The Polynomial Path Order and the Rules of Predicative Recursion with Parameter Substitution

Martin Avanzini and Georg Moser

Computational Logic
Faculty of Computer Science, University of Innsbruck

WST '09

Automatic Complexity Analysis

Goal

- purely automatic complexity analysis

Approach

- employ term rewriting as model of computation
- translate termination proofs into complexity certificates

Automatic Complexity Analysis

Goal

- purely automatic complexity analysis

Approach

- employ term rewriting as model of computation
- translate termination proofs into complexity certificates
- to detect feasible computation, restrictions on termination technique usually inevitable

Outline

2. Complexity Analysis and Polynomial Path Orders

M. Avanzini and G. Moser
Complexity Analysis by Rewriting
In Proc. of FLOPS'08, LNCS vol. 4989, pp. 130–146, 2008

Outline

1. Predicative Recursion

   S. Bellantoni and S. Cook
   A new Recursion-Theoretic Characterization of the Polytime Functions
   Computation and Complexity, 2(2), pp. 97–110, 1992

2. Complexity Analysis and Polynomial Path Orders

   M. Avanzini and G. Moser
   Complexity Analysis by Rewriting
   In Proc. of FLOPS'08, LNCS vol. 4989, pp. 130–146, 2008

Outline

1. Predicative Recursion

   S. Bellantoni and S. Cook
   A new Recursion-Theoretic Characterization of the Polytime Functions
   Computation and Complexity, 2(2), pp. 97–110, 1992

2. Complexity Analysis and Polynomial Path Orders

   M. Avanzini and G. Moser
   Complexity Analysis by Rewriting
   In Proc. of FLOPS'08, LNCS vol. 4989, pp. 130–146, 2008

3. POP^* and Parameter Substitution

   M. Avanzini and G. Moser
   The Polynomial Path Order and the Rules of Predicative Recursion with Parameter Substitution
   In Proc. of WST'09

The Primitive Recursive Functions

- initial functions 0, s, Π_i^k
- composition

$$f(\vec{x}) = g(h_1(\vec{x}), \ldots, h_m(\vec{x}))$$

- primitive recursion

$$f(z+1, \vec{x}) = g(\vec{x})$$
$$f(z+1, \vec{x}) = h(z, \vec{x}, f(z, \vec{x}))$$

The Primitive Recursive Functions

- initial functions ε, s_0, s_1, Π_i^k
- composition

$$f(\vec{x}) = g(h_1(\vec{x}), \ldots, h_m(\vec{x}))$$

- primitive recursion on notation

$$f(\varepsilon, \vec{x}) = g(\vec{x})$$
$$f(z{\cdot}i, \vec{x}) = h_i(z, \vec{x}, f(z, \vec{x})), \quad i \in \{0, 1\}$$

The Primitive Recursive Functions

- initial functions ε, s_0, s_1, Π_i^k
- composition

$$f(\vec{x}) = g(h_1(\vec{x}), \dots, h_m(\vec{x}))$$

- primitive recursion on notation

$$\begin{aligned} f(\varepsilon, \vec{x}) &= g(\vec{x}) \\ f(z{\cdot}i, \vec{x}) &= h_i(z, \vec{x}, f(z, \vec{x})),\ i \in \{0,1\} \end{aligned}$$

$$\begin{aligned} dup(\varepsilon) &= \varepsilon \\ dup(x{\cdot}i) &= s_i(s_i(dup(x)) \end{aligned} \qquad \begin{aligned} exp(\varepsilon) &= s_1(\varepsilon) \\ exp(x{\cdot}i) &= dup(exp(x)) \end{aligned}$$

Bellantoni and Cook's Definition of $\mathcal{BC}$

- initial functions ε, s_0, s_1, Π^k_i, ...
- composition

$$f(\vec{x}) = g(h_1(\vec{x}), \ldots, h_m(\vec{x}))$$

- predicative recursion on notation

$$f(\varepsilon, \vec{x}; \vec{y}) = g(\vec{x}; \vec{y})$$
$$f(z{\cdot}i, \vec{x}; \vec{y}) = h_i(z, \vec{x}; \vec{y}, f(z, \vec{x}; \vec{y})), \; i \in \{0, 1\}$$

$$f(\underbrace{x_1, \ldots, x_m}_{\text{normal}}; \underbrace{y_1, \ldots, y_n}_{\text{safe}})$$

Bellantoni and Cook's Definition of $\mathcal{BC}$

- initial functions $\varepsilon,\ s_0,\ s_1,\ \Pi^k_i,\ \dots$
- predicative composition

$$f(\vec{x};\vec{y}) = g(n_1(\vec{x};),\dots,n_j(\vec{x};);\ s_1(\vec{x};\vec{y}),\dots,s_k(\vec{x};\vec{y}))$$

- predicative recursion on notation

$$\begin{aligned} f(\varepsilon,\vec{x};\vec{y}) &= g(\vec{x};\vec{y}) \\ f(z{\cdot}i,\vec{x};\vec{y}) &= h_i(z,\vec{x};\vec{y},f(z,\vec{x};\vec{y})),\ i \in \{0,1\} \end{aligned}$$

$$f(\underbrace{x_1,\dots,x_m}_{\text{normal}};\underbrace{y_1,\dots,y_n}_{\text{safe}})$$

Bellantoni and Cook's Definition of $\mathcal{BC}$

- initial functions ε, s_0, s_1, Π_i^k, ...
- predicative composition

$$f(\vec{x};\vec{y}) = g(n_1(\vec{x};),\ldots,n_j(\vec{x};);\ s_1(\vec{x};\vec{y}),\ldots,s_k(\vec{x};\vec{y}))$$

- predicative recursion on notation

$$f(\varepsilon,\vec{x};\vec{y}) = g(\vec{x};\vec{y})$$
$$f(z{\cdot}i,\vec{x};\vec{y}) = h_i(z,\vec{x};\vec{y},f(z,\vec{x};\vec{y})),\ i\in\{0,1\}$$

Theorem

$$\mathcal{BC} = \mathbf{FP}$$

polytime computable functions

Bellantoni and Cook's Definition of $\mathcal{BC}$

- initial functions ε, s_0, s_1, Π^k_i, ...
- predicative composition

$$f(\vec{x};\vec{y}) = g(n_1(\vec{x};), \dots, n_j(\vec{x};);\ s_1(\vec{x};\vec{y}), \dots, s_k(\vec{x};\vec{y}))$$

- predicative recursion on notation

$$f(\varepsilon, \vec{x};\vec{y}) = g(\vec{x};\vec{y})$$
$$f(z{\cdot}i, \vec{x};\vec{y}) = h_i(z, \vec{x};\vec{y}, f(z, \vec{x};\vec{y})),\ i \in \{0,1\}$$

$$dup(\varepsilon) = \varepsilon \qquad exp(\varepsilon) = s_1(\varepsilon)$$
$$dup(x{\cdot}i) = s_i(s_i(dup(x))) \qquad exp(x{\cdot}i) = dup(exp(x))$$

Bellantoni and Cook's Definition of $\mathcal{BC}$

- initial functions $\varepsilon,\ s_0,\ s_1,\ \Pi^k_i,\ \ldots$
- predicative composition

$$f(\vec{x};\vec{y}) = g(n_1(\vec{x};),\ldots,n_j(\vec{x};);\ s_1(\vec{x};\vec{y}),\ldots,s_k(\vec{x};\vec{y}))$$

- predicative recursion on notation

$$f(\varepsilon,\vec{x};\vec{y}) = g(\vec{x};\vec{y})$$
$$f(z\cdot i,\vec{x};\vec{y}) = h_i(z,\vec{x};\vec{y},f(z,\vec{x};\vec{y})),\ i \in \{0,1\}$$

$$dup(\varepsilon;) = \varepsilon \qquad exp(\varepsilon) = s_1(;\varepsilon)$$
$$dup(x\cdot i;) = s_i(;s_i(;dup(x;))) \qquad exp(x\cdot i) = dup(exp(x);)$$

Bellantoni and Cook's Definition of $\mathcal{BC}$

- initial functions ε, s_0, s_1, Π_i^k, ...
- predicative composition

$$f(\vec{x};\vec{y}) = g(n_1(\vec{x};), \ldots, n_j(\vec{x};);\ s_1(\vec{x};\vec{y}), \ldots, s_k(\vec{x};\vec{y}))$$

- predicative recursion on notation

$$\begin{aligned} f(\varepsilon, \vec{x}; \vec{y}) &= g(\vec{x};\vec{y}) \\ f(z{\cdot}i, \vec{x}; \vec{y}) &= h_i(z, \vec{x}; \vec{y}, f(z, \vec{x}; \vec{y})),\ i \in \{0,1\} \end{aligned}$$

$$\begin{aligned} dup(\varepsilon;) &= \varepsilon \\ dup(x{\cdot}i;) &= s_i(;s_i(;dup(x;))) \end{aligned}$$

APPROVED

$$\begin{aligned} exp(\varepsilon) &= s_1(;\varepsilon) \\ exp(x{\cdot}i) &= dup(exp(x);) \end{aligned}$$

FAILED

Automatic Complexity Analysis by Rewriting

$$dup(\varepsilon) = \varepsilon \qquad exp(\varepsilon) = s_1(\varepsilon)$$

$$dup(x \cdot i) = s_i(s_i(dup(x))) \qquad exp(x \cdot i) = dup(exp(x))$$

$$\Downarrow$$

$$\mathsf{dup}(\epsilon) \to \epsilon \qquad \mathsf{exp}(\epsilon) \to \mathsf{s}_1(\epsilon)$$

$$\mathsf{dup}(\mathsf{s}_i(x)) \to \mathsf{s}_i(\mathsf{s}_i(\mathsf{dup}(x))) \qquad \mathsf{exp}(\mathsf{s}_i(x)) \to \mathsf{dup}(\mathsf{exp}(x))$$

Automatic Complexity Analysis by Rewriting

$$dup(\varepsilon) = \varepsilon \qquad exp(\varepsilon) = s_1(\varepsilon)$$
$$dup(x{\cdot}i) = s_i(s_i(dup(x))) \qquad exp(x{\cdot}i) = dup(exp(x))$$

$$\Downarrow$$

$$\mathsf{dup}(\epsilon) \to \epsilon \qquad \mathsf{exp}(\epsilon) \to \mathsf{s}_1(\epsilon)$$
$$\mathsf{dup}(\mathsf{s}_i(x)) \to \mathsf{s}_i(\mathsf{s}_i(\mathsf{dup}(x))) \qquad \mathsf{exp}(\mathsf{s}_i(x)) \to \mathsf{dup}(\mathsf{exp}(x))$$

$$f(w_1, \ldots, w_n) = v \iff \mathsf{f}(\ulcorner w_1 \urcorner, \ldots, \ulcorner w_n \urcorner) \to^{!}_{\mathcal{R}} \ulcorner v \urcorner$$

computation

Runtime Complexity of TRSs

- derivation length

$$\mathsf{dl}(t, \to) = \max\{n \mid \exists s.\ t \to^n s\}$$

$$\mathsf{dl}(n, T, \to) = \max\{\mathsf{dl}(t, \to) \mid t \in T \text{ and } |t| \leqslant n\}$$

Runtime Complexity of TRSs

- derivation length

$$\mathsf{dl}(t, \rightarrow) = \max\{n \mid \exists s.\ t \rightarrow^n s\}$$

$$\mathsf{dl}(n, T, \rightarrow) = \max\{\mathsf{dl}(t, \rightarrow) \mid t \in T \text{ and } |t| \leqslant n\}$$

- derivational complexity

$$\mathsf{dc}_{\mathcal{R}}(n) = \mathsf{dl}(n, \mathcal{T}(\mathcal{F}, \mathcal{V}), \rightarrow_{\mathcal{R}})$$

- runtime complexity

$$\mathsf{rc}_{\mathcal{R}}(n) = \mathsf{dl}(n, \mathcal{T}_{\mathsf{b}}, \rightarrow_{\mathcal{R}})$$

$$\mathcal{T}_{\mathsf{b}} := \{f(c_1, \dots, c_n) \mid f \in \mathcal{D} \text{ and } c_i \in \mathcal{T}(\mathcal{C}, \mathcal{V})\}$$

Runtime Complexity of TRSs

- derivation length

$$\mathsf{dl}(t, \to) = \max\{n \mid \exists s.\ t \to^n s\}$$

$$\mathsf{dl}(n, T, \to) = \max\{\mathsf{dl}(t, \to) \mid t \in T \text{ and } |t| \leqslant n\}$$

- derivational complexity

$$\mathsf{dc}_{\mathcal{R}}(n) = \mathsf{dl}(n, \mathcal{T}(\mathcal{F}, \mathcal{V}), \to_{\mathcal{R}})$$

- innermost runtime complexity

$$\mathsf{rc}^{\mathsf{i}}_{\mathcal{R}}(n) = \mathsf{dl}(n, \mathcal{T}_{\mathsf{b}}, \xrightarrow{\mathsf{i}}_{\mathcal{R}})$$

$$\mathcal{T}_{\mathsf{b}} := \{f(c_1, \dots, c_n) \mid f \in \mathcal{D} \text{ and } c_i \in \mathcal{T}(\mathcal{C}, \mathcal{V})\}$$

Polynomial Path Orders $>_{\mathsf{pop*}}$

- $>_{\mathsf{pop*}} \subsetneq >_{\mathsf{mpo}}$
- $>_{\mathsf{pop*}}$ induced by precedence $\succsim$ and safe mapping safe
- $>_{\mathsf{pop*}} \approx >_{\mathsf{mpo}} +$ predicative recursion

Theorem

$$\mathcal{R} \subseteq >_{\mathsf{pop*}} \Rightarrow \mathsf{rc}^{\mathsf{i}}_{\mathcal{R}} \textit{ polynomially bounded}$$

for constructor TRS $\mathcal{R}$

Polynomial Path Orders $>_{\mathsf{pop}*}$

- $>_{\mathsf{pop}*} \subseteq >_{\mathsf{mpo}}$
- $>_{\mathsf{pop}*}$ induced by precedence $\succsim$ and safe mapping safe
- $>_{\mathsf{pop}*} \approx >_{\mathsf{mpo}} +$ predicative recursion

Theorem

$$\mathcal{R} \subseteq >_{\mathsf{pop}*} \Rightarrow \mathsf{rc}^{\mathsf{i}}_{\mathcal{R}} \textit{ polynomially bounded}$$

for constructor TRS $\mathcal{R}$

$$\mathsf{rev}(xs;\,) \to \mathsf{rev}_{\mathsf{tl}}(xs;\, [\,])$$

$$\mathsf{rev}_{\mathsf{tl}}([\,];\, ys) \to ys$$

$$\mathsf{rev}_{\mathsf{tl}}(x{:}xs;\, ys) \to \mathsf{rev}_{\mathsf{tl}}(xs;\, x{:}ys)$$

Polynomial Path Orders $>_{\mathsf{pop}*}$

- $>_{\mathsf{pop}*} \subseteq >_{\mathsf{mpo}}$
- $>_{\mathsf{pop}*}$ induced by precedence $\gtrsim$ and safe mapping safe
- $>_{\mathsf{pop}*} \approx >_{\mathsf{mpo}}$ + predicative recursion

Theorem

$$\mathcal{R} \subseteq >_{\mathsf{pop}*} \Rightarrow \mathsf{rc}^{\mathsf{i}}_{\mathcal{R}} \text{ polynomially bounded}$$

for constructor TRS $\mathcal{R}$

$$\mathsf{rev}(xs;) \to \mathsf{rev}_{\mathsf{tl}}(xs; [\,]) \qquad \mathsf{rev}_{\mathsf{tl}}([\,]; ys) \to ys$$

$$\mathsf{rev}_{\mathsf{tl}}(x{:}xs; ys) \to \mathsf{rev}_{\mathsf{tl}}(xs; x{:}ys)$$

$$\mathsf{rev}_{\mathsf{tl}}(x{:}xs; ys) \not>_{\mathsf{mpo}} \mathsf{rev}_{\mathsf{tl}}(xs; x{:}ys) \qquad ✗$$

Polynomial Path Orders $>_{\mathsf{pop}*}$

- $>_{\mathsf{pop}*} \subsetneq >_{\mathsf{mpo}}$
- $>_{\mathsf{pop}*}$ induced by precedence $\succsim$ and safe mapping safe
- $>_{\mathsf{pop}*} \approx >_{\mathsf{mpo}} +$ predicative recursion

Theorem

$$\mathcal{R} \subseteq >_{\mathsf{pop}*} \Rightarrow \mathsf{rc}^{\mathsf{i}}_{\mathcal{R}} \text{ polynomially bounded}$$

for constructor TRS $\mathcal{R}$

$$\mathsf{rev}(xs;) \to \mathsf{rev}_{\mathsf{tl}}(xs; [\,]) \qquad \mathsf{rev}_{\mathsf{tl}}([\,]; ys) \to ys$$

$$\mathsf{rev}_{\mathsf{tl}}(x{:}xs; ys) \to \mathsf{rev}_{\mathsf{tl}}(xs; x{:}ys)$$

$$\mathcal{R}_{\mathsf{rev}} \not\subseteq >_{\mathsf{pop}*}$$

Predicative Recursion with Parameter Substitution

$$f(\varepsilon, \vec{x}; \vec{y}) = g(\vec{x}; \vec{y})$$
$$f(z{\cdot}i, \vec{x}; \vec{y}) = h_i(z, \vec{x}; \vec{y}, f(z, \vec{x}; \vec{y})),\ \ i \in \{0, 1\}$$

$$\Downarrow$$

$$f(\varepsilon, \vec{x}; \vec{y}) = g(\vec{x}; \vec{y})$$
$$f(z{\cdot}i, \vec{x}; \vec{y}) = h_i(z, \vec{x}; \vec{y}, f(z, \vec{x}; \vec{p}(z, \vec{x}; \vec{y}))),\ \ i \in \{0, 1\}$$

predicative recursion with parameter substitution

Theorem (Bellantoni,1993)

$\mathcal{BC}$ is closed under predicative recursion with parameter substitution scheme.

Polynomial Path Order with Parameter Substitution $>_{\mathsf{pps*}}$

$s = f(s_1, \dots, s_o; s_{o+1}, \dots, s_m) >_{\mathsf{pps*}} t$ if

1. $s_i \gtrsim_{\mathsf{pps*}} t$ for some $i \in \{1, \dots, m\}$
2. $t = g(t_1, \dots, t_p; t_{p+1}, \dots, t_n)$, $f \succ g$, $f \in \mathcal{D}$
   - $s >_{\mathsf{pps}} t_j$ for all $j \in \{1, \dots, p\}$, $s >_{\mathsf{pps*}} t_j$ for all $j \in \{p+1, \dots, n\}$
   - $t_j \in \mathcal{T}(\mathcal{F}^{\prec f}, \mathcal{V})$ for all but one $j \in \{p+1, \dots, n\}$
3. $t = g(t_1, \dots, t_p; t_{p+1}, \dots, t_n)$, $f \approx g$, $f \in \mathcal{D}$
   - $\{\!\{s_1, \dots, s_o\}\!\} >^{\mathsf{mul}}_{\mathsf{pps*}} \{\!\{t_1, \dots, t_p\}\!\}$
   - for all $j \in \{p+1, \dots, n\}$, $s >_{\mathsf{pps*}} t_j$ and $t_j \in \mathcal{T}(\mathcal{F}^{\prec f}, \mathcal{V})$

RPOs $\rightsquigarrow$ $\mathrm{POP}^*_{\mathsf{ps}}$

$s = f(s_1, \dots, s_o; s_{o+1}, \dots, s_m) >_{\mathsf{pps}*} t$ if

1. $s_i \gtrsim_{\mathsf{pps}*} t$ for some $i \in \{1, \dots, m\}$
2. $t = g(t_1, \dots, t_p; t_{p+1}, \dots, t_n)$, $f \succ g$
   - $s >_{\mathsf{pps}*} t_j$ for all $j \in \{1, \dots, n\}$
3. $t = g(t_1, \dots, t_p; t_{p+1}, \dots, t_n)$, $f \approx g$
   - $\{\!\{s_1, \dots, s_o\}\!\} >^{\mathsf{mul}}_{\mathsf{pps}*} \{\!\{t_1, \dots, t_p\}\!\}$
   - for all $j \in \{p+1, \dots, n\}$, $s >_{\mathsf{pps}*} t_j$

RPOs $\rightsquigarrow$ $\mathrm{POP}^*_{\mathsf{ps}}$

$s = f(s_1, \ldots, s_o; s_{o+1}, \ldots, s_m) >_{\mathsf{pps}*} t$ if

1. $s_i \gtrsim_{\mathsf{pps}*} t$ for some $i \in \{1, \ldots, m\}$
2. $t = g(t_1, \ldots, t_p; t_{p+1}, \ldots, t_n)$, $f \succ g$
   - $s >_{\mathsf{pps}*} t_j$ for all $j \in \{1, \ldots, n\}$
3. $t = g(t_1, \ldots, t_p; t_{p+1}, \ldots, t_n)$, $f \approx g$
   - $\{\!\{s_1, \ldots, s_o\}\!\} >^{\mathsf{mul}}_{\mathsf{pps}*} \{\!\{t_1, \ldots, t_p\}\!\}$
   - for all $j \in \{p+1, \ldots, n\}$, $s >_{\mathsf{pps}*} t_j$

$\mathsf{f}(\mathsf{s}(;x); y) >_{\mathsf{pps}*} \mathsf{f}(x; \mathsf{f}(x; y))$ ✗

RPOs $\rightsquigarrow$ $\mathrm{POP}^*_{\mathsf{ps}}$

$s = f(s_1, \dots, s_o; s_{o+1}, \dots, s_m) >_{\mathsf{pps}*} t$ if

1. $s_i \gtrsim_{\mathsf{pps}*} t$ for some $i \in \{1, \dots, m\}$
2. $t = g(t_1, \dots, t_p; t_{p+1}, \dots, t_n)$, $f \succ g$
   - $s >_{\mathsf{pps}*} t_j$ for all $j \in \{1, \dots, n\}$
3. $t = g(t_1, \dots, t_p; t_{p+1}, \dots, t_n)$, $f \approx g$
   - $\{\{s_1, \dots, s_o\}\} >^{\mathsf{mul}}_{\mathsf{pps}*} \{\{t_1, \dots, t_p\}\}$
   - for all $j \in \{p+1, \dots, n\}$, $s >_{\mathsf{pps}*} t_j$ and $t_j \in \mathcal{T}(\mathcal{F}^{\prec f}, \mathcal{V})$

$$\mathsf{f}(\mathsf{s}(;x); y) \not>_{\mathsf{pps}*} \mathsf{f}(x; \mathsf{f}(x; y)) \quad \checkmark$$

RPOs $\rightsquigarrow$ $\mathrm{POP}^*_{\mathsf{ps}}$

$s = f(s_1, \ldots, s_o; s_{o+1}, \ldots, s_m) >_{\mathsf{pps}*} t$ if

1. $s_i \gtrsim_{\mathsf{pps}*} t$ for some $i \in \{1, \ldots, m\}$
2. $t = g(t_1, \ldots, t_p; t_{p+1}, \ldots, t_n)$, $f \succ g$
   - $s >_{\mathsf{pps}*} t_j$ for all $j \in \{1, \ldots, n\}$
3. $t = g(t_1, \ldots, t_p; t_{p+1}, \ldots, t_n)$, $f \approx g$
   - $\{\!\{s_1, \ldots, s_o\}\!\} >^{\mathsf{mul}}_{\mathsf{pps}*} \{\!\{t_1, \ldots, t_p\}\!\}$
   - for all $j \in \{p+1, \ldots, n\}$, $s >_{\mathsf{pps}*} t_j$ and $t_j \in \mathcal{T}(\mathcal{F}^{\prec f}, \mathcal{V})$

$$\mathsf{f}(\mathsf{s}(;x); y) >_{\mathsf{pps}*} \mathsf{g}(\mathsf{f}(x; y), \mathsf{f}(x; y))$$ ✗

RPOs $\rightsquigarrow$ $\mathrm{POP}^*_{\mathsf{ps}}$

$s = f(s_1, \dots, s_o; s_{o+1}, \dots, s_m) >_{\mathsf{pps}*} t$ if

1. $s_i \gtrsim_{\mathsf{pps}*} t$ for some $i \in \{1, \dots, m\}$
2. $t = g(t_1, \dots, t_p; t_{p+1}, \dots, t_n)$, $f \succ g$
   - $s >_{\mathsf{pps}*} t_j$ for all $j \in \{1, \dots, n\}$
   - $t_j \in \mathcal{T}(\mathcal{F}^{\prec f}, \mathcal{V})$ for all but one $j \in \{p+1, \dots, n\}$
3. $t = g(t_1, \dots, t_p; t_{p+1}, \dots, t_n)$, $f \approx g$
   - $\{\!\{s_1, \dots, s_o\}\!\} >^{\mathsf{mul}}_{\mathsf{pps}*} \{\!\{t_1, \dots, t_p\}\!\}$
   - for all $j \in \{p+1, \dots, n\}$, $s >_{\mathsf{pps}*} t_j$ and $t_j \in \mathcal{T}(\mathcal{F}^{\prec f}, \mathcal{V})$

$$\mathsf{f}(\mathsf{s}(;x); y) >_{\mathsf{pps}*} \mathsf{g}(\mathsf{f}(x; y), \mathsf{f}(x; y)) \qquad \checkmark$$

RPOs $\rightsquigarrow$ $\mathrm{POP}^*_{\mathsf{ps}}$

$s = f(s_1, \dots, s_o; s_{o+1}, \dots, s_m) >_{\mathsf{pps}*} t$ if

1. $s_i \gtrsim_{\mathsf{pps}*} t$ for some $i \in \{1, \dots, m\}$
2. $t = g(t_1, \dots, t_p; t_{p+1}, \dots, t_n)$, $f \succ g$
   - $s >_{\mathsf{pps}*} t_j$ for all $j \in \{1, \dots, n\}$
   - $t_j \in \mathcal{T}(\mathcal{F}^{\prec f}, \mathcal{V})$ for all but one $j \in \{p+1, \dots, n\}$
3. $t = g(t_1, \dots, t_p; t_{p+1}, \dots, t_n)$, $f \approx g$
   - $\{\{s_1, \dots, s_o\}\} >^{\mathsf{mul}}_{\mathsf{pps}*} \{\{t_1, \dots, t_p\}\}$
   - for all $j \in \{p+1, \dots, n\}$, $s >_{\mathsf{pps}*} t_j$ and $t_j \in \mathcal{T}(\mathcal{F}^{\prec f}, \mathcal{V})$

$$f(\vec{x}; \vec{y}) = g(n_1(\vec{x};), \dots, n_j(\vec{x};); s_1(\vec{x}; \vec{y}), \dots, s_k(\vec{x}; \vec{y}))$$

RPOs $\rightsquigarrow$ $\mathrm{POP}^*_{\mathsf{ps}}$

$s = f(s_1, \ldots, s_o; s_{o+1}, \ldots, s_m) >_{\mathsf{pps}*} t$ if

1. $s_i \gtrsim_{\mathsf{pps}*} t$ for some $i \in \{1, \ldots, m\}$
2. $t = g(t_1, \ldots, t_p; t_{p+1}, \ldots, t_n)$, $f \succ g$
   - $s >_{\mathsf{pps}*} t_j$ for all $j \in \{1, \ldots, n\}$
   - $t_j \in \mathcal{T}(\mathcal{F}^{\prec f}, \mathcal{V})$ for all but one $j \in \{p+1, \ldots, n\}$
3. $t = g(t_1, \ldots, t_p; t_{p+1}, \ldots, t_n)$, $f \approx g$
   - $\{\!\{s_1, \ldots, s_o\}\!\} >^{\mathsf{mul}}_{\mathsf{pps}*} \{\!\{t_1, \ldots, t_p\}\!\}$
   - for all $j \in \{p+1, \ldots, n\}$, $s >_{\mathsf{pps}*} t_j$ and $t_j \in \mathcal{T}(\mathcal{F}^{\prec f}, \mathcal{V})$

$$f(\vec{x}; \vec{y}) >_{\mathsf{pps}*} g(n(\vec{x}; \vec{y}); s(\vec{x}; \vec{y})) \quad ✗$$

Auxiliary Order $>_{\mathsf{pps}}$

$s = f(s_1, \dots, s_o; s_{o+1}, \dots, s_m) >_{\mathsf{pps}} t$ if

1. $s_i \gtrsim_{\mathsf{pps}} t$ for some $i \in \{1, \dots, m\}$ and
   - if $f \in \mathcal{D}$ then $i \in \{1, \dots, o\}$
2. $t = g(t_1, \dots, t_p; t_{p+1}, \dots, t_n)$, $f \succ g$, $f \in \mathcal{D}$
   - $s >_{\mathsf{pps}} t_j$ for all $j \in \{1, \dots, n\}$

Auxiliary Order $>_{\mathsf{pps}}$

$s = f(s_1, \dots, s_o; s_{o+1}, \dots, s_m) >_{\mathsf{pps}} t$ if

1. $s_i \gtrsim_{\mathsf{pps}} t$ for some $i \in \{1, \dots, m\}$ and
   - if $f \in \mathcal{D}$ then $i \in \{1, \dots, o\}$
2. $t = g(t_1, \dots, t_p; t_{p+1}, \dots, t_n)$, $f \succ g$, $f \in \mathcal{D}$
   - $s >_{\mathsf{pps}} t_j$ for all $j \in \{1, \dots, n\}$

$$\mathsf{f}(x; y) \not>_{\mathsf{pps}} \mathsf{g}(x; y)$$

Auxiliary Order $>_{\mathsf{pps}}$

$s = f(s_1, \ldots, s_o; s_{o+1}, \ldots, s_m) >_{\mathsf{pps}} t$ if

1. $s_i \gtrsim_{\mathsf{pps}} t$ for some $i \in \{1, \ldots, m\}$ and
   - if $f \in \mathcal{D}$ then $i \in \{1, \ldots, o\}$
2. $t = g(t_1, \ldots, t_p; t_{p+1}, \ldots, t_n)$, $f \succ g$, $f \in \mathcal{D}$
   - $s >_{\mathsf{pps}} t_j$ for all $j \in \{1, \ldots, n\}$

$$\mathsf{f}(x; y) \not>_{\mathsf{pps}} y$$

Auxiliary Order $>_{\mathsf{pps}}$

$s = f(s_1, \dots, s_o; s_{o+1}, \dots, s_m) >_{\mathsf{pps}} t$ if

1. $s_i \gtrsim_{\mathsf{pps}} t$ for some $i \in \{1, \dots, m\}$ and
   - if $f \in \mathcal{D}$ then $i \in \{1, \dots, o\}$
2. $t = g(t_1, \dots, t_p; t_{p+1}, \dots, t_n)$, $f \succ g$, $f \in \mathcal{D}$
   - $s >_{\mathsf{pps}} t_j$ for all $j \in \{1, \dots, n\}$

$$\mathsf{f}(x; y) >_{\mathsf{pps}} x$$

Auxiliary Order $>_{\mathsf{pps}}$

$s = f(s_1, \dots, s_o; s_{o+1}, \dots, s_m) >_{\mathsf{pps}} t$ if

1. $s_i \gtrsim_{\mathsf{pps}} t$ for some $i \in \{1, \dots, m\}$ and
   - if $f \in \mathcal{D}$ then $i \in \{1, \dots, o\}$
2. $t = g(t_1, \dots, t_p; t_{p+1}, \dots, t_n)$, $f \succ g$, $f \in \mathcal{D}$
   - $s >_{\mathsf{pps}} t_j$ for all $j \in \{1, \dots, n\}$

$$\mathsf{f}(x; y) \ >_{\mathsf{pps}} \mathsf{g}(x;)$$

RPOs $\rightsquigarrow$ $\mathrm{POP}^*_{\mathsf{ps}}$

$s = f(s_1, \ldots, s_o; s_{o+1}, \ldots, s_m) >_{\mathsf{pps}*} t$ if

1. $s_i \gtrsim_{\mathsf{pps}*} t$ for some $i \in \{1, \ldots, m\}$
2. $t = g(t_1, \ldots, t_p; t_{p+1}, \ldots, t_n)$, $f \succ g$
   - $s >_{\mathsf{pps}*} t_j$ for all $j \in \{1, \ldots, n\}$
   - $t_j \in \mathcal{T}(\mathcal{F}^{\prec f}, \mathcal{V})$ for all but one $j \in \{p+1, \ldots, n\}$
3. $t = g(t_1, \ldots, t_p; t_{p+1}, \ldots, t_n)$, $f \approx g$
   - $\{\!\{s_1, \ldots, s_o\}\!\} >^{\mathsf{mul}}_{\mathsf{pps}*} \{\!\{t_1, \ldots, t_p\}\!\}$
   - for all $j \in \{p+1, \ldots, n\}$, $s >_{\mathsf{pps}*} t_j$ and $t_j \in \mathcal{T}(\mathcal{F}^{\prec f}, \mathcal{V})$

$$f(\vec{x}; \vec{y}) >_{\mathsf{pps}*} g(n(\vec{x}; \vec{y}); s(\vec{x}; \vec{y})) \quad ✗$$

RPOs $\rightsquigarrow$ $\mathrm{POP}^*_{\mathsf{ps}}$

$s = f(s_1, \ldots, s_o; s_{o+1}, \ldots, s_m) >_{\mathsf{pps}*} t$ if

1. $s_i \gtrsim_{\mathsf{pps}*} t$ for some $i \in \{1, \ldots, m\}$
2. $t = g(t_1, \ldots, t_p; t_{p+1}, \ldots, t_n)$, $f \succ g$
   - $s >_{\mathsf{pps}} t_j$ for all $j \in \{1, \ldots, p\}$, $s >_{\mathsf{pps}*} t_j$ for all $j \in \{p+1, \ldots, n\}$
   - $t_j \in \mathcal{T}(\mathcal{F}^{\prec f}, \mathcal{V})$ for all but one $j \in \{p+1, \ldots, n\}$
3. $t = g(t_1, \ldots, t_p; t_{p+1}, \ldots, t_n)$, $f \approx g$
   - $\{\!\{s_1, \ldots, s_o\}\!\} >^{\mathsf{mul}}_{\mathsf{pps}*} \{\!\{t_1, \ldots, t_p\}\!\}$
   - for all $j \in \{p+1, \ldots, n\}$, $s >_{\mathsf{pps}*} t_j$ and $t_j \in \mathcal{T}(\mathcal{F}^{\prec f}, \mathcal{V})$

$$f(\vec{x}; \vec{y}) \not>_{\mathsf{pps}*} g(n(\vec{x}; \vec{y}); s(\vec{x}; \vec{y})) \qquad \checkmark$$

RPOs $\rightsquigarrow$ $\mathrm{POP}^*_{\mathsf{ps}}$

$s = f(s_1, \ldots, s_o; s_{o+1}, \ldots, s_m) >_{\mathsf{pps}*} t$ if

1. $s_i \gtrsim_{\mathsf{pps}*} t$ for some $i \in \{1, \ldots, m\}$
2. $t = g(t_1, \ldots, t_p; t_{p+1}, \ldots, t_n)$, $f \succ g$
   - $s >_{\mathsf{pps}} t_j$ for all $j \in \{1, \ldots, p\}$, $s >_{\mathsf{pps}*} t_j$ for all $j \in \{p+1, \ldots, n\}$
   - $t_j \in \mathcal{T}(\mathcal{F}^{\prec f}, \mathcal{V})$ for all but one $j \in \{p+1, \ldots, n\}$
3. $t = g(t_1, \ldots, t_p; t_{p+1}, \ldots, t_n)$, $f \approx g$
   - $\{\!\{s_1, \ldots, s_o\}\!\} >^{\mathsf{mul}}_{\mathsf{pps}*} \{\!\{t_1, \ldots, t_p\}\!\}$
   - for all $j \in \{p+1, \ldots, n\}$, $s >_{\mathsf{pps}*} t_j$ and $t_j \in \mathcal{T}(\mathcal{F}^{\prec f}, \mathcal{V})$

$$f(\vec{x}; \vec{y}) >_{\mathsf{pps}*} g(n(\vec{x};); s(\vec{x}; \vec{y})) \quad \checkmark$$

Polynomial Path Order with Parameter Substitution $>_{\mathsf{pps}*}$

$s = f(s_1, \dots, s_o; s_{o+1}, \dots, s_m) >_{\mathsf{pps}*} t$ if

1. $s_i \gtrsim_{\mathsf{pps}*} t$ for some $i \in \{1, \dots, m\}$
2. $t = g(t_1, \dots, t_p; t_{p+1}, \dots, t_n)$, $f \succ g$, $f \in \mathcal{D}$
   - $s >_{\mathsf{pps}} t_j$ for all $j \in \{1, \dots, p\}$, $s >_{\mathsf{pps}*} t_j$ for all $j \in \{p+1, \dots, n\}$
   - $t_j \in \mathcal{T}(\mathcal{F}^{\prec f}, \mathcal{V})$ for all but one $j \in \{p+1, \dots, n\}$
3. $t = g(t_1, \dots, t_p; t_{p+1}, \dots, t_n)$, $f \approx g$, $f \in \mathcal{D}$
   - $\{\!\{s_1, \dots, s_o\}\!\} >^{\mathsf{mul}}_{\mathsf{pps}*} \{\!\{t_1, \dots, t_p\}\!\}$
   - for all $j \in \{p+1, \dots, n\}$, $s >_{\mathsf{pps}*} t_j$ and $t_j \in \mathcal{T}(\mathcal{F}^{\prec f}, \mathcal{V})$

Polynomial Path Order with Parameter Substitution $>_{\mathsf{pps*}}$

Theorem

$$\mathcal{R} \subseteq >_{\mathsf{pps*}} \Rightarrow \mathsf{rc}^{\mathsf{i}}_{\mathcal{R}} \textit{ polynomially bounded}$$

for constructor TRS $\mathcal{R}$

Polynomial Path Order with Parameter Substitution $>_{\mathsf{pps*}}$

Theorem

$$\mathcal{R} \subseteq >_{\mathsf{pps*}} \Rightarrow \mathsf{rc}^{\mathsf{i}}_{\mathcal{R}} \textit{ polynomially bounded}$$

for constructor TRS $\mathcal{R}$

Observation

$$>_{\mathsf{pop*}} \subsetneq >_{\mathsf{pps*}} \not\subseteq >_{\mathsf{mpo}}$$

$$\mathsf{rev}(xs;\,) \rightarrow \mathsf{rev_{tl}}(xs;\,[\,]) \qquad \mathsf{rev_{tl}}([\,];\,ys) \rightarrow ys$$

$$\mathsf{rev_{tl}}(x{:}xs;\,ys) \rightarrow \mathsf{rev_{tl}}(xs;\,x{:}ys)$$

Polynomial Path Order with Parameter Substitution $>_{\mathsf{pps}*}$

Theorem

$$\mathcal{R} \subseteq >_{\mathsf{pps}*} \Rightarrow \mathsf{rc}^{\mathsf{i}}_{\mathcal{R}} \text{ polynomially bounded}$$

for constructor TRS $\mathcal{R}$

Observation

$$>_{\mathsf{pop}*} \subsetneq >_{\mathsf{pps}*} \not\subseteq >_{\mathsf{mpo}}$$

$\mathsf{rev}(xs;\,) >_{\mathsf{pps}*} \mathsf{rev}_{\mathsf{tl}}(xs;\,[\,])$

$\mathsf{rev}_{\mathsf{tl}}([\,];\,ys) >_{\mathsf{pps}*} ys$

$\mathsf{rev}_{\mathsf{tl}}(x{:}xs;\,ys) >_{\mathsf{pps}*} \mathsf{rev}_{\mathsf{tl}}(xs;\,x{:}ys)$

Polytime Computability and $>_{\mathsf{pps}*}$

Theorem

If $\mathcal{R} \subseteq >_{\mathsf{pps}*}$ then the functions computed by $\mathcal{R}$ are polytime computable.

Polytime Computability and $>_{\mathsf{pps}*}$

terms grow only
polynomial in size

Theorem

- *Let $\mathcal{R}$ be an S-sorted constructor TRS based on a simple signature. If $\mathcal{R} \subseteq >_{\mathsf{pps}*}$ then the functions computed by $\mathcal{R}$ are polytime computable.*

Polytime Computability and $>_{\mathsf{pps}*}$

terms grow only polynomial in size

Theorem

- *Let $\mathcal{R}$ be an S-sorted constructor TRS based on a simple signature. If $\mathcal{R} \subseteq >_{\mathsf{pps}*}$ then the functions computed by $\mathcal{R}$ are polytime computable.*

Example

- $s_1 : \mathsf{Word} \to \mathsf{Word}$ ✓
- $(:) : \mathsf{a} \times \mathsf{List(a)} \to \mathsf{List(a)}$ ✓
- $\mathsf{node} : \mathsf{Tree} \times \mathsf{Tree} \to \mathsf{Tree}$ ✗

Polytime Computability and $>_{\mathsf{pps}*}$

terms grow only polynomial in size

Theorem

- *Let $\mathcal{R}$ be an S-sorted constructor TRS based on a simple signature. If $\mathcal{R} \subseteq >_{\mathsf{pps}*}$ then the functions computed by $\mathcal{R}$ are polytime computable.*
- *Each polytime computable function is computable by an S-sorted, orthogonal, constructor TRS compatible with $>_{\mathsf{pps}*}$.*

Example

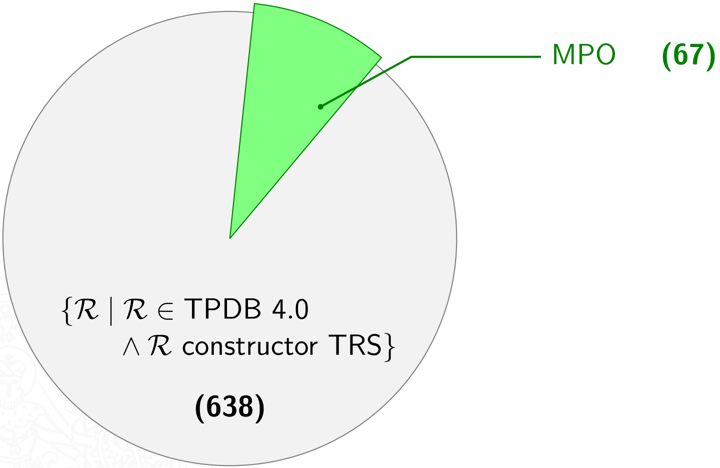
- s_1 : $\mathsf{Word} \to \mathsf{Word}$ ✓
- $(:)$: $\mathsf{a} \times \mathsf{List(a)} \to \mathsf{List(a)}$ ✓
- node : $\mathsf{Tree} \times \mathsf{Tree} \to \mathsf{Tree}$ ✗

Experimental Results

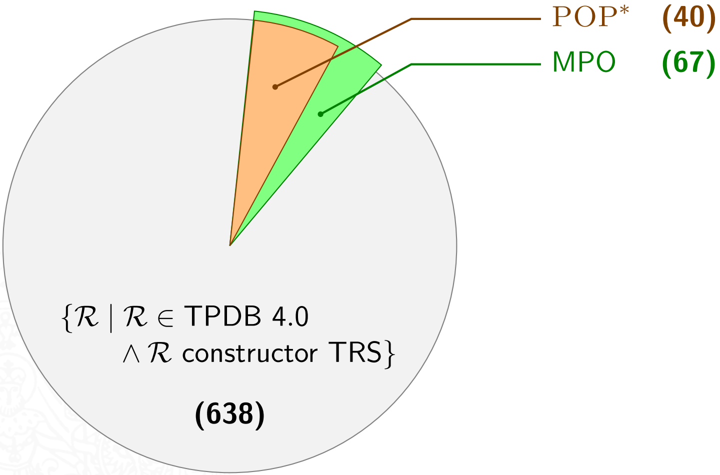
$\{\mathcal{R} \mid \mathcal{R} \in \text{TPDB 4.0}$
$\quad \wedge \mathcal{R} \text{ constructor TRS}\}$

(638)

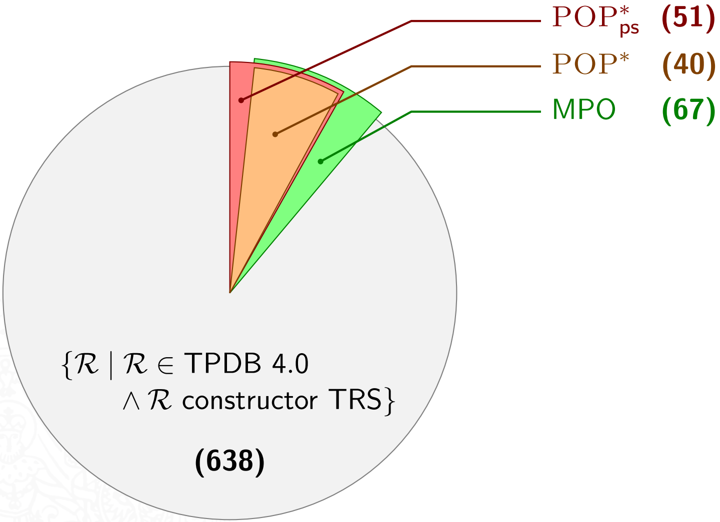
Experimental Results



Experimental Results



Experimental Results



Conclusion

- integrating the rules of predicative recursion with parameter substitution increases the power of polynomial path orders considerably

☺

- term rewriting allows us to combine different techniques for complexity analysis
  - $>_{\mathsf{pps}*}$ + argument filterings used with weak dependency pairs
  - $>_{\mathsf{pps}*}$ + (finite) semantic labeling
  - ...

☺

- $>_{\mathsf{pps}*}$ extensible to non-constructor TRS by putting additional constraints on the precedence

hack ☹

Polynomial Runtime Complexity $\not\Rightarrow$ Polytime Computability

$$\begin{aligned} \mathsf{f}(\epsilon) &\to \mathsf{tip} & \qquad \mathsf{mkTree}(x) &\to \mathsf{node}(x, x) \\ \mathsf{f}(\mathsf{s}_i(x)) &\to \mathsf{mkTree}(\mathsf{f}(x)) \end{aligned}$$

Polynomial Runtime Complexity $\not\Rightarrow$ Polytime Computability

$$\begin{aligned} \mathsf{f}(\epsilon;) &\to \mathsf{tip} & \mathsf{mkTree}(;x) &\to \mathsf{node}(;x,x) \\ \mathsf{f}(\mathsf{s}_i(;x);) &\to \mathsf{mkTree}(;\mathsf{f}(x;)) \end{aligned}$$

$$\mathsf{f} > \mathsf{mkTree} > \mathsf{node} > \mathsf{tip}$$

Polynomial Runtime Complexity $\not\Rightarrow$ Polytime Computability

$$\mathsf{f}(\epsilon;) >_{\mathsf{pps}*} \mathsf{tip} \qquad \mathsf{mkTree}(;x) >_{\mathsf{pps}*} \mathsf{node}(;x,x)$$

$$\mathsf{f}(\mathsf{s}_i(;x);) >_{\mathsf{pps}*} \mathsf{mkTree}(;\mathsf{f}(x;))$$

$$\mathsf{f} > \mathsf{mkTree} > \mathsf{node} > \mathsf{tip}$$

Polynomial Runtime Complexity $\not\Rightarrow$ Polytime Computability

$$\begin{aligned} \mathsf{f}(\epsilon;\,) &\to \mathsf{tip} & \qquad \mathsf{mkTree}(;x) &\to \mathsf{node}(;x,x) \\ \mathsf{f}(\mathsf{s}_i(;x);\,) &\to \mathsf{mkTree}(;\mathsf{f}(x;\,)) \end{aligned}$$

Observations

- $\mathsf{rc}^{\mathsf{i}}_{\mathcal{R}}$ of above TRS $\mathcal{R}$ polynomial
- $\xrightarrow{\mathsf{i}}_{\mathcal{R}}$ does not give a (direct) polytime algorithm for computing f
  - terms grow exponentially in size

Controlling Size Growth

Definition

an S-sorted signature is called simple if for each constructor $c : s_1 \times \cdots \times s_n \to s$

- $\mathrm{rk}(s_i) \leqslant \mathrm{rk}(s)$ for all $i \in \{1, \ldots, n\}$
- $\mathrm{rk}(s_i) = \mathrm{rk}(s)$ for at most one $i \in \{1, \ldots, n\}$

$\mathrm{rk} : S \to \mathbb{N}$

Controlling Size Growth

Definition

an S-sorted signature is called simple if for each constructor
$c : s_1 \times \cdots \times s_n \to s$

- $\mathrm{rk}(s_i) \leqslant \mathrm{rk}(s)$ for all $i \in \{1, \ldots, n\}$
- $\mathrm{rk}(s_i) = \mathrm{rk}(s)$ for at most one $i \in \{1, \ldots, n\}$

$\mathrm{rk} : S \to \mathbb{N}$

Example

- $\mathrm{true} : \mathrm{Bool}$
- $s_1 : \mathrm{Word} \to \mathrm{Word}$
- $(:) : \mathrm{a} \times \mathrm{List(a)} \to \mathrm{List(a)}$
- $\mathrm{node} : \mathrm{Tree} \times \mathrm{Tree} \to \mathrm{Tree}$

Controlling Size Growth

Definition

an S-sorted signature is called simple if for each constructor $c : s_1 \times \cdots \times s_n \to s$

- $\mathrm{rk}(s_i) \leqslant \mathrm{rk}(s)$ for all $i \in \{1, \ldots, n\}$
- $\mathrm{rk}(s_i) = \mathrm{rk}(s)$ for at most one $i \in \{1, \ldots, n\}$

$\mathrm{rk} : S \to \mathbb{N}$

Example

- $\mathrm{true} : \mathrm{Bool}$ ✓
- $s_1 : \mathrm{Word} \to \mathrm{Word}$
- $(:) : \mathrm{a} \times \mathrm{List(a)} \to \mathrm{List(a)}$
- $\mathrm{node} : \mathrm{Tree} \times \mathrm{Tree} \to \mathrm{Tree}$

Controlling Size Growth

Definition

an S-sorted signature is called simple if for each constructor $c : s_1 \times \cdots \times s_n \to s$

- $\mathsf{rk}(s_i) \leqslant \mathsf{rk}(s)$ for all $i \in \{1, \ldots, n\}$
- $\mathsf{rk}(s_i) = \mathsf{rk}(s)$ for at most one $i \in \{1, \ldots, n\}$

$\mathsf{rk} : S \to \mathbb{N}$

Example

- $\mathsf{true} : \mathsf{Bool}$ ✓
- $s_1 : \mathsf{Word} \to \mathsf{Word}$ ✓
- $(:) : \mathsf{a} \times \mathsf{List(a)} \to \mathsf{List(a)}$
- $\mathsf{node} : \mathsf{Tree} \times \mathsf{Tree} \to \mathsf{Tree}$

Controlling Size Growth

Definition

an S-sorted signature is called simple if for each constructor $c : s_1 \times \cdots \times s_n \to s$

- $\mathrm{rk}(s_i) \leqslant \mathrm{rk}(s)$ for all $i \in \{1, \ldots, n\}$
- $\mathrm{rk}(s_i) = \mathrm{rk}(s)$ for at most one $i \in \{1, \ldots, n\}$

$\mathrm{rk} : S \to \mathbb{N}$

Example

- $\mathsf{true} : \mathsf{Bool}$ ✓
- $s_1 : \mathsf{Word} \to \mathsf{Word}$ ✓
- $(:) : \mathsf{a} \times \mathsf{List(a)} \to \mathsf{List(a)}$ ✓
- $\mathsf{node} : \mathsf{Tree} \times \mathsf{Tree} \to \mathsf{Tree}$

Controlling Size Growth

Definition

an S-sorted signature is called simple if for each constructor $c : s_1 \times \cdots \times s_n \to s$

- $\mathsf{rk}(s_i) \leqslant \mathsf{rk}(s)$ for all $i \in \{1, \ldots, n\}$
- $\mathsf{rk}(s_i) = \mathsf{rk}(s)$ for at most one $i \in \{1, \ldots, n\}$

$\mathsf{rk} : S \to \mathbb{N}$

Example

- $\mathsf{true} : \mathsf{Bool}$ ✓
- $s_1 : \mathsf{Word} \to \mathsf{Word}$ ✓
- $(:) : \mathsf{a} \times \mathsf{List(a)} \to \mathsf{List(a)}$ ✓
- $\mathsf{node} : \mathsf{Tree} \times \mathsf{Tree} \to \mathsf{Tree}$ ✗

Polytime Computability and $>_{\mathsf{pps}*}$

Theorem

let $\mathcal{R}$ be an S-sorted constructor TRS based on a simple signature

- *if $\mathcal{R}$ is compatible with $>_{\mathsf{pps}*}$ then the functions computed by $\mathcal{R}$ are polytime computable*
- *each polytime computable function is computable by such a TRS compatible with $>_{\mathsf{pps}*}$*

Polytime Computability and $>_{\mathsf{pps}*}$

Theorem

let $\mathcal{R}$ be an S-sorted constructor TRS based on a simple signature

- *if $\mathcal{R}$ is compatible with $>_{\mathsf{pps}*}$ then the functions computed by $\mathcal{R}$ are polytime computable*
- *each polytime computable function is computable by such a TRS compatible with $>_{\mathsf{pps}*}$*

$$\mathsf{f}(\epsilon;) \to \mathsf{tip} \qquad \mathsf{mkTree}(;x) \to \mathsf{node}(;x,x)$$

$$\mathsf{f}(\mathsf{s}_i(;x);) \to \mathsf{mkTree}(;\mathsf{f}(x;))$$

$$\mathsf{node} : \mathsf{Tree} \times \mathsf{Tree} \to \mathsf{Tree} \qquad ✗$$

Polytime Computability and $>_{\mathsf{pps}*}$

Theorem

let $\mathcal{R}$ be an S-sorted constructor TRS based on a simple signature

- *if $\mathcal{R}$ is compatible with $>_{\mathsf{pps}*}$ then the functions computed by $\mathcal{R}$ are polytime computable*
- *each polytime computable function is computable by such a TRS compatible with $>_{\mathsf{pps}*}$*

$$\mathsf{rev}(xs;) \to \mathsf{rev}_{\mathsf{tl}}(xs; []) \qquad \mathsf{rev}_{\mathsf{tl}}([]; ys) \to ys$$

$$\mathsf{rev}_{\mathsf{tl}}(x{:}xs; ys) \to \mathsf{rev}_{\mathsf{tl}}(xs; x{:}ys)$$

Polytime Computability and $>_{\mathsf{pps}*}$

Theorem

let $\mathcal{R}$ be an S-sorted constructor TRS based on a simple signature

- *if $\mathcal{R}$ is compatible with $>_{\mathsf{pps}*}$ then the functions computed by $\mathcal{R}$ are polytime computable*
- *each polytime computable function is computable by such a TRS compatible with $>_{\mathsf{pps}*}$*

$$\mathsf{rev}(xs;\,) \to \mathsf{rev}_{\mathsf{tl}}(xs;[\,]) \qquad \mathsf{rev}_{\mathsf{tl}}([\,];ys) \to ys$$

$$\mathsf{rev}_{\mathsf{tl}}(x{:}xs;ys) \to \mathsf{rev}_{\mathsf{tl}}(xs;x{:}ys)$$

$$[\,] \colon \mathsf{List(a)} \qquad \mathsf{rev} \colon \mathsf{List(a)} \to \mathsf{List(a)}$$

$$(:) \colon \mathsf{a} \times \mathsf{List(a)} \to \mathsf{List(a)} \qquad \mathsf{rev}_{\mathsf{tl}} \colon \mathsf{List(a)} \times \mathsf{List(a)} \to \mathsf{List(a)}$$

Polytime Computability and $>_{\mathsf{pps}*}$

Theorem

let $\mathcal{R}$ be an S-sorted constructor TRS based on a simple signature

- *if $\mathcal{R}$ is compatible with $>_{\mathsf{pps}*}$ then the functions computed by $\mathcal{R}$ are polytime computable*
- *each polytime computable function is computable by such a TRS compatible with $>_{\mathsf{pps}*}$*

$$\mathsf{rev}(xs;\,) \to \mathsf{rev}_{\mathsf{tl}}(xs;[\,]) \qquad \begin{aligned} \mathsf{rev}_{\mathsf{tl}}([\,];ys) &\to ys \\ \mathsf{rev}_{\mathsf{tl}}(x{:}xs;ys) &\to \mathsf{rev}_{\mathsf{tl}}(xs;x{:}ys) \end{aligned}$$

$$\begin{aligned} [\,] &: \mathsf{List}(\mathsf{a}) \\ (:) &: \mathsf{a} \times \mathsf{List}(\mathsf{a}) \to \mathsf{List}(\mathsf{a}) \end{aligned} \qquad \begin{aligned} \mathsf{rev} &: \mathsf{List}(\mathsf{a}) \to \mathsf{List}(\mathsf{a}) \\ \mathsf{rev}_{\mathsf{tl}} &: \mathsf{List}(\mathsf{a}) \times \mathsf{List}(\mathsf{a}) \to \mathsf{List}(\mathsf{a}) \end{aligned}$$