A New Order-theoretic Characterisation of the Polytime Computable Functions

Martin Avanzini[1] *Naohi Eguchi*[2] *Georg Moser*[1]

[1]Institute of Computer Science
University of Innsbruck, Austria

[2]Mathematical Institute
Tohoku University, Japan

March 28, 2012

Implicit Computational Complexity (ICC)

characterise complexity classes

- − external restriction of resources
- \+ restrict program structure

Motivation

- broaden understanding
- apply in programming language theory

Cobhams Definition of FP

Alan Cobham

The Intrinsic Computational Difficulty of Functions.

Proc. of the 1964 International Congress for Logic, Methodology, and the Philosophy of Science, pages 24–30, 1964

Cobhams Definition of FP

Alan Cobham

The Intrinsic Computational Difficulty of Functions.

Proc. of the 1964 International Congress for Logic, Methodology, and the Philosophy of Science, pages 24–30, 1964

Bounded Recursion on Notation

$$f(\varepsilon, \vec{y}) = g(\vec{y})$$
$$f(zi, \vec{y}) = h_i(z, \vec{y}, f(z, \vec{y})) \quad \text{for } i = 0, 1$$

- provided $f(z, \vec{y}) \leqslant p(z, \vec{y})$ for previously defined p and all $z, \vec{y}$
- requires smash function $x\#y := 2^{|x|\cdot|y|}$

Cobhams Definition of FP

Alan Cobham

The Intrinsic Computational Difficulty of Functions.

Proc. of the 1964 International Congress for Logic, Methodology, and the Philosophy of Science, pages 24–30, 1964

Bounded Recursion on Notation

$$f(\varepsilon, \vec{y}) = g(\vec{y})$$
$$f(zi, \vec{y}) = h_i(z, \vec{y}, f(z, \vec{y})) \quad \text{for } i = 0, 1$$

- provided $|f(z, \vec{y})| \leqslant |p(z, \vec{y})|$ for previously defined p and all $z, \vec{y}$
- requires smash function $x\#y := 2^{|x|\cdot|y|}$

Bellantoni & Cooks Definition of FP

Stephen Bellantoni and Stephen A. Cook
A New Recursion-Theoretic Characterization of the Polytime Functions.
Computational Complexity, Vol. 2, pages 97–110, 1992

Bellantoni & Cooks Definition of FP

Stephen Bellantoni and Stephen A. Cook
A New Recursion-Theoretic Characterization of the Polytime Functions.
Computational Complexity, Vol. 2, pages 97–110, 1992

Predicative Recursion on Notation

$$f(\varepsilon, \vec{x}; \vec{y}) = g(\vec{x}; \vec{y})$$
$$f(zi, \vec{x}; \vec{y}) = h_i(z, \vec{x}; \vec{y}, f(z, \vec{x}; \vec{y})) \quad \text{for } i = 0, 1$$

- uses separation of arguments

$$f(\underbrace{n_1, \ldots, n_l}_{\textit{normal}}; \underbrace{n_{l+1}, \ldots, n_{l+k}}_{\textit{safe}})$$

Bellantoni & Cooks Definition of FP

Stephen Bellantoni and Stephen A. Cook
A New Recursion-Theoretic Characterization of the Polytime Functions.
Computational Complexity, Vol. 2, pages 97–110, 1992

Predicative Recursion on Notation

$$f(\varepsilon, \vec{x}; \vec{y}) = g(\vec{x}; \vec{y})$$
$$f(zi, \vec{x}; \vec{y}) = h_i(z, \vec{x}; \vec{y}, f(z, \vec{x}; \vec{y})) \quad \text{for } i = 0, 1$$

- uses separation of arguments

$$f(\underbrace{n_1, \ldots, n_l}_{\textit{normal}}; \underbrace{n_{l+1}, \ldots, n_{l+k}}_{\textit{safe}})$$

- *"complexity depends only on normal arguments"*

Runtime Complexity Analysis of TRSs

Computation

- conceive TRS $\mathcal{R}$ as first order program over signature $\mathcal{D} \uplus \mathcal{C}$
- values $\mathcal{V}\mathrm{al}$ are terms built from constructors $\mathcal{C}$
- for each $\mathsf{f} \in \mathcal{D}$, $\mathcal{R}$ computes functions $f : \mathcal{V}\mathrm{al}^n \to \mathcal{V}\mathrm{al}$ such that

$$f(v_1, \ldots, v_n) := u \qquad \Longleftrightarrow \qquad \mathsf{f}(v_1, \ldots, v_n) \to^{!}_{\mathcal{R}} u$$

Runtime Complexity Analysis of TRSs

Computation

- conceive TRS $\mathcal{R}$ as first order program over signature $\mathcal{D} \uplus \mathcal{C}$
- values $\mathcal{V}\mathrm{al}$ are terms built from constructors $\mathcal{C}$
- for each $\mathsf{f} \in \mathcal{D}$, $\mathcal{R}$ computes functions $f : \mathcal{V}\mathrm{al}^n \to \mathcal{V}\mathrm{al}$ such that

$$f(v_1, \ldots, v_n) := u \qquad \Longleftrightarrow \qquad \mathsf{f}(v_1, \ldots, v_n) \to_{\mathcal{R}}^{!} u$$

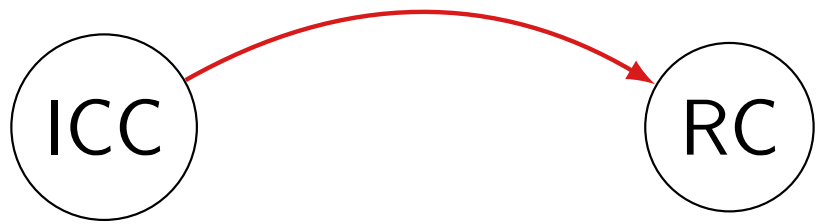
Runtime Complexity

- runtime complexity function

$$\mathrm{rc}_{\mathcal{R}}(n) := \max\{\mathrm{dh}(t, \to_{\mathcal{R}}) \mid t \text{ is basic term of size up to } n\}$$

$$\text{where } \mathrm{dh}(t, \to) := \max\{\ell \mid t \to t_1 \to t_2 \cdots \to t_\ell\}$$

- term $\mathsf{f}(v_1, \ldots, v_n)$ is basic if $f \in \mathcal{D}$ and all $v_i \in \mathcal{V}\mathrm{al}$


ICC
RC

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

M. Avanzini and G. Moser
Complexity Analysis by Rewriting.
Proc. of 9th International Symposium on Functional and Logic Programming, LNCS Vol. 4989, pages 130–146, 2008

- order $>_{\mathsf{pop}*}$ embodies predicative recursion on $>_{\mathsf{mpo}}$

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

M. Avanzini and G. Moser
Complexity Analysis by Rewriting.
Proc. of 9th International Symposium on Functional and Logic Programming, LNCS Vol. 4989, pages 130–146, 2008

- order $>_{\mathsf{pop}*}$ embodies predicative recursion on $>_{\mathsf{mpo}}$

Example

$$
\begin{array}{ll}
1: & +(0,y) \to y \\
2: & +(\mathsf{s}(x),y) \to \mathsf{s}(+(x,y)) \\
5: & \mathsf{sq}(x\ ) \to \times(x,x\ )
\end{array}
\qquad
\begin{array}{ll}
3: & \times(0,y\ ) \to 0 \\
4: & \times(\mathsf{s}(x),y\ ) \to +(y,\times(x,y\ ))
\end{array}
$$

- TRS is compatible with $>_{\mathsf{mpo}}$ using precedence $\mathsf{sq} \succ \times \succ + \succ \mathsf{s}$

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

M. Avanzini and G. Moser
Complexity Analysis by Rewriting.
Proc. of 9th International Symposium on Functional and Logic Programming, LNCS Vol. 4989, pages 130–146, 2008

- order $>_{\mathsf{pop}*}$ embodies predicative recursion on $>_{\mathsf{mpo}}$

Example

$$
\begin{array}{ll}
1: & +(0;y) \to y \\
2: & +(\mathsf{s}(x);y) \to \mathsf{s}(+(x;y)) \\
5: & \mathsf{sq}(x;) \to \times(x,x;)
\end{array}
\qquad
\begin{array}{ll}
3: & \times(0,y;) \to 0 \\
4: & \times(\mathsf{s}(x),y;) \to +(y;\times(x,y;))
\end{array}
$$

- TRS is compatible with $>_{\mathsf{pop}*}$ using precedence $\mathsf{sq} \succ \times \succ + \succ \mathsf{s}$

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

Runtime Complexity Analysis

Let $\mathcal{R}$ denote a constructor TRS. There exists $k \in \mathbb{N}$ such that

$$\mathcal{R} \subseteq >_{\mathsf{pop}*} \quad \Longrightarrow \quad \mathsf{rc}^{\mathsf{i}}_{\mathcal{R}} \in \mathsf{O}(n^k)$$

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

Runtime Complexity Analysis

Let $\mathcal{R}$ denote a constructor TRS. There exists $k \in \mathbb{N}$ such that

$$\mathcal{R} \subseteq >_{\mathsf{pop}*} \quad \Longrightarrow \quad \mathsf{rc}^{\mathsf{i}}_{\mathcal{R}} \in \mathsf{O}(n^k)$$

Implicit Computational Complexity

1. **Soundness**
   Let $\mathcal{R}$ be a constructor TRS that computes a function f.
   If $\mathcal{R} \subseteq >_{\mathsf{pop}*}$ then f is polytime computable.

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

Runtime Complexity Analysis

Let $\mathcal{R}$ denote a constructor TRS. There exists $k \in \mathbb{N}$ such that

$$\mathcal{R} \subseteq >_{\mathsf{pop}*} \quad \Longrightarrow \quad \mathsf{rc}^{\mathsf{i}}_{\mathcal{R}} \in \mathsf{O}(n^k)$$

Implicit Computational Complexity

1. **Soundness**
   Let $\mathcal{R}$ be a constructor TRS that computes a function f.
   If $\mathcal{R} \subseteq >_{\mathsf{pop}*}$ then f is polytime computable.

2. **Completeness**
   Let f be a polytime computable function.
   There exists a constructor TRS $\mathcal{R}_f$ computing f with $\mathcal{R}_f \subseteq >_{\mathsf{pop}*}$.

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

Runtime Complexity Analysis

Let $\mathcal{R}$ denote a constructor TRS. There exists $k \in \mathbb{N}$ such that

$$\mathcal{R} \subseteq >_{\mathsf{pop}*} \quad \Longrightarrow \quad \mathsf{rc}^{\mathsf{i}}_{\mathcal{R}} \in \mathsf{O}(n^k)$$

Implicit Computational Complexity

1. **Soundness**
   Let $\mathcal{R}$ be a constructor TRS that computes a function f.
   If $\mathcal{R} \subseteq >_{\mathsf{pop}*}$ then f is polytime computable.
2. **Completeness**
   Let f be a polytime computable function.
   There exists a constructor TRS $\mathcal{R}_f$ computing f with $\mathcal{R}_f \subseteq >_{\mathsf{pop}*}$.

Drawback

order $>_{\mathsf{pop}*}$ cannot precisely estimate degree of polynomial

Small Polynomial Path Order $>_{\mathsf{spop}*}$

- restriction of polynomial path order $>_{\mathsf{spop}*} \subseteq >_{\mathsf{pop}*}$
- estimate degree of complexity certificate in *"depth of recursion"*

Small Polynomial Path Order $>_{\mathsf{spop}*}$

- restriction of polynomial path order $>_{\mathsf{spop}*} \subseteq >_{\mathsf{pop}*}$

- estimate degree of complexity certificate in *"depth of recursion"*
  1. restrict composition to safe arguments — weak safe composition

$$f(\vec{x};\vec{y}) = g(\vec{x};h_1(\vec{x};\vec{y}),\ldots,h_k(\vec{x};\vec{y}))$$

Small Polynomial Path Order $>_{\mathsf{spop}*}$

- restriction of polynomial path order $>_{\mathsf{spop}*} \subseteq >_{\mathsf{pop}*}$

- estimate degree of complexity certificate in *"depth of recursion"*

  1. restrict composition to safe arguments — weak safe composition

$$f(\vec{x};\vec{y}) = g(\vec{x};h_1(\vec{x};\vec{y}),\ldots,h_k(\vec{x};\vec{y}))$$

  2. uses product-status

$$\begin{aligned}
\mathsf{f}_k(\mathsf{s}(x_1),x_2,x_3,\ldots,x_k;) &\to \mathsf{f}_k(x_1,x_2,x_3,\ldots,x_k;)\\
\mathsf{f}_k(0,\mathsf{s}(x_2),x_3,\ldots,x_k;) &\to \mathsf{f}_k(x_2,x_2,x_3,\ldots,x_k;)\\
&\vdots\\
\mathsf{f}_k(0,\ldots,0,\mathsf{s}(x_k);) &\to \mathsf{f}_k(x_k,\ldots,x_k,x_k;)
\end{aligned}$$

  - compatible with $>_{\mathsf{pop}*}$, admits runtime complexity in $\Theta(n^k)$

Small Polynomial Path Order $>_{\mathsf{spop}*}$

$f(s_1, \ldots, s_k; s_{k+1}, \ldots, s_{k+l}) >_{\mathsf{spop}*} t$ if

1. $s_i \geqslant_{\mathsf{spop}*} t$ for some argument s_i

Small Polynomial Path Order $>_{\text{spop}*}$

$f(s_1, \ldots, s_k; s_{k+1}, \ldots, s_{k+l}) >_{\text{spop}*} t$ if

1. $s_i \geqslant_{\text{spop}*} t$ for some argument s_i
2. $t = g(t_1, \ldots, t_m; t_{m+1}, \ldots, t_{m+n})$ where $f \in \mathcal{D}$ and $f \succ g$
   - $f(s_1, \ldots, s_k; s_{k+1}, \ldots, s_{k+l}) \rhd_n t_j$ for all normal arguments t_j
   - $f(s_1, \ldots, s_k; s_{k+1}, \ldots, s_{k+l}) >_{\text{spop}*} t_j$ for all safe arguments t_j

$$f(\vec{x}; \vec{y}) = g(\vec{x}; h_1(\vec{x}; \vec{y}), \ldots, h_k(\vec{x}; \vec{y}))$$

Small Polynomial Path Order $>_{\mathsf{spop}*}$

$f(s_1, \ldots, s_k; s_{k+1}, \ldots, s_{k+l}) >_{\mathsf{spop}*} t$ if

1. $s_i \geqslant_{\mathsf{spop}*} t$ for some argument s_i
2. $t = g(t_1, \ldots, t_m; t_{m+1}, \ldots, t_{m+n})$ where $f \in \mathcal{D}$ and $f \succ g$
   - $f(s_1, \ldots, s_k; s_{k+1}, \ldots, s_{k+l}) \rhd_{\mathsf{n}} t_j$ for all normal arguments t_j
   - $f(s_1, \ldots, s_k; s_{k+1}, \ldots, s_{k+l}) >_{\mathsf{spop}*} t_j$ for all safe arguments t_j

$$f(s_1, \ldots, s_k; \ldots) \rhd_{\mathsf{n}} t \quad :\Leftrightarrow \quad s_i \trianglerighteq \cdot \sim t$$

Small Polynomial Path Order $>_{\text{spop}*}$

$f(s_1, \ldots, s_k; s_{k+1}, \ldots, s_{k+l}) >_{\text{spop}*} t$ if

1. $s_i \geqslant_{\text{spop}*} t$ for some argument s_i
2. $t = g(t_1, \ldots, t_m; t_{m+1}, \ldots, t_{m+n})$ where $f \in \mathcal{D}$ and $f \succ g$
   - $f(s_1, \ldots, s_k; s_{k+1}, \ldots, s_{k+l}) \rhd_n t_j$ for all normal arguments t_j
   - $f(s_1, \ldots, s_k; s_{k+1}, \ldots, s_{k+l}) >_{\text{spop}*} t_j$ for all safe arguments t_j
3. $t = g(t_1, \ldots, t_k; t_{k+1}, \ldots, t_{k+l})$ where $f \in \mathcal{D}$ and $f \sim g$
   - $\langle s_1, \ldots, s_k \rangle >_{\text{spop}*} \langle t_{\pi(1)}, \ldots, t_{\pi(k)} \rangle$
   - $\langle s_{k+1}, \ldots, s_{k+l} \rangle \geqslant_{\text{spop}*} \langle t_{\tau(k+1)}, \ldots, t_{\tau(k+l)} \rangle$

$$f(\varepsilon, \vec{x}; \vec{y}) = g(\vec{x}; \vec{y})$$
$$f(zi, \vec{x}; \vec{y}) = h_i(z, \vec{x}; \vec{y}, f(z, \vec{x}; \vec{y})) \quad \text{for } i = 0, 1$$

Small Polynomial Path Order $>_{\mathsf{spop}*}$

$f(s_1, \ldots, s_k; s_{k+1}, \ldots, s_{k+l}) >_{\mathsf{spop}*} t$ if

1. $s_i \geqslant_{\mathsf{spop}*} t$ for some argument s_i
2. $t = g(t_1, \ldots, t_m; t_{m+1}, \ldots, t_{m+n})$ where $f \in \mathcal{D}$ and $f \succ g$
   - $f(s_1, \ldots, s_k; s_{k+1}, \ldots, s_{k+l}) \rhd_{\mathsf{n}} t_j$ for all normal arguments t_j
   - $f(s_1, \ldots, s_k; s_{k+1}, \ldots, s_{k+l}) >_{\mathsf{spop}*} t_j$ for all safe arguments t_j
   - t contains at most one defined symbol h with $h \succcurlyeq f$
3. $t = g(t_1, \ldots, t_k; t_{k+1}, \ldots, t_{k+l})$ where $f \in \mathcal{D}$ and $f \sim g$
   - $\langle s_1, \ldots, s_k \rangle >_{\mathsf{spop}*} \langle t_{\pi(1)}, \ldots, t_{\pi(k)} \rangle$
   - $\langle s_{k+1}, \ldots, s_{k+l} \rangle \geqslant_{\mathsf{spop}*} \langle t_{\tau(k+1)}, \ldots, t_{\tau(k+l)} \rangle$

$$\mathsf{f}(\mathsf{s}(x);) \not\geqslant_{\mathsf{spop}*} \mathsf{c}(;\mathsf{f}(x;),\mathsf{f}(x;))$$

Small Polynomial Path Order $>_{\text{spop}*}$

$f(s_1, \ldots, s_k; s_{k+1}, \ldots, s_{k+l}) >_{\text{spop}*} t$ if

1. $s_i \geqslant_{\text{spop}*} t$ for some argument s_i
2. $t = g(t_1, \ldots, t_m; t_{m+1}, \ldots, t_{m+n})$ where $f \in \mathcal{D}$ and $f \succ g$
   - $f(s_1, \ldots, s_k; s_{k+1}, \ldots, s_{k+l}) \rhd_n t_j$ for all normal arguments t_j
   - $f(s_1, \ldots, s_k; s_{k+1}, \ldots, s_{k+l}) >_{\text{spop}*} t_j$ for all safe arguments t_j
   - t contains at most one defined symbol h with $h \succcurlyeq f$
3. $t = g(t_1, \ldots, t_k; t_{k+1}, \ldots, t_{k+l})$ where $f \in \mathcal{D}_{\text{rec}}$ and $f \sim g$
   - $\langle s_1, \ldots, s_k \rangle >_{\text{spop}*} \langle t_{\pi(1)}, \ldots, t_{\pi(k)} \rangle$
   - $\langle s_{k+1}, \ldots, s_{k+l} \rangle \geqslant_{\text{spop}*} \langle t_{\tau(k+1)}, \ldots, t_{\tau(k+l)} \rangle$

assume designated set of recursive symbols $\mathcal{D}_{\text{rec}} \subseteq \mathcal{D}$

Small Polynomial Path Order $>_{\mathsf{spop}*}$

$f(s_1, \ldots, s_k; s_{k+1}, \ldots, s_{k+l}) >_{\mathsf{spop}*} t$ if

1. $s_i \geqslant_{\mathsf{spop}*} t$ for some argument s_i
2. $t = g(t_1, \ldots, t_m; t_{m+1}, \ldots, t_{m+n})$ where $f \in \mathcal{D}$ and $f \succ g$
   - $f(s_1, \ldots, s_k; s_{k+1}, \ldots, s_{k+l}) \rhd_{\mathsf{n}} t_j$ for all normal arguments t_j
   - $f(s_1, \ldots, s_k; s_{k+1}, \ldots, s_{k+l}) >_{\mathsf{spop}*} t_j$ for all safe arguments t_j
   - t contains at most one defined symbol h with $h \succcurlyeq f$
3. $t = g(t_1, \ldots, t_k; t_{k+1}, \ldots, t_{k+l})$ where $f \in \mathcal{D}_{\mathsf{rec}}$ and $f \sim g$
   - $\langle s_1, \ldots, s_k \rangle >_{\mathsf{spop}*} \langle t_{\pi(1)}, \ldots, t_{\pi(k)} \rangle$
   - $\langle s_{k+1}, \ldots, s_{k+l} \rangle \geqslant_{\mathsf{spop}*} \langle t_{\tau(k+1)}, \ldots, t_{\tau(k+l)} \rangle$

Small Polynomial Path Order $>_{\mathsf{spop}*}$

Main Result

- the depth of recursion $\mathsf{rd}(f)$ is defined as follows:

$$\mathsf{rd}(f) := \begin{cases} 1 + \max\{\mathsf{rd}(g) \mid f \succ g\} & \text{if } f \in \mathcal{D}_{\mathsf{rec}} \\ \max\{\mathsf{rd}(g) \mid f \succ g\} & \text{otherwise} \end{cases}$$

Small Polynomial Path Order $>_{\mathsf{spop}*}$

Main Result

- the depth of recursion $\mathsf{rd}(f)$ is defined as follows:

$$\mathsf{rd}(f) := \begin{cases} 1 + \max\{\mathsf{rd}(g) \mid f \succ g\} & \text{if } f \in \mathcal{D}_{\mathsf{rec}} \\ \max\{\mathsf{rd}(g) \mid f \succ g\} & \text{otherwise} \end{cases}$$

- constructor TRS $\mathcal{R}$ is predicative recursive of degree d if
  - $\mathcal{R}$ is compatible with $>_{\mathsf{spop}*}$
  - depth of recursion of symbol in $\mathcal{R}$ is at most d

Small Polynomial Path Order $>_{\mathsf{spop}*}$

Main Result

- the depth of recursion $\mathsf{rd}(f)$ is defined as follows:

$$\mathsf{rd}(f) := \begin{cases} 1 + \max\{\mathsf{rd}(g) \mid f \succ g\} & \text{if } f \in \mathcal{D}_{\mathsf{rec}} \\ \max\{\mathsf{rd}(g) \mid f \succ g\} & \text{otherwise} \end{cases}$$

- constructor TRS $\mathcal{R}$ is predicative recursive of degree d if
  - $\mathcal{R}$ is compatible with $>_{\mathsf{spop}*}$
  - depth of recursion of symbol in $\mathcal{R}$ is at most d

Theorem

1. if $\mathcal{R}$ is a predicative recursive TRS of degree d, then $\mathsf{rc}^{\mathsf{i}}_{\mathcal{R}} \in \mathsf{O}(n^d)$

Small Polynomial Path Order $>_{\mathsf{spop}*}$

Main Result

- the depth of recursion $\mathsf{rd}(f)$ is defined as follows:

$$\mathsf{rd}(f) := \begin{cases} 1 + \max\{\mathsf{rd}(g) \mid f \succ g\} & \text{if } f \in \mathcal{D}_{\mathsf{rec}} \\ \max\{\mathsf{rd}(g) \mid f \succ g\} & \text{otherwise} \end{cases}$$

- constructor TRS $\mathcal{R}$ is predicative recursive of degree d if
  - $\mathcal{R}$ is compatible with $>_{\mathsf{spop}*}$
  - depth of recursion of symbol in $\mathcal{R}$ is at most d

Theorem

1. if $\mathcal{R}$ is a predicative recursive TRS of degree d, then $\mathsf{rc}^{\mathsf{i}}_{\mathcal{R}} \in \mathsf{O}(n^d)$
2. for all d there exists a predicative recursive TRS of degree d such that $\mathsf{rc}^{\mathsf{i}}_{\mathcal{R}} \in \Theta(n^d)$

Small Polynomial Path Order $>_{\mathsf{spop}*}$

Example

$$
\begin{aligned}
1 &: \quad +(0; y) \to y & 3 &: \quad \times(0, y;) \to 0 \\
2 &: +(\mathsf{s}(x); y) \to \mathsf{s}(+(x; y)) & 4 &: \times(\mathsf{s}(x), y;) \to +(y; \times(x, y;)) \\
5 &: \quad \mathsf{sq}(x;) \to \times(x, x;)
\end{aligned}
$$

- TRS is compatible with $>_{\mathsf{spop}*}$ using precedence $\mathsf{sq} \succ \times \succ + \succ \mathsf{s}$
- only $\times$ and $+$ are recursive
- innermost runtime complexity is quadratic

Experimental Results

bound	MPO	POP*	sPOP*
yes	$76_{\backslash 0.09}$	$43_{\backslash 0.05}$	$39_{\backslash 0.07}$

Table: 757 constructor TRSs from TPDB 8.0

Experimental Results

bound	MPO	POP*	sPOP*
$O(1)$			$9_{\backslash 0.06}$
$O(n^1)$			$32_{\backslash 0.07}$
$O(n^2)$			$38_{\backslash 0.09}$
$O(n^3)$			$39_{\backslash 0.20}$
$O(n^k)$		$43_{\backslash 0.05}$	$39_{\backslash 0.20}$
yes	$76_{\backslash 0.09}$	$43_{\backslash 0.05}$	$39_{\backslash 0.07}$

Table: 757 constructor TRSs from TPDB 8.0

Experimental Results

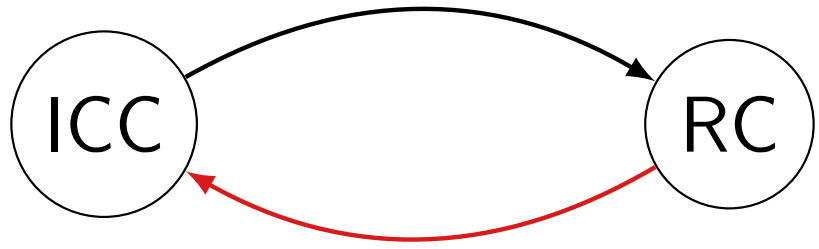
bound	MPO	POP*	sPOP*	POP^*_{PS}	$sPOP^*_{PS}$
$O(1)$			$9_{\backslash 0.06}$		$9_{\backslash 0.06}$
$O(n^1)$			$32_{\backslash 0.07}$		$46_{\backslash 0.09}$
$O(n^2)$			$38_{\backslash 0.09}$		$53_{\backslash 0.10}$
$O(n^3)$			$39_{\backslash 0.20}$		$54_{\backslash 0.22}$
$O(n^k)$		$43_{\backslash 0.05}$	$39_{\backslash 0.20}$	$56_{\backslash 0.05}$	$54_{\backslash 0.22}$
yes	$76_{\backslash 0.09}$	$43_{\backslash 0.05}$	$39_{\backslash 0.07}$	$56_{\backslash 0.05}$	$54_{\backslash 0.08}$

Table: 757 constructor TRSs from TPDB 8.0

- Parameter Substitution

$$f(\varepsilon, \vec{x}; \vec{y}) = g(\vec{x}; \vec{y})$$

$$f(zi, \vec{x}; \vec{y}) = h_i(z, \vec{x}; \vec{y}, f(z, \vec{x}; p_1(z, \vec{x}; \vec{y}), \ldots, p_k(z, \vec{x}; \vec{y}))) \quad \text{for } i = 0, 1$$


ICC
RC

Applications in ICC

Main Results

Theorem

$>_{\mathsf{spop}*}$ is sound and complete for polytime functions

Applications in ICC

Main Results

Theorem

$>_{\mathsf{spop}*}$ is sound and complete for polytime functions

- innermost runtime complexity is an invariant cost model

Applications in ICC

The Class $\mathcal{B}_{\mathsf{wsc}}$

class $\mathcal{B}_{\mathsf{wsc}}$ is smallest class of functions that

1. contains a small set of initial functions successors, conditional, ...
2. is closed under weak predicative composition

$$f(\vec{x};\vec{y}) = g(\vec{x}; h_1(\vec{x};\vec{y}), \ldots, h_k(\vec{x};\vec{y}))$$

3. is closed under safe recursion on notation

$$f(\varepsilon, \vec{x};\vec{y}) = g(\vec{x};\vec{y})$$
$$f(zi, \vec{x};\vec{y}) = h_i(z, \vec{x};\vec{y}, f(z, \vec{x};\vec{y})) \quad \text{for } i = 0, 1$$

Applications in ICC

The Class $\mathcal{B}_{\mathsf{wsc}}$

class $\mathcal{B}_{\mathsf{wsc}}$ is smallest class of functions that

1. contains a small set of initial functions successors, conditional, ...
2. is closed under weak predicative composition

$$f(\vec{x};\vec{y}) = g(\vec{x}; h_1(\vec{x};\vec{y}), \ldots, h_k(\vec{x};\vec{y}))$$

3. is closed under safe recursion on notation

$$f(\varepsilon, \vec{x};\vec{y}) = g(\vec{x};\vec{y})$$
$$f(zi, \vec{x};\vec{y}) = h_i(z, \vec{x};\vec{y}, f(z, \vec{x};\vec{y})) \quad \text{for } i = 0, 1$$

Theorem

$\mathcal{B}_{\mathsf{wsc}}$ equals the polytime computable functions

Applications in ICC

Main Results

Theorem

$>_{\mathsf{spop}*}$ is sound and complete for polytime functions

- innermost runtime complexity is an invariant cost model
- every $f \in \mathcal{B}_{\mathsf{wsc}}$ can be formulated as predicative recursive TRS

Applications in ICC

More Precise...

Theorem *RM* $\Rightarrow$ *TRS*

Suppose R is register machine operating in time $O(n^d)$.
If f is computed by R, then f is computable by some predicative recursive TRS $\mathcal{R}_f$ of degree d.

Applications in ICC

More Precise...

Theorem *RM ⇒ TRS*

Suppose R is register machine operating in time $O(n^d)$.
If f is computed by R, then f is computable by some predicative recursive TRS $\mathcal{R}_f$ of degree d.

Theorem *TRS ⇒ RM*

Suppose predicative TRS $\mathcal{R}$ of degree d computes functions on strings.
If f is computed by $\mathcal{R}$, then f is computable on a register machine R_f operating in time $O(n^d)$.

Future Work

- can Theorem TRS $\Rightarrow$ RM be generalised?

Future Work

- can Theorem TRS $\Rightarrow$ RM be generalised?
- can we replace weak safe composition by $f(\vec{x};\vec{y}) = g(h(\vec{x};);i(\vec{x};\vec{y}))$?

$$g \in \mathrm{O}(n^{d_g}), h \in \mathrm{O}(n^{d_h}), i \in \mathrm{O}(n^{d_i}) \quad \Rightarrow \quad f \in \mathrm{O}(n^{d})$$

where $d = \max\{\min\{1, d_g\} \cdot d_h, d_i\}$

Future Work

- can Theorem TRS $\Rightarrow$ RM be generalised?
- can we replace weak safe composition by $f(\vec{x};\vec{y}) = g(h(\vec{x};);i(\vec{x};\vec{y}))$?

$$g \in \mathrm{O}(n^{d_g}), h \in \mathrm{O}(n^{d_h}), i \in \mathrm{O}(n^{d_i}) \quad \Rightarrow \quad f \in \mathrm{O}(n^{d})$$

where $d = \max\{\min\{1, d_g\} \cdot d_h, d_i\}$

Conjecture

for any function f on strings, the following are equivalent

1. f is defined in $\mathcal{B}_{\mathrm{wsc}}$ via at most d-fold nested application of SRN
2. f is computed by predicative recursive TRS $\mathcal{R}_f$ of degree d
3. f is computed by register machine R_f operating in time $\mathrm{O}(n^d)$