Complexity Analysis for Term Rewriting by Integer Transition Systems

Florian Frohn[1]
Matthias Naaf[1], Marc Brockschmidt[2], Carsten Fuhs[3], and Jürgen Giesl[1]

[1]RWTH Aachen University, Germany
[2]Microsoft Research, Cambridge, UK
[3]Birkbeck, University of London, UK

September 8, 2017

Analyzing Insertion Sort by Hand

Example

$$
\begin{array}{rcl}
\mathbf{isort}(\mathsf{nil}, ys) & \to & ys \\
\mathbf{isort}(\mathsf{cons}(x, xs), ys) & \to & \mathbf{isort}(xs, \mathbf{insert}(x, ys)) \\
\mathbf{insert}(x, \mathsf{nil}) & \to & \mathsf{cons}(x, \mathsf{nil}) \\
\mathbf{insert}(x, \mathsf{cons}(y, ys)) & \to & \mathbf{if}(\mathbf{gt}(x, y), x, \mathsf{cons}(y, ys)) \\
\mathbf{if}(\mathsf{true}, x, \mathsf{cons}(y, ys)) & \to & \mathsf{cons}(y, \mathbf{insert}(x, ys)) \\
\mathbf{if}(\mathsf{false}, x, \mathsf{cons}(y, ys)) & \to & \mathsf{cons}(x, \mathsf{cons}(y, ys)) \\
\mathbf{gt}(0, y) & \to & \mathsf{false} \\
\mathbf{gt}(\mathsf{s}(x), 0) & \to & \mathsf{true} \\
\mathbf{gt}(\mathsf{s}(x), \mathsf{s}(y)) & \to & \mathbf{gt}(x, y)
\end{array}
$$

Analyzing Insertion Sort by Hand

Example

$$
\begin{aligned}
\mathbf{isort}(\mathsf{nil}, ys) &\longrightarrow ys \\
\mathbf{isort}(\mathsf{cons}(x, xs), ys) &\longrightarrow \mathbf{isort}(xs, \mathbf{insert}(x, ys)) \\
\mathbf{insert}(x, \mathsf{nil}) &\longrightarrow \mathsf{cons}(x, \mathsf{nil}) \\
\mathbf{insert}(x, \mathsf{cons}(y, ys)) &\longrightarrow \mathbf{if}(\mathbf{gt}(x, y), x, \mathsf{cons}(y, ys)) \\
\mathbf{if}(\mathsf{true}, x, \mathsf{cons}(y, ys)) &\longrightarrow \mathsf{cons}(y, \mathbf{insert}(x, ys)) \\
\mathbf{if}(\mathsf{false}, x, \mathsf{cons}(y, ys)) &\longrightarrow \mathsf{cons}(x, \mathsf{cons}(y, ys)) \\
\mathbf{gt}(0, y) &\longrightarrow \mathsf{false} \\
\mathbf{gt}(\mathsf{s}(x), 0) &\longrightarrow \mathsf{true} \\
\mathbf{gt}(\mathsf{s}(x), \mathsf{s}(y)) &\longrightarrow \mathbf{gt}(x, y)
\end{aligned}
$$

Note: innermost reduction strategy

Analyzing Insertion Sort by Hand

Example

$$
\begin{array}{rcl}
\mathbf{isort}(\mathsf{nil}, ys) & \to & ys \\
\mathbf{isort}(\mathsf{cons}(x, xs), ys) & \to & \mathbf{isort}(xs, \mathbf{insert}(x, ys)) \\
\mathbf{insert}(x, \mathsf{nil}) & \to & \mathsf{cons}(x, \mathsf{nil}) \\
\mathbf{insert}(x, \mathsf{cons}(y, ys)) & \to & \mathbf{if}(\mathbf{gt}(x, y), x, \mathsf{cons}(y, ys)) \\
\mathbf{if}(\mathsf{true}, x, \mathsf{cons}(y, ys)) & \to & \mathsf{cons}(y, \mathbf{insert}(x, ys)) \\
\mathbf{if}(\mathsf{false}, x, \mathsf{cons}(y, ys)) & \to & \mathsf{cons}(x, \mathsf{cons}(y, ys)) \\
\mathbf{gt}(0, y) & \to & \mathsf{false} \\
\mathbf{gt}(\mathsf{s}(x), 0) & \to & \mathsf{true} \\
\mathbf{gt}(\mathsf{s}(x), \mathsf{s}(y)) & \to & \mathbf{gt}(x, y)
\end{array}
$$

- $\mathbf{gt}(x, y)$: $\mathcal{O}(x)$

Note: innermost reduction strategy

Analyzing Insertion Sort by Hand

Example

$$
\begin{array}{rcl}
\mathbf{isort}(\mathsf{nil}, ys) & \to & ys \\
\mathbf{isort}(\mathsf{cons}(x, xs), ys) & \to & \mathbf{isort}(xs, \mathbf{insert}(x, ys)) \\
\mathbf{insert}(x, \mathsf{nil}) & \to & \mathsf{cons}(x, \mathsf{nil}) \\
\mathbf{insert}(x, \mathsf{cons}(y, ys)) & \to & \mathbf{if}(\mathbf{gt}(x, y), x, \mathsf{cons}(y, ys)) \\
\mathbf{if}(\mathsf{true}, x, \mathsf{cons}(y, ys)) & \to & \mathsf{cons}(y, \mathbf{insert}(x, ys)) \\
\mathbf{if}(\mathsf{false}, x, \mathsf{cons}(y, ys)) & \to & \mathsf{cons}(x, \mathsf{cons}(y, ys)) \\
\mathbf{gt}(0, y) & \to & \mathsf{false} \\
\mathbf{gt}(\mathsf{s}(x), 0) & \to & \mathsf{true} \\
\mathbf{gt}(\mathsf{s}(x), \mathsf{s}(y)) & \to & \mathbf{gt}(x, y)
\end{array}
$$

- $\mathbf{gt}(x, y)$: $\mathcal{O}(x)$
- $\mathbf{insert}(x, ys)$: $\mathcal{O}(\mathsf{length}(ys) \cdot \ldots)$

Note: innermost reduction strategy

Analyzing Insertion Sort by Hand

Example

$$
\begin{array}{rcl}
\mathbf{isort}(\mathsf{nil}, ys) & \to & ys \\
\mathbf{isort}(\mathsf{cons}(x, xs), ys) & \to & \mathbf{isort}(xs, \mathbf{insert}(x, ys)) \\
\mathbf{insert}(x, \mathsf{nil}) & \to & \mathsf{cons}(x, \mathsf{nil}) \\
\mathbf{insert}(x, \mathsf{cons}(y, ys)) & \to & \mathbf{if}(\mathbf{gt}(x, y), x, \mathsf{cons}(y, ys)) \\
\mathbf{if}(\mathsf{true}, x, \mathsf{cons}(y, ys)) & \to & \mathsf{cons}(y, \mathbf{insert}(x, ys)) \\
\mathbf{if}(\mathsf{false}, x, \mathsf{cons}(y, ys)) & \to & \mathsf{cons}(x, \mathsf{cons}(y, ys)) \\
\mathbf{gt}(0, y) & \to & \mathsf{false} \\
\mathbf{gt}(\mathsf{s}(x), 0) & \to & \mathsf{true} \\
\mathbf{gt}(\mathsf{s}(x), \mathsf{s}(y)) & \to & \mathbf{gt}(x, y)
\end{array}
$$

- $\mathbf{gt}(x, y)$: $\mathcal{O}(x)$
- $\mathbf{insert}(x, ys)$: $\mathcal{O}(\mathsf{length}(ys) \cdot x)$

Note: innermost reduction strategy

Analyzing Insertion Sort by Hand

Example

$$
\begin{aligned}
\mathbf{isort}(\mathsf{nil}, ys) &\rightarrow ys \\
\mathbf{isort}(\mathsf{cons}(x, xs), ys) &\rightarrow \mathbf{isort}(xs, \mathbf{insert}(x, ys)) \\
\mathbf{insert}(x, \mathsf{nil}) &\rightarrow \mathsf{cons}(x, \mathsf{nil}) \\
\mathbf{insert}(x, \mathsf{cons}(y, ys)) &\rightarrow \mathbf{if}(\mathbf{gt}(x, y), x, \mathsf{cons}(y, ys)) \\
\mathbf{if}(\mathsf{true}, x, \mathsf{cons}(y, ys)) &\rightarrow \mathsf{cons}(y, \mathbf{insert}(x, ys)) \\
\mathbf{if}(\mathsf{false}, x, \mathsf{cons}(y, ys)) &\rightarrow \mathsf{cons}(x, \mathsf{cons}(y, ys)) \\
\mathbf{gt}(0, y) &\rightarrow \mathsf{false} \\
\mathbf{gt}(\mathsf{s}(x), 0) &\rightarrow \mathsf{true} \\
\mathbf{gt}(\mathsf{s}(x), \mathsf{s}(y)) &\rightarrow \mathbf{gt}(x, y)
\end{aligned}
$$

- $\mathbf{gt}(x, y)$: $\mathcal{O}(x)$
- $\mathbf{insert}(x, ys)$: $\mathcal{O}(\mathsf{length}(ys) \cdot x)$
- $\mathbf{isort}(xs, ys)$: $\mathcal{O}(\mathsf{length}(xs) \cdot \ldots)$

Note: innermost reduction strategy

Analyzing Insertion Sort by Hand

Example

$$
\begin{array}{rcl}
\mathbf{isort}(\mathsf{nil}, ys) & \to & ys \\
\mathbf{isort}(\mathsf{cons}(x, xs), ys) & \to & \mathbf{isort}(xs, \mathbf{insert}(x, ys)) \\
\mathbf{insert}(x, \mathsf{nil}) & \to & \mathsf{cons}(x, \mathsf{nil}) \\
\mathbf{insert}(x, \mathsf{cons}(y, ys)) & \to & \mathbf{if}(\mathbf{gt}(x, y), x, \mathsf{cons}(y, ys)) \\
\mathbf{if}(\mathsf{true}, x, \mathsf{cons}(y, ys)) & \to & \mathsf{cons}(y, \mathbf{insert}(x, ys)) \\
\mathbf{if}(\mathsf{false}, x, \mathsf{cons}(y, ys)) & \to & \mathsf{cons}(x, \mathsf{cons}(y, ys)) \\
\mathbf{gt}(0, y) & \to & \mathsf{false} \\
\mathbf{gt}(\mathsf{s}(x), 0) & \to & \mathsf{true} \\
\mathbf{gt}(\mathsf{s}(x), \mathsf{s}(y)) & \to & \mathbf{gt}(x, y)
\end{array}
$$

- $\mathbf{gt}(x, y)$: $\mathcal{O}(x)$
- $\mathbf{insert}(x, ys)$: $\mathcal{O}(\mathsf{length}(ys) \cdot x)$
- $\mathbf{isort}(xs, ys)$: $\mathcal{O}(\mathsf{length}(xs) \cdot (\mathsf{length}(xs) + \mathsf{length}(ys)) \cdot \mathsf{max}(xs))$

Note: innermost reduction strategy

Relative Rules

Example

$$
\begin{array}{rcl}
\mathbf{isort}(\mathsf{nil}, ys) & \rightarrow & ys \\
\mathbf{isort}(\mathsf{cons}(x, xs), ys) & \rightarrow & \mathbf{isort}(xs, \mathbf{insert}(x, ys)) \\
\mathbf{insert}(x, \mathsf{nil}) & \rightarrow & \mathsf{cons}(x, \mathsf{nil}) \\
\mathbf{insert}(x, \mathsf{cons}(y, ys)) & \rightarrow & \mathbf{if}(\mathbf{gt}(x, y), x, \mathsf{cons}(y, ys)) \\
\mathbf{if}(\mathsf{true}, x, \mathsf{cons}(y, ys)) & \rightarrow & \mathsf{cons}(y, \mathbf{insert}(x, ys)) \\
\mathbf{if}(\mathsf{false}, x, \mathsf{cons}(y, ys)) & \rightarrow & \mathsf{cons}(x, \mathsf{cons}(y, ys)) \\
\mathbf{gt}(0, y) & \xrightarrow{=} & \mathsf{false} \\
\mathbf{gt}(\mathsf{s}(x), 0) & \xrightarrow{=} & \mathsf{true} \\
\mathbf{gt}(\mathsf{s}(x), \mathsf{s}(y)) & \xrightarrow{=} & \mathbf{gt}(x, y)
\end{array}
$$

- $\mathbf{gt}(x, y)$: $\mathcal{O}(x)$
- $\mathbf{insert}(x, ys)$: $\mathcal{O}(\mathsf{length}(ys) \cdot x)$
- $\mathbf{isort}(xs, ys)$: $\mathcal{O}(\mathsf{length}(xs) \cdot (\mathsf{length}(xs) + \mathsf{length}(ys)) \cdot \max(xs))$

Note: innermost reduction strategy

Relative Rules

Example

$$
\begin{array}{rcl}
\mathbf{isort}(\mathsf{nil}, ys) & \to & ys \\
\mathbf{isort}(\mathsf{cons}(x, xs), ys) & \to & \mathbf{isort}(xs, \mathbf{insert}(x, ys)) \\
\mathbf{insert}(x, \mathsf{nil}) & \to & \mathsf{cons}(x, \mathsf{nil}) \\
\mathbf{insert}(x, \mathsf{cons}(y, ys)) & \to & \mathbf{if}(\mathbf{gt}(x, y), x, \mathsf{cons}(y, ys)) \\
\mathbf{if}(\mathsf{true}, x, \mathsf{cons}(y, ys)) & \to & \mathsf{cons}(y, \mathbf{insert}(x, ys)) \\
\mathbf{if}(\mathsf{false}, x, \mathsf{cons}(y, ys)) & \to & \mathsf{cons}(x, \mathsf{cons}(y, ys)) \\
\mathbf{gt}(0, y) & \xrightarrow{=} & \mathsf{false} \\
\mathbf{gt}(\mathsf{s}(x), 0) & \xrightarrow{=} & \mathsf{true} \\
\mathbf{gt}(\mathsf{s}(x), \mathsf{s}(y)) & \xrightarrow{=} & \mathbf{gt}(x, y)
\end{array}
$$

- $\mathbf{gt}(x, y)$: $\mathcal{O}(1)$
- $\mathbf{insert}(x, ys)$: $\mathcal{O}(\mathsf{length}(ys) \cdot x)$
- $\mathbf{isort}(xs, ys)$: $\mathcal{O}(\mathsf{length}(xs) \cdot (\mathsf{length}(xs) + \mathsf{length}(ys)) \cdot \mathsf{max}(xs))$

Note: innermost reduction strategy

Relative Rules

Example

$$
\begin{array}{rcl}
\mathbf{isort}(\mathsf{nil}, ys) & \rightarrow & ys \\
\mathbf{isort}(\mathsf{cons}(x, xs), ys) & \rightarrow & \mathbf{isort}(xs, \mathbf{insert}(x, ys)) \\
\mathbf{insert}(x, \mathsf{nil}) & \rightarrow & \mathsf{cons}(x, \mathsf{nil}) \\
\mathbf{insert}(x, \mathsf{cons}(y, ys)) & \rightarrow & \mathbf{if}(\mathbf{gt}(x, y), x, \mathsf{cons}(y, ys)) \\
\mathbf{if}(\mathsf{true}, x, \mathsf{cons}(y, ys)) & \rightarrow & \mathsf{cons}(y, \mathbf{insert}(x, ys)) \\
\mathbf{if}(\mathsf{false}, x, \mathsf{cons}(y, ys)) & \rightarrow & \mathsf{cons}(x, \mathsf{cons}(y, ys)) \\
\mathbf{gt}(0, y) & \xrightarrow{=} & \mathsf{false} \\
\mathbf{gt}(\mathsf{s}(x), 0) & \xrightarrow{=} & \mathsf{true} \\
\mathbf{gt}(\mathsf{s}(x), \mathsf{s}(y)) & \xrightarrow{=} & \mathbf{gt}(x, y)
\end{array}
$$

- $\mathbf{gt}(x, y)$: $\mathcal{O}(1)$
- $\mathbf{insert}(x, ys)$: $\mathcal{O}(\mathsf{length}(ys))$
- $\mathbf{isort}(xs, ys)$: $\mathcal{O}(\mathsf{length}(xs) \cdot (\mathsf{length}(xs) + \mathsf{length}(ys)) \cdot \mathsf{max}(xs))$

Note: innermost reduction strategy

Relative Rules

Example

$$
\begin{array}{rcl}
\mathbf{isort}(\mathsf{nil}, ys) & \to & ys \\
\mathbf{isort}(\mathsf{cons}(x, xs), ys) & \to & \mathbf{isort}(xs, \mathbf{insert}(x, ys)) \\
\mathbf{insert}(x, \mathsf{nil}) & \to & \mathsf{cons}(x, \mathsf{nil}) \\
\mathbf{insert}(x, \mathsf{cons}(y, ys)) & \to & \mathbf{if}(\mathbf{gt}(x, y), x, \mathsf{cons}(y, ys)) \\
\mathbf{if}(\mathsf{true}, x, \mathsf{cons}(y, ys)) & \to & \mathsf{cons}(y, \mathbf{insert}(x, ys)) \\
\mathbf{if}(\mathsf{false}, x, \mathsf{cons}(y, ys)) & \to & \mathsf{cons}(x, \mathsf{cons}(y, ys)) \\
\mathbf{gt}(0, y) & \xrightarrow{=} & \mathsf{false} \\
\mathbf{gt}(\mathsf{s}(x), 0) & \xrightarrow{=} & \mathsf{true} \\
\mathbf{gt}(\mathsf{s}(x), \mathsf{s}(y)) & \xrightarrow{=} & \mathbf{gt}(x, y)
\end{array}
$$

- $\mathbf{gt}(x, y)$: $\mathcal{O}(1)$
- $\mathbf{insert}(x, ys)$: $\mathcal{O}(\mathsf{length}(ys))$
- $\mathbf{isort}(xs, ys)$: $\mathcal{O}(\mathsf{length}(xs) \cdot (\mathsf{length}(xs) + \mathsf{length}(ys)))$

Note: innermost reduction strategy

AProVE

Example

$$
\begin{aligned}
\mathbf{isort}(\mathsf{nil}, ys) &\to ys \\
\mathbf{isort}(\mathsf{cons}(x, xs), ys) &\to \mathbf{isort}(xs, \mathbf{insert}(x, ys)) \\
\mathbf{insert}(x, \mathsf{nil}) &\to \mathsf{cons}(x, \mathsf{nil}) \\
\mathbf{insert}(x, \mathsf{cons}(y, ys)) &\to \mathbf{if}(\mathbf{gt}(x, y), x, \mathsf{cons}(y, ys)) \\
\mathbf{if}(\mathsf{true}, x, \mathsf{cons}(y, ys)) &\to \mathsf{cons}(y, \mathbf{insert}(x, ys)) \\
\mathbf{if}(\mathsf{false}, x, \mathsf{cons}(y, ys)) &\to \mathsf{cons}(x, \mathsf{cons}(y, ys)) \\
\mathbf{gt}(0, y) &\xrightarrow{=} \mathsf{false} \\
\mathbf{gt}(\mathsf{s}(x), 0) &\xrightarrow{=} \mathsf{true} \\
\mathbf{gt}(\mathsf{s}(x), \mathsf{s}(y)) &\xrightarrow{=} \mathbf{gt}(x, y)
\end{aligned}
$$

- the recursive **isort** rule is at most applied linearly often

AProVE

Example

$$
\begin{array}{rcl}
\mathbf{isort}(\mathsf{nil}, ys) & \to & ys \\
\mathbf{isort}(\mathsf{cons}(x, xs), ys) & \to & \mathbf{isort}(xs, \mathbf{insert}(x, ys)) \\
\mathbf{insert}(x, \mathsf{nil}) & \to & \mathsf{cons}(x, \mathsf{nil}) \\
\mathbf{insert}(x, \mathsf{cons}(y, ys)) & \to & \mathbf{if}(\mathbf{gt}(x, y), x, \mathsf{cons}(y, ys)) \\
\mathbf{if}(\mathsf{true}, x, \mathsf{cons}(y, ys)) & \to & \mathsf{cons}(y, \mathbf{insert}(x, ys)) \\
\mathbf{if}(\mathsf{false}, x, \mathsf{cons}(y, ys)) & \to & \mathsf{cons}(x, \mathsf{cons}(y, ys)) \\
\mathbf{gt}(0, y) & \xrightarrow{=} & \mathsf{false} \\
\mathbf{gt}(\mathsf{s}(x), 0) & \xrightarrow{=} & \mathsf{true} \\
\mathbf{gt}(\mathsf{s}(x), \mathsf{s}(y)) & \xrightarrow{=} & \mathbf{gt}(x, y)
\end{array}
$$

- the recursive **isort** rule is at most applied linearly often
- the recursive **insert** rule is at most applied quadratically often

AProVE

Example

$$
\begin{aligned}
\mathbf{isort}(\mathrm{nil}, ys) &\rightarrow ys \\
\mathbf{isort}(\mathrm{cons}(x, xs), ys) &\rightarrow \mathbf{isort}(xs, \mathbf{insert}(x, ys)) \\
\mathbf{insert}(x, \mathrm{nil}) &\rightarrow \mathrm{cons}(x, \mathrm{nil}) \\
\mathbf{insert}(x, \mathrm{cons}(y, ys)) &\rightarrow \mathbf{if}(\mathbf{gt}(x, y), x, \mathrm{cons}(y, ys)) \\
\mathbf{if}(\mathrm{true}, x, \mathrm{cons}(y, ys)) &\rightarrow \mathrm{cons}(y, \mathbf{insert}(x, ys)) \\
\mathbf{if}(\mathrm{false}, x, \mathrm{cons}(y, ys)) &\rightarrow \mathrm{cons}(x, \mathrm{cons}(y, ys)) \\
\mathbf{gt}(0, y) &\xrightarrow{=} \mathrm{false} \\
\mathbf{gt}(\mathrm{s}(x), 0) &\xrightarrow{=} \mathrm{true} \\
\mathbf{gt}(\mathrm{s}(x), \mathrm{s}(y)) &\xrightarrow{=} \mathbf{gt}(x, y)
\end{aligned}
$$

- the recursive **isort** rule is at most applied linearly often
- the recursive **insert** rule is at most applied quadratically often
  - Note: Requires reasoning about **isort**, **insert**, and **if** rules!

AProVE

Example

$$
\begin{array}{rcl}
\mathbf{isort}(\mathsf{nil}, ys) & \to & ys \\
\mathbf{isort}(\mathsf{cons}(x, xs), ys) & \to & \mathbf{isort}(xs, \mathbf{insert}(x, ys)) \\
\mathbf{insert}(x, \mathsf{nil}) & \to & \mathsf{cons}(x, \mathsf{nil}) \\
\mathbf{insert}(x, \mathsf{cons}(y, ys)) & \to & \mathbf{if}(\mathbf{gt}(x, y), x, \mathsf{cons}(y, ys)) \\
\mathbf{if}(\mathsf{true}, x, \mathsf{cons}(y, ys)) & \to & \mathsf{cons}(y, \mathbf{insert}(x, ys)) \\
\mathbf{if}(\mathsf{false}, x, \mathsf{cons}(y, ys)) & \to & \mathsf{cons}(x, \mathsf{cons}(y, ys)) \\
\mathbf{gt}(0, y) & \xrightarrow{=} & \mathsf{false} \\
\mathbf{gt}(\mathsf{s}(x), 0) & \xrightarrow{=} & \mathsf{true} \\
\mathbf{gt}(\mathsf{s}(x), \mathsf{s}(y)) & \xrightarrow{=} & \mathbf{gt}(x, y)
\end{array}
$$

- the recursive **isort** rule is at most applied linearly often
- the recursive **insert** rule is at most applied quadratically often
  - Note: Requires reasoning about **isort**, **insert**, and **if** rules!
- the recursive **if** rule is applied as often as the recursive **insert** rule

Bird's Eye View

Example

$$
\begin{aligned}
\mathbf{isort}(\mathsf{nil}, ys) &\to ys \\
\mathbf{isort}(\mathsf{cons}(x, xs), ys) &\to \mathbf{isort}(xs, \mathbf{insert}(x, ys)) \\
\mathbf{insert}(x, \mathsf{nil}) &\to \mathsf{cons}(x, \mathsf{nil}) \\
\mathbf{insert}(x, \mathsf{cons}(y, ys)) &\to \mathbf{if}(\mathbf{gt}(x, y), x, \mathsf{cons}(y, ys)) \\
\mathbf{if}(\mathsf{true}, x, \mathsf{cons}(y, ys)) &\to \mathsf{cons}(y, \mathbf{insert}(x, ys)) \\
\mathbf{if}(\mathsf{false}, x, \mathsf{cons}(y, ys)) &\to \mathsf{cons}(x, \mathsf{cons}(y, ys)) \\
\mathbf{gt}(0, y) &\xrightarrow{=} \mathsf{false} \\
\mathbf{gt}(\mathsf{s}(x), 0) &\xrightarrow{=} \mathsf{true} \\
\mathbf{gt}(\mathsf{s}(x), \mathsf{s}(y)) &\xrightarrow{=} \mathbf{gt}(x, y)
\end{aligned}
$$

1. abstract terms to integers

Bird's Eye View

Example

$$
\begin{array}{rcll}
\mathbf{isort}(xs', ys) & \xrightarrow{1} & ys & \mid \quad xs' = 1 \\
\mathbf{isort}(\mathsf{cons}(x, xs), ys) & \to & \mathbf{isort}(xs, \mathbf{insert}(x, ys)) & \\
\mathbf{insert}(x, \mathsf{nil}) & \to & \mathsf{cons}(x, \mathsf{nil}) & \\
\mathbf{insert}(x, \mathsf{cons}(y, ys)) & \to & \mathbf{if}(\mathbf{gt}(x, y), x, \mathsf{cons}(y, ys)) & \\
\mathbf{if}(\mathsf{true}, x, \mathsf{cons}(y, ys)) & \to & \mathsf{cons}(y, \mathbf{insert}(x, ys)) & \\
\mathbf{if}(\mathsf{false}, x, \mathsf{cons}(y, ys)) & \to & \mathsf{cons}(x, \mathsf{cons}(y, ys)) & \\
\mathbf{gt}(0, y) & \xrightarrow{=} & \mathsf{false} & \\
\mathbf{gt}(\mathsf{s}(x), 0) & \xrightarrow{=} & \mathsf{true} & \\
\mathbf{gt}(\mathsf{s}(x), \mathsf{s}(y)) & \xrightarrow{=} & \mathbf{gt}(x, y) &
\end{array}
$$

1. abstract terms to integers

Bird's Eye View

Example

$$
\begin{array}{rcll}
\mathbf{isort}(xs', ys) & \xrightarrow{1} & ys & \mid \quad xs' = 1 \\
\mathbf{isort}(xs', ys) & \xrightarrow{1} & \mathbf{isort}(xs, \mathbf{insert}(x, ys)) & \mid \quad xs' = 1 + x + xs \\
\mathbf{insert}(x, \mathsf{nil}) & \to & \mathsf{cons}(x, \mathsf{nil}) & \\
\mathbf{insert}(x, \mathsf{cons}(y, ys)) & \to & \mathbf{if}(\mathbf{gt}(x, y), x, \mathsf{cons}(y, ys)) & \\
\mathbf{if}(\mathsf{true}, x, \mathsf{cons}(y, ys)) & \to & \mathsf{cons}(y, \mathbf{insert}(x, ys)) & \\
\mathbf{if}(\mathsf{false}, x, \mathsf{cons}(y, ys)) & \to & \mathsf{cons}(x, \mathsf{cons}(y, ys)) & \\
\mathbf{gt}(0, y) & \xrightarrow{=} & \mathsf{false} & \\
\mathbf{gt}(\mathsf{s}(x), 0) & \xrightarrow{=} & \mathsf{true} & \\
\mathbf{gt}(\mathsf{s}(x), \mathsf{s}(y)) & \xrightarrow{=} & \mathbf{gt}(x, y) &
\end{array}
$$

1. abstract terms to integers

Bird's Eye View

Example

$$
\begin{array}{rcll}
\mathbf{isort}(xs', ys) & \xrightarrow{1} & ys & \mid \quad xs' = 1 \\
\mathbf{isort}(xs', ys) & \xrightarrow{1} & \mathbf{isort}(xs, \mathbf{insert}(x, ys)) & \mid \quad xs' = 1 + x + xs \\
\mathbf{insert}(x, ys') & \xrightarrow{1} & 2 + x & \mid \quad ys' = 1 \\
\mathbf{insert}(x, \mathsf{cons}(y, ys)) & \to & \mathbf{if}(\mathbf{gt}(x, y), x, \mathsf{cons}(y, ys)) & \\
\mathbf{if}(\mathsf{true}, x, \mathsf{cons}(y, ys)) & \to & \mathsf{cons}(y, \mathbf{insert}(x, ys)) & \\
\mathbf{if}(\mathsf{false}, x, \mathsf{cons}(y, ys)) & \to & \mathsf{cons}(x, \mathsf{cons}(y, ys)) & \\
\mathbf{gt}(0, y) & \xrightarrow{=} & \mathsf{false} & \\
\mathbf{gt}(\mathsf{s}(x), 0) & \xrightarrow{=} & \mathsf{true} & \\
\mathbf{gt}(\mathsf{s}(x), \mathsf{s}(y)) & \xrightarrow{=} & \mathbf{gt}(x, y) &
\end{array}
$$

1. abstract terms to integers

Bird's Eye View

Example

$$
\begin{aligned}
\mathbf{isort}(xs', ys) &\xrightarrow{1} ys & \mid \quad & xs' = 1 \\
\mathbf{isort}(xs', ys) &\xrightarrow{1} \mathbf{isort}(xs, \mathbf{insert}(x, ys)) & \mid \quad & xs' = 1 + x + xs \\
\mathbf{insert}(x, ys') &\xrightarrow{1} 2 + x & \mid \quad & ys' = 1 \\
\mathbf{insert}(x, \mathsf{cons}(y, ys)) &\to \mathbf{if}(\mathbf{gt}(x, y), x, \mathsf{cons}(y, ys)) \\
\mathbf{if}(\mathsf{true}, x, \mathsf{cons}(y, ys)) &\to \mathsf{cons}(y, \mathbf{insert}(x, ys)) \\
\mathbf{if}(\mathsf{false}, x, \mathsf{cons}(y, ys)) &\to \mathsf{cons}(x, \mathsf{cons}(y, ys)) \\
\mathbf{gt}(0, y) &\xrightarrow{=} \mathsf{false} \\
\mathbf{gt}(\mathsf{s}(x), 0) &\xrightarrow{=} \mathsf{true} \\
\mathbf{gt}(\mathsf{s}(x), \mathsf{s}(y)) &\xrightarrow{=} \mathbf{gt}(x, y)
\end{aligned}
$$

1. abstract terms to integers

Bird's Eye View

Example

$$\begin{array}{rcll}
\mathbf{isort}(xs', ys) & \xrightarrow{1} & ys & \mid \quad xs' = 1 \\
\mathbf{isort}(xs', ys) & \xrightarrow{1} & \mathbf{isort}(xs, \mathbf{insert}(x, ys)) & \mid \quad xs' = 1 + x + xs \\
\mathbf{insert}(x, ys') & \xrightarrow{1} & 2 + x & \mid \quad ys' = 1 \\
\mathbf{insert}(x, ys') & \xrightarrow{1} & \mathbf{if}(\mathbf{gt}(x, y), x, ys') & \mid \quad ys' = 1 + y + ys \\
\mathbf{if}(b, x, ys') & \xrightarrow{1} & 1 + y + \mathbf{insert}(x, ys) & \mid \quad b = 1 \wedge ys' = 1 + y + ys \\
\mathbf{if}(b, x, ys') & \xrightarrow{1} & 1 + ys' & \mid \quad b = 1 \wedge ys' = 1 + y + ys \\
\mathbf{gt}(x', y') & \xrightarrow{0} & 1 & \mid \quad x' = 1 \\
\mathbf{gt}(x', y') & \xrightarrow{0} & 1 & \mid \quad x' = 1 + x \wedge y' = 1 \\
\mathbf{gt}(x', y') & \xrightarrow{0} & \mathbf{gt}(x, y) & \mid \quad x' = 1 + x \wedge y' = 1 + y
\end{array}$$

1. abstract terms to integers

Bird's Eye View

Example

$$
\begin{array}{rcll}
\mathbf{isort}(xs', ys) & \xrightarrow{1} & ys & \mid \quad xs' = 1 \\
\mathbf{isort}(xs', ys) & \xrightarrow{1} & \mathbf{isort}(xs, \mathbf{insert}(x, ys)) & \mid \quad xs' = 1 + x + xs \\
\mathbf{insert}(x, ys') & \xrightarrow{1} & 2 + x & \mid \quad ys' = 1 \\
\mathbf{insert}(x, ys') & \xrightarrow{1} & \mathbf{if}(\mathbf{gt}(x, y), x, ys') & \mid \quad ys' = 1 + y + ys \\
\mathbf{if}(b, x, ys') & \xrightarrow{1} & 1 + y + \mathbf{insert}(x, ys) & \mid \quad b = 1 \wedge ys' = 1 + y + ys \\
\mathbf{if}(b, x, ys') & \xrightarrow{1} & 1 + ys' & \mid \quad b = 1 \wedge ys' = 1 + y + ys \\
\mathbf{gt}(x', y') & \xrightarrow{0} & 1 & \mid \quad x' = 1 \\
\mathbf{gt}(x', y') & \xrightarrow{0} & 1 & \mid \quad x' = 1 + x \wedge y' = 1 \\
\mathbf{gt}(x', y') & \xrightarrow{0} & \mathbf{gt}(x, y) & \mid \quad x' = 1 + x \wedge y' = 1 + y
\end{array}
$$

1. abstract terms to integers
   - note: variables range over $\mathbb{N}$
   - just $+$ and $\cdot$

Bird's Eye View

Example

$$
\begin{array}{rcll}
\mathbf{isort}(xs', ys) & \xrightarrow{1} & ys & \mid \quad xs' = 1 \\
\mathbf{isort}(xs', ys) & \xrightarrow{1} & \mathbf{isort}(xs, \mathbf{insert}(x, ys)) & \mid \quad xs' = 1 + x + xs \\
\mathbf{insert}(x, ys') & \xrightarrow{1} & 2 + x & \mid \quad ys' = 1 \\
\mathbf{insert}(x, ys') & \xrightarrow{1} & \mathbf{if}(\mathbf{gt}(x, y), x, ys') & \mid \quad ys' = 1 + y + ys \\
\mathbf{if}(b, x, ys') & \xrightarrow{1} & 1 + y + \mathbf{insert}(x, ys) & \mid \quad b = 1 \wedge ys' = 1 + y + ys \\
\mathbf{if}(b, x, ys') & \xrightarrow{1} & 1 + ys' & \mid \quad b = 1 \wedge ys' = 1 + y + ys \\
\mathbf{gt}(x', y') & \xrightarrow{0} & 1 & \mid \quad x' = 1 \\
\mathbf{gt}(x', y') & \xrightarrow{0} & 1 & \mid \quad x' = 1 + x \wedge y' = 1 \\
\mathbf{gt}(x', y') & \xrightarrow{0} & \mathbf{gt}(x, y) & \mid \quad x' = 1 + x \wedge y' = 1 + y
\end{array}
$$

1. abstract terms to integers
   - note: variables range over $\mathbb{N}$
   - just $+$ and $\cdot$
2. analyze result-size for bottom-SCC using standard tools

Bird's Eye View

Example

$$\begin{array}{llll}
\mathbf{gt}(x', y') & \xrightarrow{0} 1 & \mid & x' = 1 \\
\mathbf{gt}(x', y') & \xrightarrow{0} 1 & \mid & x' = 1 + x \wedge y' = 1 \\
\mathbf{gt}(x', y') & \xrightarrow{0} \mathbf{gt}(x, y) & \mid & x' = 1 + x \wedge y' = 1 + y
\end{array}$$

1. abstract terms to integers
   - note: variables range over $\mathbb{N}$
   - just $+$ and $\cdot$
2. analyze result-size for bottom-SCC using standard tools

Bird's Eye View

Example

$$
\begin{array}{lcll}
\mathbf{gt}(x', y') & \xrightarrow{0} & 1 & \mid \quad x' = 1 \\
\mathbf{gt}(x', y') & \xrightarrow{0} & 1 & \mid \quad x' = 1 + x \wedge y' = 1 \\
\mathbf{gt}(x', y') & \xrightarrow{0} & \mathbf{gt}(x, y) & \mid \quad x' = 1 + x \wedge y' = 1 + y
\end{array}
$$

1. abstract terms to integers
   - note: variables range over $\mathbb{N}$
   - just $+$ and $\cdot$
2. analyze result-size for bottom-SCC using standard tools
3. analyze runtime of bottom-SCC using standard tools

Bird's Eye View

Example

$$
\begin{array}{rcll}
\mathbf{isort}(xs', ys) & \xrightarrow{1} & ys & \mid \quad xs' = 1 \\
\mathbf{isort}(xs', ys) & \xrightarrow{1} & \mathbf{isort}(xs, \mathbf{insert}(x, ys)) & \mid \quad xs' = 1 + x + xs \\
\mathbf{insert}(x, ys') & \xrightarrow{1} & 2 + x & \mid \quad ys' = 1 \\
\mathbf{insert}(x, ys') & \xrightarrow{1} & \mathbf{if}(b, x, ys') & \mid \quad ys' = 1 + y + ys \wedge b \leq 1 \\
\mathbf{if}(b, x, ys') & \xrightarrow{1} & 1 + y + \mathbf{insert}(x, ys) & \mid \quad b = 1 \wedge ys' = 1 + y + ys \\
\mathbf{if}(b, x, ys') & \xrightarrow{1} & 1 + ys' & \mid \quad b = 1 \wedge ys' = 1 + y + ys
\end{array}
$$

1. abstract terms to integers
   - note: variables range over $\mathbb{N}$
   - just $+$ and $\cdot$
2. analyze result-size for bottom-SCC using standard tools
3. analyze runtime of bottom-SCC using standard tools

Abstracting Terms to Integers

Terminating Variants

Term Rewriting	**Integer Transition Systems**
arbitrary start terms	ground start terms

Example

$\mathbf{h}(x) \to \mathbf{f}(\mathbf{g}(x)) \qquad \mathbf{f}(x) \to \mathbf{f}(x) \qquad \mathbf{g}(\mathsf{a}) \xrightarrow{=} \mathbf{g}(\mathsf{a})$

Terminating Variants

Term Rewriting	**Integer Transition Systems**
arbitrary start terms	ground start terms

Example

$$\mathbf{h}(x) \longrightarrow \mathbf{f}(\mathbf{g}(x)) \qquad \mathbf{f}(x) \longrightarrow \mathbf{f}(x) \qquad \mathbf{g}(\mathsf{a}) \xrightarrow{=} \mathbf{g}(\mathsf{a})$$

innermost rewriting: $\mathbf{h}(x) \longrightarrow \mathbf{f}(\mathbf{g}(x)) \longrightarrow \mathbf{f}(\mathbf{g}(x)) \longrightarrow \ldots$

Terminating Variants

Term Rewriting	**Integer Transition Systems**
arbitrary start terms	ground start terms

Example

$$\mathbf{h}(x) \longrightarrow \mathbf{f}(\mathbf{g}(x)) \qquad \mathbf{f}(x) \longrightarrow \mathbf{f}(x) \qquad \mathbf{g}(\mathsf{a}) \xrightarrow{=} \mathbf{g}(\mathsf{a})$$

innermost rewriting: $\mathbf{h}(x) \longrightarrow \mathbf{f}(\mathbf{g}(x)) \longrightarrow \mathbf{f}(\mathbf{g}(x)) \longrightarrow \ldots$ $\mathcal{O}(\infty)$

Terminating Variants

Term Rewriting	**Integer Transition Systems**
arbitrary start terms	ground start terms

Example

$$\mathbf{h}(x) \longrightarrow \mathbf{f}(\mathbf{g}(x)) \qquad \mathbf{f}(x) \longrightarrow \mathbf{f}(x) \qquad \mathbf{g}(\mathsf{a}) \xrightarrow{=} \mathbf{g}(\mathsf{a})$$

innermost rewriting: $\mathbf{h}(x) \longrightarrow \mathbf{f}(\mathbf{g}(x)) \longrightarrow \mathbf{f}(\mathbf{g}(x)) \longrightarrow \dots$ $\mathcal{O}(\infty)$

- Just ground rewriting?

Terminating Variants

Term Rewriting	**Integer Transition Systems**
arbitrary start terms	ground start terms

Example

$$\mathbf{h}(x) \longrightarrow \mathbf{f}(\mathbf{g}(x)) \qquad \mathbf{f}(x) \longrightarrow \mathbf{f}(x) \qquad \mathbf{g}(\mathsf{a}) \xrightarrow{=} \mathbf{g}(\mathsf{a})$$

innermost rewriting: $\mathbf{h}(x) \longrightarrow \mathbf{f}(\mathbf{g}(x)) \longrightarrow \mathbf{f}(\mathbf{g}(x)) \longrightarrow \dots$ $\mathcal{O}(\infty)$

ground rewriting: $\mathbf{h}(\mathsf{a}) \longrightarrow \mathbf{f}(\mathbf{g}(\mathsf{a})) \xrightarrow{=} \mathbf{f}(\mathbf{g}(\mathsf{a})) \xrightarrow{=} \dots$

- Just ground rewriting?

Terminating Variants

Term Rewriting	**Integer Transition Systems**
arbitrary start terms	ground start terms

Example

$$\mathbf{h}(x) \longrightarrow \mathbf{f}(\mathbf{g}(x)) \qquad \mathbf{f}(x) \longrightarrow \mathbf{f}(x) \qquad \mathbf{g}(\mathsf{a}) \xrightarrow{=} \mathbf{g}(\mathsf{a})$$

innermost rewriting: $\mathbf{h}(x) \longrightarrow \mathbf{f}(\mathbf{g}(x)) \longrightarrow \mathbf{f}(\mathbf{g}(x)) \longrightarrow \ldots$ $\mathcal{O}(\infty)$

ground rewriting: $\mathbf{h}(\mathsf{a}) \longrightarrow \mathbf{f}(\mathbf{g}(\mathsf{a})) \xrightarrow{=} \mathbf{f}(\mathbf{g}(\mathsf{a})) \xrightarrow{=} \ldots$ $\mathcal{O}(1)$

- Just ground rewriting?

Terminating Variants

Term Rewriting	**Integer Transition Systems**
arbitrary start terms	ground start terms

Example

$$\mathbf{h}(x) \to \mathbf{f}(\mathbf{g}(x)) \qquad \mathbf{f}(x) \to \mathbf{f}(x) \qquad \mathbf{g}(\mathsf{a}) \xrightarrow{=} \mathbf{g}(\mathsf{a})$$

innermost rewriting: $\mathbf{h}(x) \to \mathbf{f}(\mathbf{g}(x)) \to \mathbf{f}(\mathbf{g}(x)) \to \ldots$ $\mathcal{O}(\infty)$

ground rewriting: $\mathbf{h}(\mathsf{a}) \to \mathbf{f}(\mathbf{g}(\mathsf{a})) \xrightarrow{=} \mathbf{f}(\mathbf{g}(\mathsf{a})) \xrightarrow{=} \ldots$ $\mathcal{O}(1)$

- Just ground rewriting?
- Add terminating variant of relative rules!

Terminating Variants

Term Rewriting	**Integer Transition Systems**
arbitrary start terms	ground start terms

Example

$$\mathbf{h}(x) \to \mathbf{f}(\mathbf{g}(x)) \qquad \mathbf{f}(x) \to \mathbf{f}(x) \qquad \mathbf{g}(\mathrm{a}) \xrightarrow{=} \mathbf{g}(\mathrm{a})$$

innermost rewriting: $\mathbf{h}(x) \to \mathbf{f}(\mathbf{g}(x)) \to \mathbf{f}(\mathbf{g}(x)) \to \dots$ $\mathcal{O}(\infty)$

ground rewriting: $\mathbf{h}(\mathrm{a}) \to \mathbf{f}(\mathbf{g}(\mathrm{a})) \xrightarrow{=} \mathbf{f}(\mathbf{g}(\mathrm{a})) \xrightarrow{=} \dots$ $\mathcal{O}(1)$

- Just ground rewriting?
- Add terminating variant of relative rules!

Definition

$\mathcal{N}$ is a terminating variant of $\mathcal{S}$ if $\mathcal{N}$ terminates and every $\mathcal{N}$-normal form is an $\mathcal{S}$-normal form.

Terminating Variants

Term Rewriting	**Integer Transition Systems**
arbitrary start terms	ground start terms

Example

$$\mathbf{h}(x) \to \mathbf{f}(\mathbf{g}(x)) \qquad \mathbf{f}(x) \to \mathbf{f}(x) \qquad \mathbf{g}(\mathsf{a}) \xrightarrow{=} \mathbf{g}(\mathsf{a}) \qquad \mathbf{g}(\mathsf{a}) \xrightarrow{=} \mathsf{a}$$

innermost rewriting: $\mathbf{h}(x) \to \mathbf{f}(\mathbf{g}(x)) \to \mathbf{f}(\mathbf{g}(x)) \to \ldots$ $\mathcal{O}(\infty)$

ground rewriting: $\mathbf{h}(\mathsf{a}) \to \mathbf{f}(\mathbf{g}(\mathsf{a})) \xrightarrow{=} \mathbf{f}(\mathbf{g}(\mathsf{a})) \xrightarrow{=} \ldots$ $\mathcal{O}(1)$

- Just ground rewriting?
- Add terminating variant of relative rules!

Definition

$\mathcal{N}$ is a terminating variant of $\mathcal{S}$ if $\mathcal{N}$ terminates and every $\mathcal{N}$-normal form is an $\mathcal{S}$-normal form.

Terminating Variants

Term Rewriting	**Integer Transition Systems**
arbitrary start terms	ground start terms

Example

$$\mathbf{h}(x) \to \mathbf{f}(\mathbf{g}(x)) \qquad \mathbf{f}(x) \to \mathbf{f}(x) \qquad \mathbf{g}(\mathsf{a}) \xrightarrow{=} \mathbf{g}(\mathsf{a}) \qquad \mathbf{g}(\mathsf{a}) \xrightarrow{=} \mathsf{a}$$

innermost rewriting: $\mathbf{h}(x) \to \mathbf{f}(\mathbf{g}(x)) \to \mathbf{f}(\mathbf{g}(x)) \to \dots$ $\mathcal{O}(\infty)$

ground rewriting: $\mathbf{h}(\mathsf{a}) \to \mathbf{f}(\mathbf{g}(\mathsf{a})) \xrightarrow{=} \mathbf{f}(\mathbf{g}(\mathsf{a})) \xrightarrow{=} \dots$ $\mathcal{O}(1)$

with terminating variant: $\mathbf{h}(\mathsf{a}) \to \mathbf{f}(\mathbf{g}(\mathsf{a})) \xrightarrow{=} \mathbf{f}(\mathsf{a}) \to \mathbf{f}(\mathsf{a}) \to \dots$

- Just ground rewriting?
- Add terminating variant of relative rules!

Definition

$\mathcal{N}$ is a terminating variant of $\mathcal{S}$ if $\mathcal{N}$ terminates and every $\mathcal{N}$-normal form is an $\mathcal{S}$-normal form.

Terminating Variants

Term Rewriting	**Integer Transition Systems**
arbitrary start terms	ground start terms

Example

$\mathbf{h}(x) \to \mathbf{f}(\mathbf{g}(x)) \qquad \mathbf{f}(x) \to \mathbf{f}(x) \qquad \mathbf{g}(\mathsf{a}) \xrightarrow{=} \mathbf{g}(\mathsf{a}) \qquad \mathbf{g}(\mathsf{a}) \xrightarrow{=} \mathsf{a}$

innermost rewriting: $\mathbf{h}(x) \to \mathbf{f}(\mathbf{g}(x)) \to \mathbf{f}(\mathbf{g}(x)) \to \dots$ $\mathcal{O}(\infty)$

ground rewriting: $\mathbf{h}(\mathsf{a}) \to \mathbf{f}(\mathbf{g}(\mathsf{a})) \xrightarrow{=} \mathbf{f}(\mathbf{g}(\mathsf{a})) \xrightarrow{=} \dots$ $\mathcal{O}(1)$

with terminating variant: $\mathbf{h}(\mathsf{a}) \to \mathbf{f}(\mathbf{g}(\mathsf{a})) \xrightarrow{=} \mathbf{f}(\mathsf{a}) \to \mathbf{f}(\mathsf{a}) \to \dots$ $\mathcal{O}(\infty)$

- Just ground rewriting?
- Add terminating variant of relative rules!

Definition

$\mathcal{N}$ is a terminating variant of $\mathcal{S}$ if $\mathcal{N}$ terminates and every $\mathcal{N}$-normal form is an $\mathcal{S}$-normal form.

Completion

Term Rewriting	**Integer Transition Systems**
arbitrary matchers	just integer substitutions

Example

$$\mathbf{f}(x) \longrightarrow \mathbf{f}(\mathbf{g}(\mathsf{a})) \qquad \mathbf{g}(\mathsf{b}(\mathsf{a})) \longrightarrow \mathsf{a}$$

Completion

Term Rewriting	**Integer Transition Systems**
arbitrary matchers	just integer substitutions

Example

$$\mathbf{f}(x) \to \mathbf{f}(\mathbf{g}(\mathsf{a})) \qquad \mathbf{g}(\mathsf{b}(\mathsf{a})) \to \mathsf{a}$$

original TRS: $\mathbf{f}(\mathsf{a}) \to \mathbf{f}(\mathbf{g}(\mathsf{a})) \to \mathbf{f}(\mathbf{g}(\mathsf{a})) \to \dots$

Completion

Term Rewriting	**Integer Transition Systems**
arbitrary matchers	just integer substitutions

Example

$$\mathbf{f}(x) \longrightarrow \mathbf{f}(\mathbf{g}(\mathsf{a})) \qquad \mathbf{g}(\mathsf{b}(\mathsf{a})) \longrightarrow \mathsf{a}$$

original TRS: $\mathbf{f}(\mathsf{a}) \longrightarrow \mathbf{f}(\mathbf{g}(\mathsf{a})) \longrightarrow \mathbf{f}(\mathbf{g}(\mathsf{a})) \longrightarrow \ldots$ $\mathcal{O}(\infty)$

Completion

Term Rewriting	**Integer Transition Systems**
arbitrary matchers	just integer substitutions

Example

$$\mathsf{f}(x) \longrightarrow \mathsf{f}(\mathsf{g}(\mathsf{a})) \qquad\qquad \mathsf{g}(\mathsf{b}(\mathsf{a})) \longrightarrow \mathsf{a}$$

original TRS: $\mathsf{f}(\mathsf{a}) \longrightarrow \mathsf{f}(\mathsf{g}(\mathsf{a})) \longrightarrow \mathsf{f}(\mathsf{g}(\mathsf{a})) \longrightarrow \ldots$ $\mathcal{O}(\infty)$

resulting ITS: $\mathsf{f}(1) \xrightarrow{1} \mathsf{f}(\mathsf{g}(1))$

Completion

Term Rewriting	Integer Transition Systems
arbitrary matchers	just integer substitutions

Example

$$\mathbf{f}(x) \longrightarrow \mathbf{f}(\mathbf{g}(\mathsf{a})) \qquad \mathbf{g}(\mathsf{b}(\mathsf{a})) \longrightarrow \mathsf{a}$$

original TRS: $\mathbf{f}(\mathsf{a}) \longrightarrow \mathbf{f}(\mathbf{g}(\mathsf{a})) \longrightarrow \mathbf{f}(\mathbf{g}(\mathsf{a})) \longrightarrow \ldots$ $\mathcal{O}(\infty)$

resulting ITS: $\mathbf{f}(1) \xrightarrow{1} \mathbf{f}(\mathbf{g}(1))$ $\mathcal{O}(1)$

Completion

Term Rewriting	Integer Transition Systems
arbitrary matchers	just integer substitutions

Example

$$\mathbf{f}(x) \to \mathbf{f}(\mathbf{g}(\mathsf{a})) \qquad \mathbf{g}(\mathsf{b}(\mathsf{a})) \to \mathsf{a}$$

original TRS: $\mathbf{f}(\mathsf{a}) \to \mathbf{f}(\mathbf{g}(\mathsf{a})) \to \mathbf{f}(\mathbf{g}(\mathsf{a})) \to \dots$ $\mathcal{O}(\infty)$

resulting ITS: $\mathbf{f}(1) \xrightarrow{1} \mathbf{f}(\mathbf{g}(1))$ $\mathcal{O}(1)$

Definition

A TRS is completely defined iff its ground normal forms do not contain defined symbols.

Completion

Term Rewriting	**Integer Transition Systems**
arbitrary matchers	just integer substitutions

Example

$$\mathbf{f}(x) \to \mathbf{f}(\mathbf{g}(\mathsf{a})) \qquad \mathbf{g}(\mathsf{b}(\mathsf{a})) \to \mathsf{a} \qquad \mathbf{g}(x) \xrightarrow{=} \mathsf{a}$$

original TRS: $\mathbf{f}(\mathsf{a}) \to \mathbf{f}(\mathbf{g}(\mathsf{a})) \to \mathbf{f}(\mathbf{g}(\mathsf{a})) \to \dots$ $\mathcal{O}(\infty)$

resulting ITS: $\mathbf{f}(1) \xrightarrow{1} \mathbf{f}(\mathbf{g}(1))$ $\mathcal{O}(1)$

Definition

A TRS is completely defined iff its ground normal forms do not contain defined symbols.

Completion

Term Rewriting	Integer Transition Systems
arbitrary matchers	just integer substitutions

Example

$$\mathbf{f}(x) \longrightarrow \mathbf{f}(\mathbf{g}(\mathsf{a})) \qquad \mathbf{g}(\mathsf{b}(\mathsf{a})) \longrightarrow \mathsf{a} \qquad \mathbf{g}(x) \xrightarrow{=} \mathsf{a}$$

original TRS: $\mathbf{f}(\mathsf{a}) \longrightarrow \mathbf{f}(\mathbf{g}(\mathsf{a})) \longrightarrow \mathbf{f}(\mathbf{g}(\mathsf{a})) \longrightarrow \dots$ $\mathcal{O}(\infty)$

resulting ITS: $\mathbf{f}(1) \xrightarrow{1} \mathbf{f}(\mathbf{g}(1))$ $\mathcal{O}(1)$

ITS after completion: $\mathbf{f}(1) \xrightarrow{1} \mathbf{f}(\mathbf{g}(1)) \xrightarrow{0} \mathbf{f}(1) \xrightarrow{1} \mathbf{f}(\mathbf{g}(1)) \xrightarrow{0} \dots$

Definition

A TRS is completely defined iff its ground normal forms do not contain defined symbols.

Completion

Term Rewriting	**Integer Transition Systems**
arbitrary matchers	just integer substitutions

Example

$$\mathbf{f}(x) \rightarrow \mathbf{f}(\mathbf{g}(\mathsf{a})) \qquad \mathbf{g}(\mathsf{b}(\mathsf{a})) \rightarrow \mathsf{a} \qquad \mathbf{g}(x) \xrightarrow{=} \mathsf{a}$$

original TRS: $\mathbf{f}(\mathsf{a}) \rightarrow \mathbf{f}(\mathbf{g}(\mathsf{a})) \rightarrow \mathbf{f}(\mathbf{g}(\mathsf{a})) \rightarrow \dots$ $\qquad \mathcal{O}(\infty)$

resulting ITS: $\mathbf{f}(1) \xrightarrow{1} \mathbf{f}(\mathbf{g}(1))$ $\qquad \mathcal{O}(1)$

ITS after completion: $\mathbf{f}(1) \xrightarrow{1} \mathbf{f}(\mathbf{g}(1)) \xrightarrow{0} \mathbf{f}(1) \xrightarrow{1} \mathbf{f}(\mathbf{g}(1)) \xrightarrow{0} \dots$ $\qquad \mathcal{O}(\infty)$

Definition

A TRS is completely defined iff its ground normal forms do not contain defined symbols.

Completion

Term Rewriting	**Integer Transition Systems**
arbitrary matchers	just integer substitutions

Example

$$\mathbf{f}(x) \longrightarrow \mathbf{f}(\mathbf{g}(\mathsf{a})) \qquad \mathbf{g}(\mathsf{b}(\mathsf{a})) \longrightarrow \mathsf{a} \qquad \mathbf{g}(x) \xrightarrow{=} \mathsf{a}$$

original TRS: $\mathbf{f}(\mathsf{a}) \longrightarrow \mathbf{f}(\mathbf{g}(\mathsf{a})) \longrightarrow \mathbf{f}(\mathbf{g}(\mathsf{a})) \longrightarrow \ldots$ $\mathcal{O}(\infty)$

resulting ITS: $\mathbf{f}(1) \xrightarrow{1} \mathbf{f}(\mathbf{g}(1))$ $\mathcal{O}(1)$

ITS after completion: $\mathbf{f}(1) \xrightarrow{1} \mathbf{f}(\mathbf{g}(1)) \xrightarrow{0} \mathbf{f}(1) \xrightarrow{1} \mathbf{f}(\mathbf{g}(1)) \xrightarrow{0} \ldots$ $\mathcal{O}(\infty)$

Definition

A TRS is completely defined iff its well-typed ground normal forms do not contain defined symbols.

Completion

Term Rewriting	**Integer Transition Systems**
arbitrary matchers	just integer substitutions

Example

$$\mathbf{f}(x) \to \mathbf{f}(\mathbf{g}(\mathsf{a})) \qquad \mathbf{g}(\mathsf{b}(\mathsf{a})) \to \mathsf{a} \qquad \mathbf{g}(x) \xrightarrow{=} \mathsf{a}$$

original TRS: $\mathbf{f}(\mathsf{a}) \to \mathbf{f}(\mathbf{g}(\mathsf{a})) \to \mathbf{f}(\mathbf{g}(\mathsf{a})) \to \dots$ $\mathcal{O}(\infty)$

resulting ITS: $\mathbf{f}(1) \xrightarrow{1} \mathbf{f}(\mathbf{g}(1))$ $\mathcal{O}(1)$

ITS after completion: $\mathbf{f}(1) \xrightarrow{1} \mathbf{f}(\mathbf{g}(1)) \xrightarrow{0} \mathbf{f}(1) \xrightarrow{1} \mathbf{f}(\mathbf{g}(1)) \xrightarrow{0} \dots$ $\mathcal{O}(\infty)$

Definition

A TRS is completely defined iff its well-typed ground normal forms do not contain defined symbols.

TRS not completely defined? $\curvearrowright$ Add suitable terminating variant!

Bird's Eye View

Example

$$
\begin{array}{rcll}
\mathbf{isort}(xs', ys) & \xrightarrow{1} & ys & \mid \quad xs' = 1 \\
\mathbf{isort}(xs', ys) & \xrightarrow{1} & \mathbf{isort}(xs, \mathbf{insert}(x, ys)) & \mid \quad xs' = 1 + x + xs \\
\mathbf{insert}(x, ys') & \xrightarrow{1} & 2 + x & \mid \quad ys' = 1 \\
\mathbf{insert}(x, ys') & \xrightarrow{1} & \mathbf{if}(b, x, ys') & \mid \quad ys' = 1 + y + ys \wedge b \leq 1 \\
\mathbf{if}(b, x, ys') & \xrightarrow{1} & 1 + y + \mathbf{insert}(x, ys) & \mid \quad b = 1 \wedge ys' = 1 + y + ys \\
\mathbf{if}(b, x, ys') & \xrightarrow{1} & 1 + ys' & \mid \quad b = 1 \wedge ys' = 1 + y + ys
\end{array}
$$

1. abstract terms to integers
2. analyze result-size for bottom-SCC using standard tools
3. analyze runtime of bottom-SCC using standard tools

Bird's Eye View

Example

$$
\begin{array}{rcll}
\mathbf{insert}(x, ys') & \xrightarrow{1} & 2 + x & \mid \quad ys' = 1 \\
\mathbf{insert}(x, ys') & \xrightarrow{1} & \mathbf{if}(b, x, ys') & \mid \quad ys' = 1 + y + ys \land b \leq 1 \\
\mathbf{if}(b, x, ys') & \xrightarrow{1} & 1 + y + \mathbf{insert}(x, ys) & \mid \quad b = 1 \land ys' = 1 + y + ys \\
\mathbf{if}(b, x, ys') & \xrightarrow{1} & 1 + ys' & \mid \quad b = 1 \land ys' = 1 + y + ys
\end{array}
$$

1. abstract terms to integers
2. analyze result-size for bottom-SCC using standard tools
3. analyze runtime of bottom-SCC using standard tools

Analyze Size Using Standard Tools

Using Runtime Analysis to Compute Size Bounds

Example

$$
\begin{array}{rcll}
\textbf{insert}(x, ys') & \xrightarrow{1} & 2 + x & \mid\quad ys' = 1 \\
\textbf{insert}(x, ys') & \xrightarrow{1} & \textbf{if}(b, x, ys') & \mid\quad ys' = 1 + y + ys \wedge b \leq 1 \\
\textbf{if}(b, x, ys') & \xrightarrow{1} & 1 + y + \textbf{insert}(x, ys) & \mid\quad b = 1 \wedge ys' = 1 + y + ys \\
\textbf{if}(b, x, ys') & \xrightarrow{1} & 1 + ys' & \mid\quad b = 1 \wedge ys' = 1 + y + ys
\end{array}
$$

Using Runtime Analysis to Compute Size Bounds

Example

$$
\begin{array}{rcll}
\mathbf{insert}(x, ys') & \xrightarrow{1} & 2 + x & \mid \quad ys' = 1 \\
\mathbf{insert}(x, ys') & \xrightarrow{1} & \mathbf{if}(b, x, ys') & \mid \quad ys' = 1 + y + ys \land b \leq 1 \\
\mathbf{if}(b, x, ys') & \xrightarrow{1} & 1 + y + \mathbf{insert}(x, ys) & \mid \quad b = 1 \land ys' = 1 + y + ys \\
\mathbf{if}(b, x, ys') & \xrightarrow{1} & 1 + ys' & \mid \quad b = 1 \land ys' = 1 + y + ys
\end{array}
$$

Idea: move “integer context” to costs

Using Runtime Analysis to Compute Size Bounds

Example

$$\begin{array}{rclcl}
\mathbf{insert}(x, ys') & \xrightarrow{2+x} & 2 + x & \mid & ys' = 1 \\
\mathbf{insert}(x, ys') & \xrightarrow{1} & \mathbf{if}(b, x, ys') & \mid & ys' = 1 + y + ys \wedge b \leq 1 \\
\mathbf{if}(b, x, ys') & \xrightarrow{1} & 1 + y + \mathbf{insert}(x, ys) & \mid & b = 1 \wedge ys' = 1 + y + ys \\
\mathbf{if}(b, x, ys') & \xrightarrow{1} & 1 + ys' & \mid & b = 1 \wedge ys' = 1 + y + ys
\end{array}$$

Idea: move "integer context" to costs

Using Runtime Analysis to Compute Size Bounds

Example

$$
\begin{array}{rclcl}
\mathbf{insert}(x, ys') & \xrightarrow{2+x} & 2 + x & \mid & ys' = 1 \\
\mathbf{insert}(x, ys') & \xrightarrow{0} & \mathbf{if}(b, x, ys') & \mid & ys' = 1 + y + ys \wedge b \leq 1 \\
\mathbf{if}(b, x, ys') & \xrightarrow{1} & 1 + y + \mathbf{insert}(x, ys) & \mid & b = 1 \wedge ys' = 1 + y + ys \\
\mathbf{if}(b, x, ys') & \xrightarrow{1} & 1 + ys' & \mid & b = 1 \wedge ys' = 1 + y + ys
\end{array}
$$

Idea: move "integer context" to costs

Using Runtime Analysis to Compute Size Bounds

Example

$$
\begin{array}{rclcl}
\mathbf{insert}(x, ys') & \xrightarrow{2+x} & 2 + x & \mid & ys' = 1 \\
\mathbf{insert}(x, ys') & \xrightarrow{0} & \mathbf{if}(b, x, ys') & \mid & ys' = 1 + y + ys \wedge b \leq 1 \\
\mathbf{if}(b, x, ys') & \xrightarrow{1+y} & 1 + y + \mathbf{insert}(x, ys) & \mid & b = 1 \wedge ys' = 1 + y + ys \\
\mathbf{if}(b, x, ys') & \xrightarrow{1} & 1 + ys' & \mid & b = 1 \wedge ys' = 1 + y + ys
\end{array}
$$

Idea: move "integer context" to costs

Using Runtime Analysis to Compute Size Bounds

Example

$$
\begin{array}{rclcl}
\mathbf{insert}(x, ys') & \xrightarrow{2+x} & 2 + x & \mid & ys' = 1 \\
\mathbf{insert}(x, ys') & \xrightarrow{0} & \mathbf{if}(b, x, ys') & \mid & ys' = 1 + y + ys \land b \leq 1 \\
\mathbf{if}(b, x, ys') & \xrightarrow{1+y} & 1 + y + \mathbf{insert}(x, ys) & \mid & b = 1 \land ys' = 1 + y + ys \\
\mathbf{if}(b, x, ys') & \xrightarrow{1+ys'} & 1 + ys' & \mid & b = 1 \land ys' = 1 + y + ys
\end{array}
$$

Idea: move "integer context" to costs

Using Runtime Analysis to Compute Size Bounds

Example

$$\begin{array}{rclcl}
\mathbf{insert}(x, ys') & \xrightarrow{2+x} & 2 + x & | & ys' = 1 \\
\mathbf{insert}(x, ys') & \xrightarrow{0} & \mathbf{if}(b, x, ys') & | & ys' = 1 + y + ys \wedge b \leq 1 \\
\mathbf{if}(b, x, ys') & \xrightarrow{1+y} & 1 + y + \mathbf{insert}(x, ys) & | & b = 1 \wedge ys' = 1 + y + ys \\
\mathbf{if}(b, x, ys') & \xrightarrow{1+ys'} & 1 + ys' & | & b = 1 \wedge ys' = 1 + y + ys
\end{array}$$

Idea: move "integer context" to costs $\curvearrowright 1 + x + ys'$

Using Runtime Analysis to Compute Size Bounds

Example

$$
\begin{array}{rcll}
\mathbf{insert}(x, ys') & \xrightarrow{2+x} & 2 + x & \mid \quad ys' = 1 \\
\mathbf{insert}(x, ys') & \xrightarrow{0} & \mathbf{if}(b, x, ys') & \mid \quad ys' = 1 + y + ys \wedge b \leq 1 \\
\mathbf{if}(b, x, ys') & \xrightarrow{1+y} & 1 + y + \mathbf{insert}(x, ys) & \mid \quad b = 1 \wedge ys' = 1 + y + ys \\
\mathbf{if}(b, x, ys') & \xrightarrow{1+ys'} & 1 + ys' & \mid \quad b = 1 \wedge ys' = 1 + y + ys
\end{array}
$$

Idea: move "integer context" to costs $\curvearrowright 1 + x + ys'$

Example

$$
\mathbf{f}(x) \quad \xrightarrow{1} \quad 2 + x \cdot \mathbf{f}(x - 1) \quad \mid \quad x > 0
$$

Using Runtime Analysis to Compute Size Bounds

Example

$$\begin{array}{rcllcl}
\textbf{insert}(x, ys') & \xrightarrow{2+x} & 2 + x & & | & ys' = 1 \\
\textbf{insert}(x, ys') & \xrightarrow{0} & \textbf{if}(b, x, ys') & & | & ys' = 1 + y + ys \wedge b \leq 1 \\
\textbf{if}(b, x, ys') & \xrightarrow{1+y} & 1 + y + \textbf{insert}(x, ys) & & | & b = 1 \wedge ys' = 1 + y + ys \\
\textbf{if}(b, x, ys') & \xrightarrow{1+ys'} & 1 + ys' & & | & b = 1 \wedge ys' = 1 + y + ys
\end{array}$$

Idea: move "integer context" to costs $\curvearrowright 1 + x + ys'$

Example

$$\mathbf{f}(x) \xrightarrow{1} 2 + x \cdot \mathbf{f}(x - 1) \quad | \quad x > 0$$

Idea: use accumulator

Using Runtime Analysis to Compute Size Bounds

Example

$$
\begin{array}{rcll}
\mathbf{insert}(x, ys') & \xrightarrow{2+x} & 2 + x & \mid \quad ys' = 1 \\
\mathbf{insert}(x, ys') & \xrightarrow{0} & \mathbf{if}(b, x, ys') & \mid \quad ys' = 1 + y + ys \wedge b \leq 1 \\
\mathbf{if}(b, x, ys') & \xrightarrow{1+y} & 1 + y + \mathbf{insert}(x, ys) & \mid \quad b = 1 \wedge ys' = 1 + y + ys \\
\mathbf{if}(b, x, ys') & \xrightarrow{1+ys'} & 1 + ys' & \mid \quad b = 1 \wedge ys' = 1 + y + ys
\end{array}
$$

Idea: move "integer context" to costs $\curvearrowright 1 + x + ys'$

Example

$$
\begin{array}{rcll}
\mathbf{f}(x) & \xrightarrow{1} & 2 + x \cdot \mathbf{f}(x-1) & \mid \quad x > 0 \\
\mathbf{f}(x, acc) & \xrightarrow{acc \cdot 2} & 2 + x \cdot \mathbf{f}(x-1, acc \cdot x) & \mid \quad x > 0
\end{array}
$$

Idea: use accumulator

Bird's Eye View

Example

$$
\begin{array}{rcll}
\mathbf{isort}(xs', ys) & \xrightarrow{1} & ys & \mid \quad xs' = 1 \\
\mathbf{isort}(xs', ys) & \xrightarrow{1} & \mathbf{isort}(xs, \mathbf{insert}(x, ys)) & \mid \quad xs' = 1 + x + xs \\
\mathbf{insert}(x, ys') & \xrightarrow{1} & 2 + x & \mid \quad ys' = 1 \\
\mathbf{insert}(x, ys') & \xrightarrow{1} & \mathbf{if}(b, x, ys') & \mid \quad ys' = 1 + y + ys \wedge b \leq 1 \\
\mathbf{if}(b, x, ys') & \xrightarrow{1} & 1 + y + \mathbf{insert}(x, ys) & \mid \quad b = 1 \wedge ys' = 1 + y + ys \\
\mathbf{if}(b, x, ys') & \xrightarrow{1} & 1 + ys' & \mid \quad b = 1 \wedge ys' = 1 + y + ys
\end{array}
$$

1. abstract terms to integers
2. analyze result-size for bottom-SCC using standard tools
3. analyze runtime of bottom-SCC using standard tools

Bird's Eye View

Example

$$
\begin{array}{llll}
\mathbf{isort}(xs', ys) & \xrightarrow{1} ys & \mid & xs' = 1 \\
\mathbf{isort}(xs', ys) & \xrightarrow{1} \mathbf{isort}(xs, \mathbf{insert}(x, ys)) & \mid & xs' = 1 + x + xs
\end{array}
$$

1. abstract terms to integers
2. analyze result-size for bottom-SCC using standard tools
3. analyze runtime of bottom-SCC using standard tools

Analyze Runtime Using Standard Tools

Removing Nested Function Calls

Example

$$
\begin{array}{llcll}
\mathbf{isort}(xs', ys) & \xrightarrow{1} & ys & \mid & xs' = 1 \\
\mathbf{isort}(xs', ys) & \xrightarrow{1} & \mathbf{isort}(xs, \mathbf{insert}(x, ys)) & \mid & xs' = 1{+}x{+}xs
\end{array}
$$

- $|\mathbf{insert}(x, ys)| \leq 1 + x + ys$
- $\mathrm{rt}(\mathbf{insert}(x, ys)) \leq 2 \cdot ys$

Removing Nested Function Calls

Example

$$
\begin{array}{llll}
\mathbf{isort}(xs', ys) & \xrightarrow{1} & ys & \mid \quad xs' = 1 \\
\mathbf{isort}(xs', ys) & \xrightarrow{1} & \mathbf{isort}(xs, \mathbf{insert}(x, ys)) & \mid \quad xs' = 1{+}x{+}xs
\end{array}
$$

- $|\mathbf{insert}(x, ys)| \leq 1 + x + ys$
- $\mathrm{rt}(\mathbf{insert}(x, ys)) \leq 2 \cdot ys$
- add costs of nested function call

Removing Nested Function Calls

Example

$$\begin{array}{llll} \mathbf{isort}(xs', ys) & \xrightarrow{1} & ys & \mid \quad xs' = 1 \\ \mathbf{isort}(xs', ys) & \xrightarrow{1+2\cdot ys} & \mathbf{isort}(xs, \mathbf{insert}(x, ys)) & \mid \quad xs' = 1{+}x{+}xs \end{array}$$

- $|\mathbf{insert}(x, ys)| \leq 1 + x + ys$
- $\mathrm{rt}(\mathbf{insert}(x, ys)) \leq 2 \cdot ys$
- add costs of nested function call

Removing Nested Function Calls

Example

$$
\begin{array}{lcll}
\mathbf{isort}(xs', ys) & \xrightarrow{1} & ys & \mid \quad xs' = 1 \\
\mathbf{isort}(xs', ys) & \xrightarrow{1+2\cdot ys} & \mathbf{isort}(xs, \mathbf{insert}(x, ys)) & \mid \quad xs' = 1+x+xs
\end{array}
$$

- $|\mathbf{insert}(x, ys)| \leq 1 + x + ys$
- $\mathrm{rt}(\mathbf{insert}(x, ys)) \leq 2 \cdot ys$
- add costs of nested function call
- replace nested function call by fresh variable x_f

Removing Nested Function Calls

Example

$$\begin{array}{llll} \mathbf{isort}(xs', ys) & \xrightarrow{1} & ys & \mid \quad xs' = 1 \\ \mathbf{isort}(xs', ys) & \xrightarrow{1+2\cdot ys} & \mathbf{isort}(xs, x_f) & \mid \quad xs' = 1{+}x{+}xs \end{array}$$

- $|\mathbf{insert}(x, ys)| \leq 1 + x + ys$
- $\mathrm{rt}(\mathbf{insert}(x, ys)) \leq 2 \cdot ys$
- add costs of nested function call
- replace nested function call by fresh variable x_f

Removing Nested Function Calls

Example

$$\begin{array}{llll} \mathbf{isort}(xs', ys) & \xrightarrow{1} & ys & \mid \quad xs' = 1 \\ \mathbf{isort}(xs', ys) & \xrightarrow{1+2\cdot ys} & \mathbf{isort}(xs, x_f) & \mid \quad xs' = 1+x+xs \end{array}$$

- $|\mathbf{insert}(x, ys)| \leq 1 + x + ys$
- $\mathrm{rt}(\mathbf{insert}(x, ys)) \leq 2 \cdot ys$
- add costs of nested function call
- replace nested function call by fresh variable x_f
- add constraint "$x_f \leq$ size bound"

Removing Nested Function Calls

Example

$$\begin{array}{llll} \mathbf{isort}(xs', ys) & \xrightarrow{1} & ys & \mid \quad xs' = 1 \\ \mathbf{isort}(xs', ys) & \xrightarrow{1+2\cdot ys} & \mathbf{isort}(xs, x_f) & \mid \quad xs' = 1{+}x{+}xs \wedge x_f \leq 1{+}x{+}ys \end{array}$$

- $|\mathbf{insert}(x, ys)| \leq 1 + x + ys$
- $\mathrm{rt}(\mathbf{insert}(x, ys)) \leq 2 \cdot ys$
- add costs of nested function call
- replace nested function call by fresh variable x_f
- add constraint "$x_f \leq$ size bound"

Removing Nested Function Calls

Example

$$\begin{array}{llll} \textbf{isort}(xs', ys) & \xrightarrow{1} & ys & \mid \quad xs' = 1 \\ \textbf{isort}(xs', ys) & \xrightarrow{1+2\cdot ys} & \textbf{isort}(xs, x_f) & \mid \quad xs' = 1{+}x{+}xs \wedge x_f \leq 1{+}x{+}ys \end{array}$$

- $|\textbf{insert}(x, ys)| \leq 1 + x + ys$
- $\mathrm{rt}(\textbf{insert}(x, ys)) \leq 2 \cdot ys$
- add costs of nested function call
- replace nested function call by fresh variable x_f
- add constraint "$x_f \leq$ size bound"

$\curvearrowright$ $\mathcal{O}(xs'^2 + xs' \cdot ys)$

Removing Nested Function Calls

Example

$$
\begin{array}{llll}
\mathbf{isort}(xs', ys) & \xrightarrow{1} & ys & \mid \quad xs' = 1 \\
\mathbf{isort}(xs', ys) & \xrightarrow{1+2 \cdot ys} & \mathbf{isort}(xs, x_f) & \mid \quad xs' = 1+x+xs \wedge x_f \leq 1+x+ys
\end{array}
$$

- $|\mathbf{insert}(x, ys)| \leq 1 + x + ys$
- $\mathrm{rt}(\mathbf{insert}(x, ys)) \leq 2 \cdot ys$
- add costs of nested function call
- replace nested function call by fresh variable x_f
- add constraint "$x_f \leq$ size bound"

$\curvearrowright$ $\mathcal{O}(xs'^2 + xs' \cdot ys)$

- similar techniques to eliminate *outer* function calls

Removing Nested Function Calls

Example

$$\begin{array}{llll} \mathbf{isort}(xs', ys) & \xrightarrow{1} & ys & \mid \quad xs' = 1 \\ \mathbf{isort}(xs', ys) & \xrightarrow{1+2\cdot ys} & \mathbf{isort}(xs, x_f) & \mid \quad xs' = 1+x+xs \wedge x_f \le 1+x+ys \end{array}$$

- $|\mathbf{insert}(x, ys)| \le 1 + x + ys$
- $\mathrm{rt}(\mathbf{insert}(x, ys)) \le 2 \cdot ys$
- add costs of nested function call
- replace nested function call by fresh variable x_f
- add constraint "$x_f \le$ size bound"

$\curvearrowright$ $\mathcal{O}(xs'^2 + xs' \cdot ys)$

- similar techniques to eliminate *outer* function calls

$$\mathbf{times}(\mathsf{s}(x), y) \longrightarrow \mathbf{plus}(\mathbf{times}(x, y), y)$$

Removing Nested Function Calls

Example

$$\begin{array}{llll} \textbf{isort}(xs', ys) & \xrightarrow{1} & ys & \mid \quad xs' = 1 \\ \textbf{isort}(xs', ys) & \xrightarrow{1+2\cdot ys} & \textbf{isort}(xs, x_f) & \mid \quad xs' = 1+x+xs \wedge x_f \leq 1+x+ys \end{array}$$

- $|\textbf{insert}(x, ys)| \leq 1 + x + ys$
- $\mathrm{rt}(\textbf{insert}(x, ys)) \leq 2 \cdot ys$
- add costs of nested function call
- replace nested function call by fresh variable x_f
- add constraint "$x_f \leq$ size bound"

$\curvearrowright$ $\mathcal{O}(xs'^2 + xs' \cdot ys)$

- similar techniques to eliminate *outer* function calls $\Longrightarrow$ see paper!

$$\textbf{times}(\mathsf{s}(x), y) \to \textbf{plus}(\textbf{times}(x, y), y)$$

- CoFloCo, KoAT, and PUBS used as backends

Experiments and Conclusion

- CoFloCo, KoAT, and PUBS used as backends
- CoFloCo supports nested defined symbols
  $\curvearrowright$ runs in parallel to our technique

Experiments and Conclusion

- CoFloCo, KoAT, and PUBS used as backends
- CoFloCo supports nested defined symbols
  $\curvearrowright$ runs in parallel to our technique

Results for 922 TPDB-examples

$\mathrm{rc}_{\mathcal{R}}(n)$	TcT	AProVE ITS	AProVE old	AProVE & TcT	AProVE new
$\mathcal{O}(1)$	47	**43**	48	53	**53**
$\leq \mathcal{O}(n)$	276	**254**	320	354	**379**
$\leq \mathcal{O}(n^2)$	362	**366**	425	463	**506**
$\leq \mathcal{O}(n^3)$	386	**402**	439	485	**541**
$\leq \mathcal{O}(n^{>3})$	393	**412**	439	491	**548**
$\leq$ *EXP*	393	**422**	439	491	**553**

Experiments and Conclusion

- CoFloCo, KoAT, and PUBS used as backends
- CoFloCo supports nested defined symbols
  $\curvearrowright$ runs in parallel to our technique

Results for 922 TPDB-examples

$\mathrm{rc}_{\mathcal{R}}(n)$	TcT	AProVE ITS	AProVE old	AProVE & TcT	AProVE new
$\mathcal{O}(1)$	47	**43**	48	53	**53**
$\leq \mathcal{O}(n)$	276	**254**	320	354	**379**
$\leq \mathcal{O}(n^2)$	362	**366**	425	463	**506**
$\leq \mathcal{O}(n^3)$	386	**402**	439	485	**541**
$\leq \mathcal{O}(n^{>3})$	393	**412**	439	491	**548**
$\leq$ *EXP*	393	**422**	439	491	**553**

Summary

- abstraction from terms to integers

Experiments and Conclusion

- CoFloCo, KoAT, and PUBS used as backends
- CoFloCo supports nested defined symbols
  $\curvearrowright$ runs in parallel to our technique

Results for 922 TPDB-examples

$\mathrm{rc}_{\mathcal{R}}(n)$	TcT	AProVE ITS	AProVE old	AProVE & TcT	AProVE new
$\mathcal{O}(1)$	47	**43**	48	53	**53**
$\leq \mathcal{O}(n)$	276	**254**	320	354	**379**
$\leq \mathcal{O}(n^2)$	362	**366**	425	463	**506**
$\leq \mathcal{O}(n^3)$	386	**402**	439	485	**541**
$\leq \mathcal{O}(n^{>3})$	393	**412**	439	491	**548**
$\leq$ *EXP*	393	**422**	439	491	**553**

Summary

- abstraction from terms to integers
- modular bottom-up approach using standard ITS tools

Experiments and Conclusion

- CoFloCo, KoAT, and PUBS used as backends
- CoFloCo supports nested defined symbols
  $\curvearrowright$ runs in parallel to our technique

Results for 922 TPDB-examples

$\mathrm{rc}_{\mathcal{R}}(n)$	TcT	AProVE ITS	AProVE old	AProVE & TcT	AProVE new
$\mathcal{O}(1)$	47	**43**	48	53	**53**
$\leq\ \mathcal{O}(n)$	276	**254**	320	354	**379**
$\leq\ \mathcal{O}(n^2)$	362	**366**	425	463	**506**
$\leq\ \mathcal{O}(n^3)$	386	**402**	439	485	**541**
$\leq\ \mathcal{O}(n^{>3})$	393	**412**	439	491	**548**
$\leq\ \mathit{EXP}$	393	**422**	439	491	**553**

Summary

- abstraction from terms to integers
- modular bottom-up approach using standard ITS tools
- significantly improves state-of-the-art