t_2) = 1$

$$\mathcal{RB}(t_3) = \quad 1 + 8 \cdot (z + 1 + x)$$

WST22
Jürgen Giesl, **Nils Lommen**, Marcel Hark, and Eleanore Meyer
RWTH Aachen University – LuFGi2

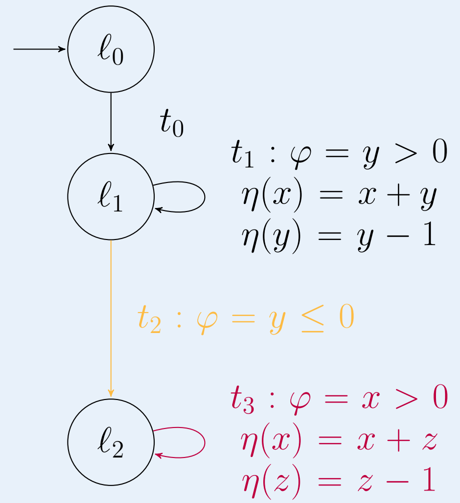
Modular Runtime Bounds from MΦRFs

Lift Runtime Bound $\mathcal{RB}(\mathcal{P}', t)$ of $t \in \mathcal{P}'$ to $\mathcal{P}$

Computing runtime bound for $t \in \mathcal{P}'$

$$\mathcal{RB}(t) \;=\; \mathcal{RB}(\mathcal{P}', t)$$

- Runtime bound $\mathcal{RB}(\mathcal{P}', t)$ of t in $\mathcal{P}'$



- Lift runtime bounds of subprogram $\mathcal{P}' = \{t_3\}$ to bounds for $\mathcal{P}$
  - How often is $\mathcal{P}'$ reached (by t_2)?
    - $\mathcal{RB}(t_2) = 1$

$$\mathcal{RB}(t_3) = 1 \cdot (1 + 8 \cdot (z + 1 + x))$$

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

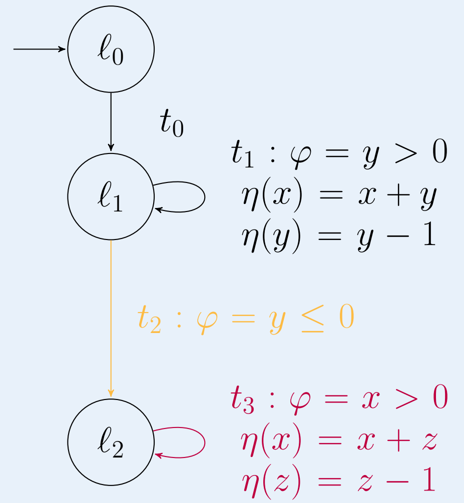
Modular Runtime Bounds from MΦRFs

Lift Runtime Bound $\mathcal{RB}(\mathcal{P}', t)$ of $t \in \mathcal{P}'$ to $\mathcal{P}$

Computing runtime bound for $t \in \mathcal{P}'$

$$\mathcal{RB}(t) = \sum_{t'} \mathcal{RB}(t') \cdot \mathcal{RB}(\mathcal{P}', t)$$

- Runtime bound $\mathcal{RB}(\mathcal{P}', t)$ of t in $\mathcal{P}'$
- t': pre-transition of $\mathcal{P}'$



- Lift runtime bounds of subprogram $\mathcal{P}' = \{t_3\}$ to bounds for $\mathcal{P}$
  - How often is $\mathcal{P}'$ reached (by t_2)?
    - $\mathcal{RB}(t_2) = 1$

$$\mathcal{RB}(t_3) = 1 \cdot (1 + 8 \cdot (z + 1 + x))$$

WST22
Jürgen Giesl, **Nils Lommen**, Marcel Hark, and Eleanore Meyer
RWTH Aachen University – LuFGi2

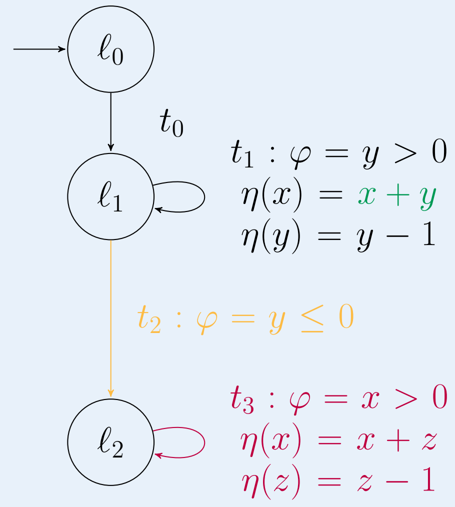
Modular Runtime Bounds from MΦRFs

Lift Runtime Bound $\mathcal{RB}(\mathcal{P}', t)$ of $t \in \mathcal{P}'$ to $\mathcal{P}$

Computing runtime bound for $t \in \mathcal{P}'$

$$\mathcal{RB}(t) \;=\; \sum_{t'} \mathcal{RB}(t') \cdot \mathcal{RB}(\mathcal{P}', t)$$

- Runtime bound $\mathcal{RB}(\mathcal{P}', t)$ of t in $\mathcal{P}'$
- t': pre-transition of $\mathcal{P}'$



- Lift runtime bounds of subprogram $\mathcal{P}' = \{t_3\}$ to bounds for $\mathcal{P}$
  - How often is $\mathcal{P}'$ reached (by t_2)?
    - $\mathcal{RB}(t_2) = 1$
  - consider initial value of variables `x` and `z` in full run before $\mathcal{P}'$

$$\mathcal{RB}(t_3) = 1 \cdot (1 + 8 \cdot (z + 1 + x))$$

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

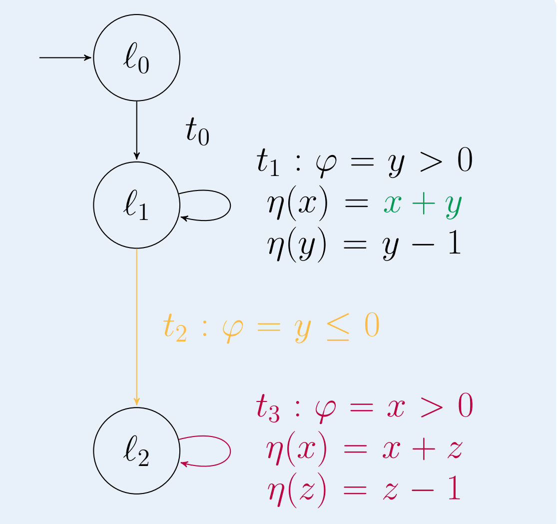
Modular Runtime Bounds from MΦRFs

Lift Runtime Bound $\mathcal{RB}(\mathcal{P}', t)$ of $t \in \mathcal{P}'$ to $\mathcal{P}$

Computing runtime bound for $t \in \mathcal{P}'$

$$\mathcal{RB}(t) \;=\; \sum_{t'} \mathcal{RB}(t') \cdot \mathcal{RB}(\mathcal{P}', t)$$

- Runtime bound $\mathcal{RB}(\mathcal{P}', t)$ of t in $\mathcal{P}'$
- t': pre-transition of $\mathcal{P}'$



- Lift runtime bounds of subprogram $\mathcal{P}' = \{t_3\}$ to bounds for $\mathcal{P}$
  - How often is $\mathcal{P}'$ reached (by t_2)?
    - $\mathcal{RB}(t_2) = 1$
  - consider initial value of variables `x` and `z` in full run before $\mathcal{P}'$
    - $\mathcal{SB}(t_2, x) = x + y^2$ and $\mathcal{SB}(t_2, z) = z$

$$\mathcal{RB}(t_3) = 1 \cdot (1 + 8 \cdot (z + 1 + x))$$

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

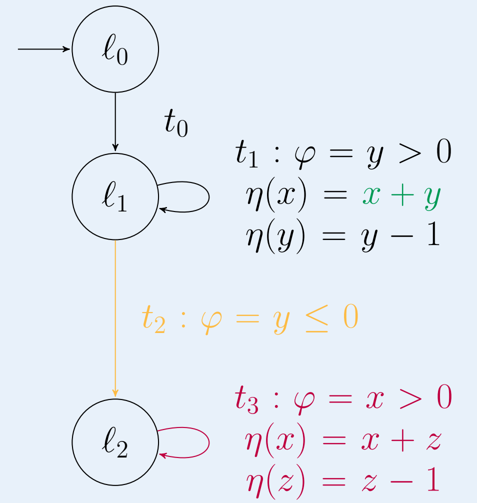
Modular Runtime Bounds from MΦRFs

Lift Runtime Bound $\mathcal{RB}(\mathcal{P}', t)$ of $t \in \mathcal{P}'$ to $\mathcal{P}$

Computing runtime bound for $t \in \mathcal{P}'$

$$\mathcal{RB}(t) \;=\; \sum_{t'} \mathcal{RB}(t') \cdot \mathcal{RB}(\mathcal{P}', t)$$

- Runtime bound $\mathcal{RB}(\mathcal{P}', t)$ of t in $\mathcal{P}'$
- t': pre-transition of $\mathcal{P}'$



- Lift runtime bounds of subprogram $\mathcal{P}' = \{t_3\}$ to bounds for $\mathcal{P}$
  - How often is $\mathcal{P}'$ reached (by t_2)?
    - $\mathcal{RB}(t_2) = 1$
  - consider initial value of variables $\mathtt{x}$ and $\mathtt{z}$ in full run before $\mathcal{P}'$
    - $\mathcal{SB}(t_2, x) = x + y^2$ and $\mathcal{SB}(t_2, z) = z$

$$\mathcal{RB}(t_3) = 1 \cdot (1 + 8 \cdot (z + 1 + x)[v \,/\, \mathcal{SB}(t_2, v)]) = 8 \cdot (z + 1 + x + y^2)$$

WST22
Jürgen Giesl, **Nils Lommen**, Marcel Hark, and Eleanore Meyer
RWTH Aachen University – LuFGi2

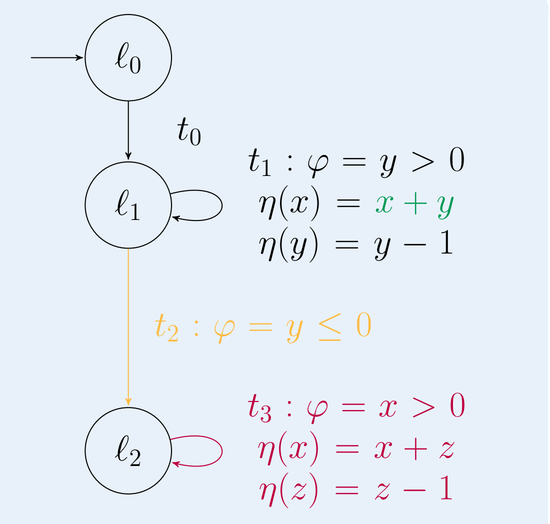
Modular Runtime Bounds from MΦRFs

Lift Runtime Bound $\mathcal{RB}(\mathcal{P}', t)$ of $t \in \mathcal{P}'$ to $\mathcal{P}$

Computing runtime bound for $t \in \mathcal{P}'$

$$\mathcal{RB}(t) \;=\; \sum_{t'} \mathcal{RB}(t') \cdot \mathcal{RB}(\mathcal{P}', t)\left[v \,/\, \mathcal{SB}(t', v)\right]$$

- Runtime bound $\mathcal{RB}(\mathcal{P}', t)$ of t in $\mathcal{P}'$
- t': pre-transition of $\mathcal{P}'$

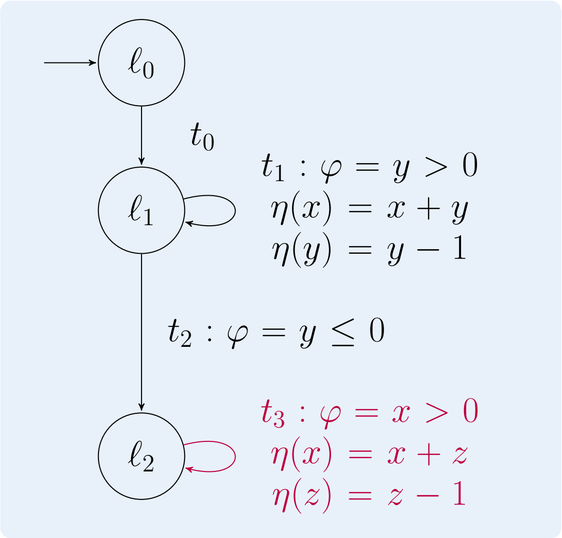


- Lift runtime bounds of subprogram $\mathcal{P}' = \{t_3\}$ to bounds for $\mathcal{P}$
  - How often is $\mathcal{P}'$ reached (by t_2)?
    - $\mathcal{RB}(t_2) = 1$
  - consider initial value of variables x and z in full run before $\mathcal{P}'$
    - $\mathcal{SB}(t_2, x) = x + y^2$ and $\mathcal{SB}(t_2, z) = z$

$$\mathcal{RB}(t_3) = 1 \cdot (1 + 8 \cdot (z + 1 + x)[v \,/\, \mathcal{SB}(t_2, v)]) = 8 \cdot (z + 1 + x + y^2)$$

WST22
Jürgen Giesl, **Nils Lommen**, Marcel Hark, and Eleanore Meyer
RWTH Aachen University – LuFGi2

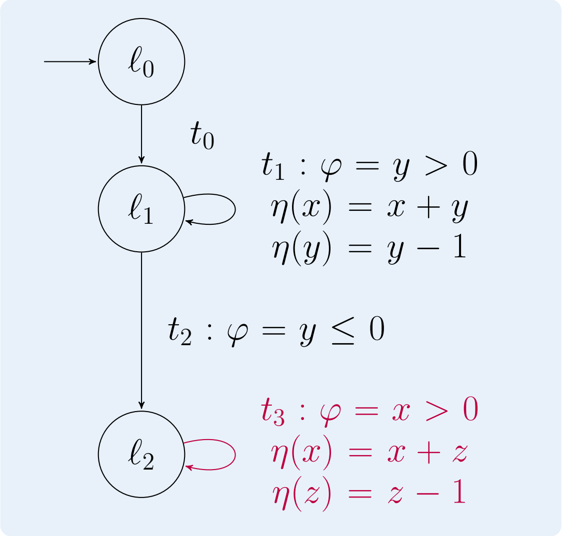
Runtime Complexity of Integer Programs



- Runtime bounds:
  - $\mathcal{RB}(t_0) = 1$
  - $\mathcal{RB}(t_1) = y$
  - $\mathcal{RB}(t_2) = 1$
  - $\mathcal{RB}(t_3) = ?$
- $\mathcal{RB}(\mathcal{P}) = 1 + y + 1 + ?$

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

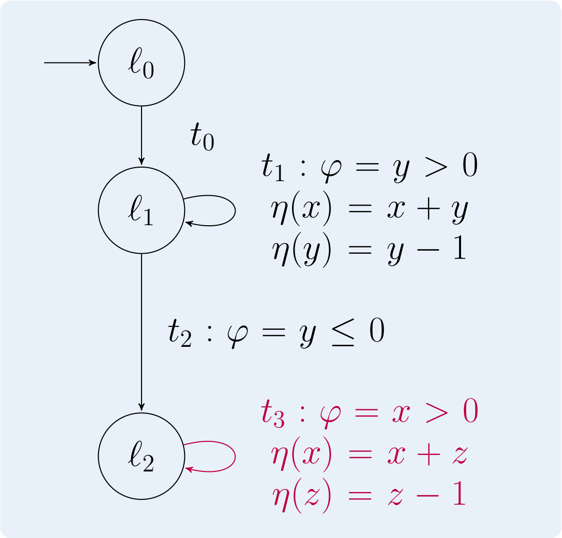
Runtime Complexity of Integer Programs



- Runtime bounds:
  - $\mathcal{RB}(t_0) = 1$
  - $\mathcal{RB}(t_1) = y$
  - $\mathcal{RB}(t_2) = 1$
  - $\mathcal{RB}(t_3) = 8 \cdot (z + 1 + x + y^2)$
- $\mathcal{RB}(\mathcal{P}) = 1 + y + 1 + 8 \cdot (z + 1 + x + y^2)$

WST22
Jürgen Giesl, **Nils Lommen**, Marcel Hark, and Eleanore Meyer
RWTH Aachen University – LuFGi2

Runtime Complexity of Integer Programs

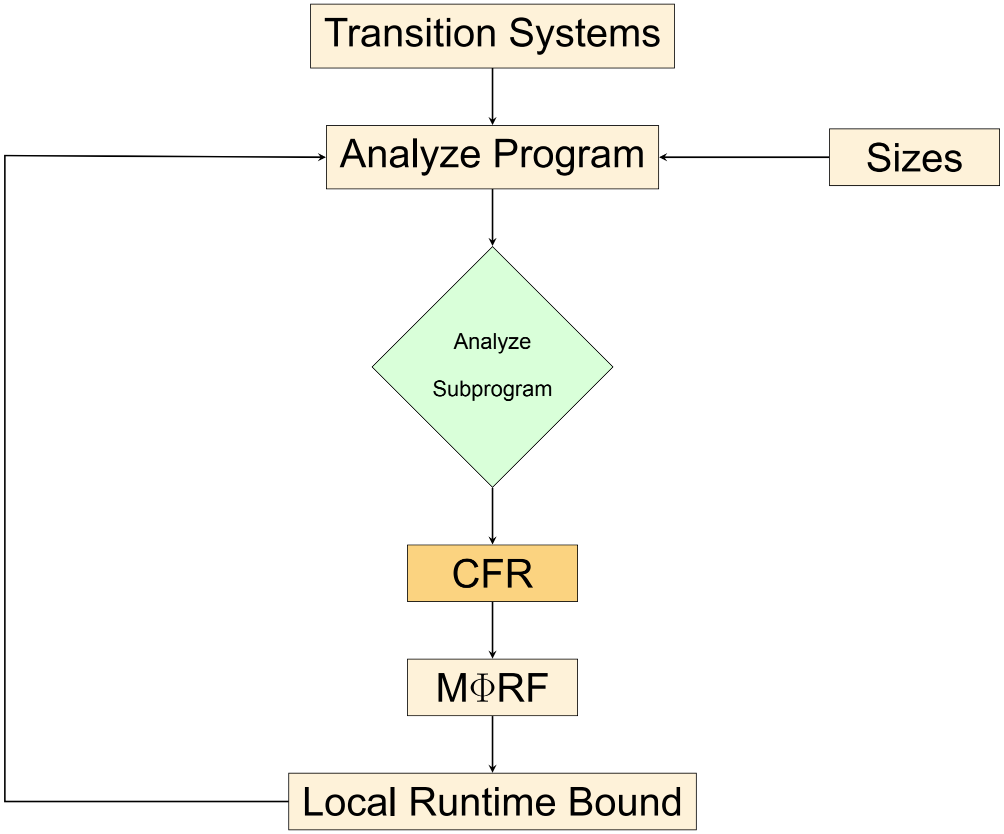


- Runtime bounds:
  - $\mathcal{RB}(t_0) = 1$
  - $\mathcal{RB}(t_1) = y$
  - $\mathcal{RB}(t_2) = 1$
  - $\mathcal{RB}(t_3) = 8 \cdot (z + 1 + x + y^2)$
- $\mathcal{RB}(\mathcal{P}) = 1 + y + 1 + 8 \cdot (z + 1 + x + y^2) \in \mathcal{O}(n^2)$

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

Overview

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



▶ Incorporate *local* control-flow refinement [Doménech et al. ’19]

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

Improvement by Modular Control-Flow Refinement

- **Problem**: complex, nested loops

```
while (x > 0) do
  if (y > 0) then
    y ← y − x
  else
    x ← x − 1
end
```


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

Improvement by Modular Control-Flow Refinement

- **Problem**: complex, nested loops
- Loop consists of *two* phases:

```
while (x > 0) do
  if (y > 0) then
    y ← y − x
  else
    x ← x − 1
end
```


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

Improvement by Modular Control-Flow Refinement

- **Problem**: complex, nested loops
- Loop consists of *two* phases:
  1. **then**-case is repeated until $y \leq 0$

```
while (x > 0) do
  if (y > 0) then
    y ← y − x
  else
    x ← x − 1
end
```


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

Improvement by Modular Control-Flow Refinement

- **Problem**: complex, nested loops
- Loop consists of *two* phases:
  1. **then**-case is repeated until $\mathrm{y} \leq 0$
  2. **else**-case is repeated until $\mathrm{x} \leq 0$

```
while (x > 0) do
  if (y > 0) then
    y ← y − x
  else
    x ← x − 1
end
```


WST22
Jürgen Giesl, **Nils Lommen**, Marcel Hark, and Eleanore Meyer
RWTH Aachen University – LuFGi2

Improvement by Modular Control-Flow Refinement

- **Problem**: complex, nested loops
- Loop consists of *two* phases:
  1. **then**-case is repeated until $y \leq 0$
  2. **else**-case is repeated until $x \leq 0$

  $\Rightarrow$ No run, where second phase is executed before first phase

```
while (x > 0) do
  if (y > 0) then
    y ← y − x
  else
    x ← x − 1
end
```


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

Improvement by Modular Control-Flow Refinement

- **Problem**: complex, nested loops
- Loop consists of *two* phases:
  1. **then**-case is repeated until $y \leq 0$
  2. **else**-case is repeated until $x \leq 0$

  $\Rightarrow$ No run, where second phase is executed before first phase

```
while (x > 0) do
  if (y > 0) then
    y ← y − x
  else
    x ← x − 1
end
```

Control-Flow Refinement by Partial Evaluation (CFR) [Doménech et al. ’19]

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

Improvement by Modular Control-Flow Refinement

- **Problem**: complex, nested loops
- Loop consists of *two* phases:
  1. **then**-case is repeated until $y \leq 0$
  2. **else**-case is repeated until $x \leq 0$

  $\Rightarrow$ No run, where second phase is executed before first phase

```
while (x > 0) do
  if (y > 0) then
    y ← y − x
  else
    x ← x − 1
end
```

Control-Flow Refinement by Partial Evaluation (CFR) [Doménech et al. ’19]

- sort out certain program paths

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

Improvement by Modular Control-Flow Refinement

- **Problem**: complex, nested loops
- Loop consists of *two* phases:
  1. **then**-case is repeated until $y \leq 0$
  2. **else**-case is repeated until $x \leq 0$

  $\Rightarrow$ No run, where second phase is executed before first phase

```
while (x > 0) do
  if (y > 0) then
    y ← y − x
  else
    x ← x − 1
end
```

⇝

```
while (x > 0 ∧ y > 0) do
  y ← y − x
end
while (x > 0 ∧ y ≤ 0) do
  x ← x − 1
end
```

Control-Flow Refinement by Partial Evaluation (CFR) [Doménech et al. ’19]

- sort out certain program paths

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

Improvement by Modular Control-Flow Refinement

- **Problem**: complex, nested loops
- Loop consists of *two* phases:
  1. **then**-case is repeated until $y \leq 0$
  2. **else**-case is repeated until $x \leq 0$

  $\Rightarrow$ No run, where second phase is executed before first phase

```
while (x > 0) do
  if (y > 0) then
    y ← y − x
  else
    x ← x − 1
end
```

⇝

```
while (x > 0 ∧ y > 0) do
  y ← y − x
end
while (x > 0 ∧ y ≤ 0) do
  x ← x − 1
end
```

Control-Flow Refinement by Partial Evaluation (CFR) [Doménech et al. '19]

- sort out certain program paths

$\Rightarrow$ integrate CFR into our modular approach

WST22
Jürgen Giesl, **Nils Lommen**, Marcel Hark, and Eleanore Meyer
RWTH Aachen University – LuFGi2

Improvement by Modular Control-Flow Refinement

- **Problem**: complex, nested loops
- Loop consists of *two* phases:
  1. **then**-case is repeated until $y \leq 0$
  2. **else**-case is repeated until $x \leq 0$

  $\Rightarrow$ No run, where second phase is executed before first phase

```
while (x > 0) do
  if (y > 0) then
    y ← y − x
  else
    x ← x − 1
end
```

⇝

```
while (x > 0 ∧ y > 0) do
  y ← y − x
end
while (x > 0 ∧ y ≤ 0) do
  x ← x − 1
end
```

Control-Flow Refinement by Partial Evaluation (CFR) [Doménech et al. ’19]

- sort out certain program paths

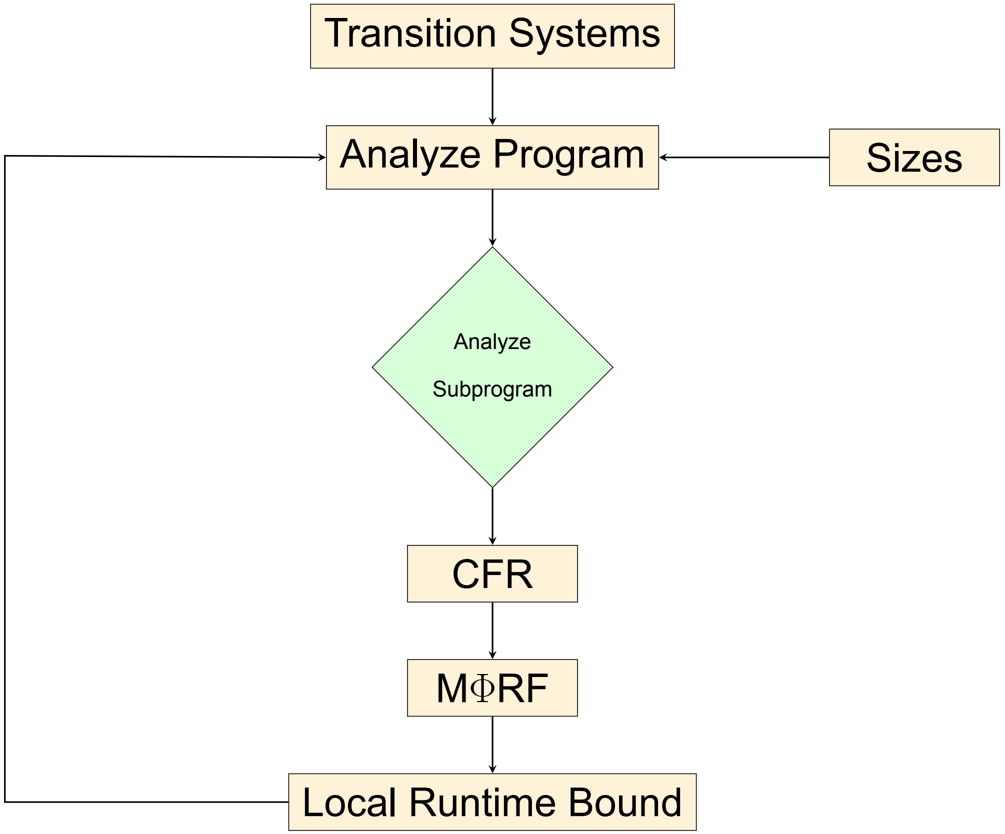
$\Rightarrow$ integrate CFR into our modular approach

- CFR *modular* for SCCs with “problematic” transitions

WST22
Jürgen Giesl, **Nils Lommen**, Marcel Hark, and Eleanore Meyer
RWTH Aachen University – LuFGi2

Overview

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



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

Evaluation of our Implementation in KoAT2

- C_Complexity consisting of 484 benchmarks from TPDB

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

Evaluation of our Implementation in `KoAT2`

- `C_Complexity` consisting of 484 benchmarks from `TPDB`
- Timeout of 300 seconds

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

Evaluation of our Implementation in KoAT2

- C_Complexity consisting of 484 benchmarks from TPDB
- Timeout of 300 seconds

	$\mathcal{O}(1)$	$\mathcal{O}(n)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n^{>2})$	$< \infty$	AVG(s)
Loopus	17	169	49	4	239	0.84
KoAT1	25	168	74	12	285	2.36
CoFloCo	22	195	66	5	288	0.81

Evaluation of our Implementation in KoAT2

- C_Complexity consisting of 484 benchmarks from TPDB
- Timeout of 300 seconds

	$\mathcal{O}(1)$	$\mathcal{O}(n)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n^{>2})$	$< \infty$	AVG(s)
Loopus	17	169	49	4	239	0.84
KoAT1	25	168	74	12	285	2.36
CoFloCo	22	195	66	5	288	0.81
KoAT2 + MΦRF	23	204	71	12	310	2.11
MaxCore	23	214	66	7	310	1.94

WST22
Jürgen Giesl, **Nils Lommen**, Marcel Hark, and Eleanore Meyer
RWTH Aachen University – LuFGi2

Evaluation of our Implementation in KoAT2

- C_Complexity consisting of 484 benchmarks from TPDB
- Timeout of 300 seconds

	$\mathcal{O}(1)$	$\mathcal{O}(n)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n^{>2})$	$< \infty$	AVG(s)
Loopus	17	169	49	4	239	0.84
KoAT1	25	168	74	12	285	2.36
CoFloCo	22	195	66	5	288	0.81
KoAT2 + MΦRF	23	204	71	12	310	2.11
MaxCore	23	214	66	7	310	1.94
KoAT2 + CFR	25	216	68	11	320	5.14

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

Evaluation of our Implementation in KoAT2

- C_Complexity consisting of 484 benchmarks from TPDB
- Timeout of 300 seconds

	$\mathcal{O}(1)$	$\mathcal{O}(n)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n^{>2})$	$< \infty$	AVG(s)
Loopus	17	169	49	4	239	0.84
KoAT1	25	168	74	12	285	2.36
CoFloCo	22	195	66	5	288	0.81
KoAT2 + MΦRF	23	204	71	12	310	2.11
MaxCore	23	214	66	7	310	1.94
KoAT2 + CFR	25	216	68	11	320	5.14
KoAT2 + CFR + MΦRF	24	228	65	11	328	4.77

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

Evaluation of our Implementation in KoAT2

- C_Complexity consisting of 484 benchmarks from TPDB
- Timeout of 300 seconds

	$\mathcal{O}(1)$	$\mathcal{O}(n)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n^{>2})$	$< \infty$	AVG(s)
Loopus	17	169	49	4	239	0.84
KoAT1	25	168	74	12	285	2.36
CoFloCo	22	195	66	5	288	0.81
KoAT2 + MΦRF	23	204	71	12	310	2.11
MaxCore	23	214	66	7	310	1.94
KoAT2 + CFR	25	216	68	11	320	5.14
KoAT2 + CFR + MΦRF	24	228	65	11	328	4.77

- At most 366 benchmarks might terminate

WST22
Jürgen Giesl, **Nils Lommen**, Marcel Hark, and Eleanore Meyer
RWTH Aachen University – LuFGi2

Evaluation of our Implementation in KoAT2

- C_Complexity consisting of 484 benchmarks from TPDB
- Timeout of 300 seconds

	$\mathcal{O}(1)$	$\mathcal{O}(n)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n^{>2})$	$< \infty$	AVG(s)	succ. rate
Loopus	17	169	49	4	239	0.84	65%
KoAT1	25	168	74	12	285	2.36	77%
CoFloCo	22	195	66	5	288	0.81	79%
KoAT2 + MΦRF	23	204	71	12	310	2.11	85%
MaxCore	23	214	66	7	310	1.94	85%
KoAT2 + CFR	25	216	68	11	320	5.14	87%
KoAT2 + CFR + MΦRF	24	228	65	11	328	4.77	90%

- At most 366 benchmarks might terminate
- KoAT2 + CFR + MΦRF solves 90% of benchmarks which might terminate

WST22
Jürgen Giesl, **Nils Lommen**, Marcel Hark, and Eleanore Meyer
RWTH Aachen University – LuFGi2

Conclusion & Additional Contributions

- Conclusion

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

Conclusion & Additional Contributions

- Conclusion
  - Automatic complexity analysis of integer programs

WST22
Jürgen Giesl, **Nils Lommen**, Marcel Hark, and Eleanore Meyer
RWTH Aachen University – LuFGi2

Conclusion & Additional Contributions

- Conclusion
  - Automatic complexity analysis of integer programs
  - Integrate *modular* MΦRF based approach

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

Conclusion & Additional Contributions

- Conclusion
  - Automatic complexity analysis of integer programs
  - Integrate *modular* MΦRF based approach
  - Integrate CFR via `iRankfinder`

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

Conclusion & Additional Contributions

- Conclusion
  - Automatic complexity analysis of integer programs
  - Integrate *modular* MΦRF based approach
  - Integrate CFR via `iRankfinder`
- Additional Contributions

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

Conclusion & Additional Contributions

- Conclusion
  - Automatic complexity analysis of integer programs
  - Integrate *modular* MΦRF based approach
  - Integrate CFR via `iRankfinder`
- Additional Contributions
  - Improvement by non-linear bounds for Triangular Weakly Non-Linear Loops

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

Conclusion & Additional Contributions

- Conclusion
  - Automatic complexity analysis of integer programs
  - Integrate *modular* MΦRF based approach
  - Integrate CFR via `iRankfinder`
- Additional Contributions
  - Improvement by non-linear bounds for Triangular Weakly Non-Linear Loops
  - Ranking function based modular approach for *probabilistic* programs

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

Conclusion & Additional Contributions

- Conclusion
  - Automatic complexity analysis of integer programs
  - Integrate *modular* MΦRF based approach
  - Integrate CFR via `iRankfinder`
- Additional Contributions
  - Improvement by non-linear bounds for Triangular Weakly Non-Linear Loops
  - Ranking function based modular approach for *probabilistic* programs

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

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

Conclusion & Additional Contributions

- Conclusion
  - Automatic complexity analysis of integer programs
  - Integrate *modular* MΦRF based approach
  - Integrate CFR via `iRankfinder`
- Additional Contributions
  - Improvement by non-linear bounds for Triangular Weakly Non-Linear Loops
  - Ranking function based modular approach for *probabilistic* programs

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

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

Conclusion & Additional Contributions

- **Conclusion**
  - Automatic complexity analysis of integer programs
  - Integrate *modular* MΦRF based approach
  - Integrate CFR via `iRankfinder`
- **Additional Contributions**
  - Improvement by non-linear bounds for Triangular Weakly Non-Linear Loops
  - Ranking function based modular approach for *probabilistic* programs

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

Thank You!

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