Termination and Complexity in Higher-Order Term Rewriting

Part 5. Complexity: tuple interpretations

Cynthia Kop

ISR 2024

Derivation height

Derivation height

A measure of the “cost” of reducing a term to normal form.

Derivation height

A measure of the “cost” of reducing a term to normal form (worst-case).

Derivation height

A measure of the "cost" of reducing a term to normal form (worst-case).

$$\begin{aligned}
\mathtt{add}(x, \mathtt{0}) &\to x \\
\mathtt{add}(x, \mathtt{s}(y)) &\to \mathtt{s}(\mathtt{add}(x, y)) \\
\mathtt{mul}(x, \mathtt{0}) &\to \mathtt{0} \\
\mathtt{mul}(x, \mathtt{s}(y)) &\to \mathtt{add}(x, \mathtt{mul}(x, y))
\end{aligned}$$

Derivation height:

- $\mathtt{add}(\mathtt{0}, \mathtt{s}(\mathtt{0}))$:

Derivation height

A measure of the "cost" of reducing a term to normal form (worst-case).

$$\begin{aligned} \mathtt{add}(x, \mathtt{0}) &\to x \\ \mathtt{add}(x, \mathtt{s}(y)) &\to \mathtt{s}(\mathtt{add}(x, y)) \\ \mathtt{mul}(x, \mathtt{0}) &\to \mathtt{0} \\ \mathtt{mul}(x, \mathtt{s}(y)) &\to \mathtt{add}(x, \mathtt{mul}(x, y)) \end{aligned}$$

Derivation height:

- $\mathtt{add}(\mathtt{0}, \mathtt{s}(\mathtt{0}))$: 2 ($\mathtt{add}(\mathtt{0}, \mathtt{s}(\mathtt{0})) \to \mathtt{s}(\mathtt{add}(\mathtt{0}, \mathtt{0})) \to \mathtt{s}(\mathtt{0})$).

Derivation height

A measure of the “cost” of reducing a term to normal form (worst-case).

$$
\begin{array}{rcl}
\mathtt{add}(x, \mathtt{0}) & \to & x \\
\mathtt{add}(x, \mathtt{s}(y)) & \to & \mathtt{s}(\mathtt{add}(x, y)) \\
\mathtt{mul}(x, \mathtt{0}) & \to & \mathtt{0} \\
\mathtt{mul}(x, \mathtt{s}(y)) & \to & \mathtt{add}(x, \mathtt{mul}(x, y))
\end{array}
$$

Derivation height:

- $\mathtt{add}(\mathtt{0}, \mathtt{s}(\mathtt{0}))$: 2 ($\mathtt{add}(\mathtt{0}, \mathtt{s}(\mathtt{0})) \to \mathtt{s}(\mathtt{add}(\mathtt{0}, \mathtt{0})) \to \mathtt{s}(\mathtt{0})$).
- $\mathtt{mul}(\ \mathtt{mul}(\mathtt{s}(\mathtt{s}(\mathtt{0})), \mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{0}))))\ ,\ \mathtt{0})$:

Derivation height

A measure of the "cost" of reducing a term to normal form (worst-case).

$$\begin{array}{rcl} \mathtt{add}(x, \mathtt{0}) & \to & x \\ \mathtt{add}(x, \mathtt{s}(y)) & \to & \mathtt{s}(\mathtt{add}(x, y)) \\ \mathtt{mul}(x, \mathtt{0}) & \to & \mathtt{0} \\ \mathtt{mul}(x, \mathtt{s}(y)) & \to & \mathtt{add}(x, \mathtt{mul}(x, y)) \end{array}$$

Derivation height:

- $\mathtt{add}(\mathtt{0}, \mathtt{s}(\mathtt{0}))$: 2 ($\mathtt{add}(\mathtt{0}, \mathtt{s}(\mathtt{0})) \to \mathtt{s}(\mathtt{add}(\mathtt{0}, \mathtt{0})) \to \mathtt{s}(\mathtt{0})$).
- $\mathtt{mul}(\ \mathtt{mul}(\mathtt{s}(\mathtt{s}(\mathtt{0})), \mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{0}))))\ ,\ \mathtt{0})$: 15

Traditional interpretations (first-order)

Idea:

Traditional interpretations (first-order)

Idea:

- map every term s to $[\![s]\!] \in \mathbb{N}$

Traditional interpretations (first-order)

Idea:

- map every term s to $[\![s]\!] \in \mathbb{N}$
- make sure that $s \to t$ implies $[\![s]\!] > [\![t]\!]$

Traditional interpretations (first-order)

Idea:

- map every term s to $[\![s]\!] \in \mathbb{N}$
- make sure that $s \to t$ implies $[\![s]\!] > [\![t]\!]$

Then: $[\![s]\!] \geq \mathtt{derivationheight}(s)$!

Traditional interpretations (first-order)

Idea:

- map every term s to $[\![s]\!] \in \mathbb{N}$
- make sure that $s \to t$ implies $[\![s]\!] > [\![t]\!]$

Then: $[\![s]\!] \geq \texttt{derivationheight}(s)$!

Approach:

Traditional interpretations (first-order)

Idea:

- map every term s to $[\![s]\!] \in \mathbb{N}$
- make sure that $s \to t$ implies $[\![s]\!] > [\![t]\!]$

Then: $[\![s]\!] \geq \mathtt{derivationheight}(s)$!

Approach:

- map every function that takes k arguments to a monotonic function in $\mathbb{N}^k \mapsto \mathbb{N}$

Traditional interpretations (first-order)

Idea:

- map every term s to $[\![s]\!] \in \mathbb{N}$
- make sure that $s \to t$ implies $[\![s]\!] > [\![t]\!]$

Then: $[\![s]\!] \geq \texttt{derivationheight}(s)$!

Approach:

- map every function that takes k arguments to a monotonic function in $\mathbb{N}^k \mapsto \mathbb{N}$
- make sure that $[\![\ell]\!] > [\![r]\!]$ for all rules $\ell \to r$

Bounding derivation height with interpretations to $\mathbb{N}$

$$\begin{aligned} \mathtt{add}(\mathtt{0}, y) &\rightarrow y \\ \mathtt{add}(\mathtt{s}(x), y) &\rightarrow \mathtt{s}(\mathtt{add}(x, y)) \end{aligned}$$

Bounding derivation height with interpretations to $\mathbb{N}$

$$
\begin{array}{rcl}
\mathtt{add}(\mathtt{0}, y) & \to & y \\
\mathtt{add}(\mathtt{s}(x), y) & \to & \mathtt{s}(\mathtt{add}(x, y))
\end{array}
$$

Let:

- $[\![\mathtt{0}]\!] = 0$
- $[\![\mathtt{s}(x)]\!] = x + 1$
- $[\![\mathtt{add}(x, y)]\!] =$

Bounding derivation height with interpretations to $\mathbb{N}$

$$\begin{array}{rcl} \mathtt{add}(\mathtt{0}, y) & \rightarrow & y \\ \mathtt{add}(\mathtt{s}(x), y) & \rightarrow & \mathtt{s}(\mathtt{add}(x, y)) \end{array}$$

Let:

- $[\![\mathtt{0}]\!] = 0$
- $[\![\mathtt{s}(x)]\!] = x + 1$
- $[\![\mathtt{add}(x, y)]\!] = 1 + y + 2 * x$

Bounding derivation height with interpretations to $\mathbb{N}$

$$
\begin{array}{rcl}
\mathtt{add}(\mathtt{0}, y) & \to & y \\
\mathtt{add}(\mathtt{s}(x), y) & \to & \mathtt{s}(\mathtt{add}(x, y))
\end{array}
$$

Let:

- $[\![\mathtt{0}]\!] = 0$
- $[\![\mathtt{s}(x)]\!] = x + 1$
- $[\![\mathtt{add}(x, y)]\!] = 1 + y + 2 * x$

Then:

$$
\begin{array}{rcccl}
[\![\mathtt{add}(\mathtt{0}, y)]\!] & = & 1 + y & > & [\![y]\!] \\
[\![\mathtt{add}(\mathtt{s}(x), y)]\!] & = & 3 + y + 2 * x & > & 2 + y + 2 * x \\
 & & & = & [\![\mathtt{s}(\mathtt{add}(x, y))]\!]
\end{array}
$$

Bounding derivation height with interpretations to $\mathbb{N}$

$$\begin{array}{rcl} \mathtt{add}(\mathtt{0}, y) & \to & y \\ \mathtt{add}(\mathtt{s}(x), y) & \to & \mathtt{s}(\mathtt{add}(x, y)) \end{array}$$

Let:

- $[\![\mathtt{0}]\!] = 0$
- $[\![\mathtt{s}(x)]\!] = x + 1$
- $[\![\mathtt{add}(x, y)]\!] = 1 + y + 2 * x$

Then:

$$\begin{array}{rcccl} [\![\mathtt{add}(\mathtt{0}, y)]\!] & = & 1 + y & > & [\![y]\!] \\ [\![\mathtt{add}(\mathtt{s}(x), y)]\!] & = & 3 + y + 2 * x & > & 2 + y + 2 * x \\ & & & = & [\![\mathtt{s}(\mathtt{add}(x, y))]\!] \end{array}$$

Hence: $[\![\mathtt{add}(\mathtt{s}^n(\mathtt{0}),\ \mathtt{s}^m(\mathtt{0}))]\!] = 1 + m + 2 * n$: linear!

Monotonic algebras: definition

Given: a set $\mathcal{A}$ with a well-founded ordering $>$

Monotonic algebras: definition

Given: a set $\mathcal{A}$ with a well-founded ordering $>$ (for example: $\mathbb{N}$)

Monotonic algebras: definition

Given: a set $\mathcal{A}$ with a well-founded ordering $>$ (for example: $\mathbb{N}$)

Choose: a function $[\mathtt{f}]$ from $\mathcal{A}^k$ to $\mathcal{A}$ for every $\mathtt{f}$ of arity k

Monotonic algebras: definition

Given: a set $\mathcal{A}$ with a well-founded ordering $>$ (for example: $\mathbb{N}$)

Choose: a function $[\mathtt{f}]$ from $\mathcal{A}^k$ to $\mathcal{A}$ for every $\mathtt{f}$ of arity k

Define: for a given α mapping variables to $\mathcal{A}$:

- $[\![x]\!] = \alpha(x)$
- $[\![\mathtt{f}(s_1, \dots, s_k)]\!] = [\mathtt{f}]([\![s_1]\!], \dots, [\![s_k]\!])$

Monotonic algebras: definition

Given: a set $\mathcal{A}$ with a well-founded ordering $>$ (for example: $\mathbb{N}$)

Choose: a function $[\mathtt{f}]$ from $\mathcal{A}^k$ to $\mathcal{A}$ for every $\mathtt{f}$ of arity k

Define: for a given α mapping variables to $\mathcal{A}$:

- $[\![x]\!] = \alpha(x)$
- $[\![\mathtt{f}(s_1, \dots, s_k)]\!] = [\mathtt{f}]([\![s_1]\!], \dots, [\![s_k]\!])$

Prove: $[\![\ell]\!] > [\![r]\!]$ for all rules $\ell \to r$, all α

Monotonic algebras: definition

Given: a set $\mathcal{A}$ with a well-founded ordering $>$ (for example: $\mathbb{N}$)

Choose: a function $[\mathtt{f}]$ from $\mathcal{A}^k$ to $\mathcal{A}$ for every $\mathtt{f}$ of arity k

Define: for a given α mapping variables to $\mathcal{A}$:

- $[\![x]\!] = \alpha(x)$
- $[\![\mathtt{f}(s_1, \ldots, s_k)]\!] = [\mathtt{f}]([\![s_1]\!], \ldots, [\![s_k]\!])$

Prove: $[\![\ell]\!] > [\![r]\!]$ for all rules $\ell \to r$, all α

Then: $[\![s]\!] > [\![t]\!]$ whenever $s \to_{\mathcal{R}} t$.

Monotonic algebras: definition

Given: a set $\mathcal{A}$ with a well-founded ordering $>$ (for example: $\mathbb{N}$)

Choose: a function $[\mathtt{f}]$ from $\mathcal{A}^k$ to $\mathcal{A}$ for every $\mathtt{f}$ of arity k

Define: for a given α mapping variables to $\mathcal{A}$:

- $[\![x]\!] = \alpha(x)$
- $[\![\mathtt{f}(s_1, \ldots, s_k)]\!] = [\mathtt{f}]([\![s_1]\!], \ldots, [\![s_k]\!])$

Prove: $[\![\ell]\!] > [\![r]\!]$ for all rules $\ell \to r$, all α

Then: $[\![s]\!] > [\![t]\!]$ whenever $s \to_{\mathcal{R}} t$.

Therefore: $[\![s]\!] \geq \mathtt{derivationheight}(s)$.

Higher-order interpretations to $\mathbb{N}$: problems

Suppose: $[\![\mathtt{s}(x)]\!] = x + 1$

Higher-order interpretations to $\mathbb{N}$: problems

Suppose: $[\![\mathtt{s}(x)]\!] = x + 1$
Question: What is $[\![\mathtt{s}]\!]$?

Higher-order interpretations to $\mathbb{N}$: problems

Suppose: $[\![\mathtt{s}(x)]\!] = x + 1$
Question: What is $[\![\mathtt{s}]\!]$?

Problem: behaviour matters!

$$\begin{array}{rcl} \mathtt{fold}(F, x, []) & \rightarrow & [] \\ \mathtt{fold}(F, x, \mathtt{cons}(y, l)) & \rightarrow & \mathtt{fold}(F, (F \cdot x \cdot y), l) \end{array}$$

Higher-order interpretations to $\mathbb{N}$: problems

Suppose: $[\![\mathtt{s}(x)]\!] = x + 1$
Question: What is $[\![\mathtt{s}]\!]$?

Problem: behaviour matters!

$$\begin{array}{rcl} \mathtt{fold}(F, x, []) & \to & [] \\ \mathtt{fold}(F, x, \mathtt{cons}(y, l)) & \to & \mathtt{fold}(F, (F \cdot x \cdot y), l) \end{array}$$

- What is the derivation height if $F := \lambda x, y.\mathtt{minimum}(x, y)$?

Higher-order interpretations to $\mathbb{N}$: problems

Suppose: $[\![\mathtt{s}(x)]\!] = x + 1$
Question: What is $[\![\mathtt{s}]\!]$?

Problem: behaviour matters!

$$\begin{array}{rcl} \mathtt{fold}(F, x, []) & \to & [] \\ \mathtt{fold}(F, x, \mathtt{cons}(y, l)) & \to & \mathtt{fold}(F, (F \cdot x \cdot y), l) \end{array}$$

- What is the derivation height if $F := \lambda x, y.\mathtt{minimum}(x, y)$?
- What if: $F := \lambda x, y.\mathtt{add}(x, y)$?

Higher-order interpretations to $\mathbb{N}$: problems

Suppose: $[\![\mathtt{s}(x)]\!] = x + 1$
Question: What is $[\![\mathtt{s}]\!]$?

Problem: behaviour matters!

$$\begin{array}{rcl} \mathtt{fold}(F, x, []) & \to & [] \\ \mathtt{fold}(F, x, \mathtt{cons}(y, l)) & \to & \mathtt{fold}(F, (F \cdot x \cdot y), l) \end{array}$$

- What is the derivation height if $F := \lambda x, y.\mathtt{minimum}(x, y)$?
- What if: $F := \lambda x, y.\mathtt{add}(x, y)$?
- What if: $F := \lambda x, y.\mathtt{add}(x, x)$?

Proposal

Let's interpret terms of function type as functions!

Proposal

Let's interpret terms of function type as functions!

Type interpretations:

Proposal

Let's interpret terms of function type as functions!

Type interpretations:

- For every base type ι: a set $\mathcal{A}_\iota$, ordering $>_\iota$ and quasi-ordering $\geq_\iota$

Proposal

Let's interpret terms of function type as functions!

Type interpretations:

- For every base type ι: a set $\mathcal{A}_\iota$, ordering $>_\iota$ and quasi-ordering $\geq_\iota$
- Define:

$$\begin{array}{rcl} (\!|\iota|\!) & = & \mathcal{A}_\iota \\ (\!|\sigma \Rightarrow \tau|\!) & = & \text{“monotonic functions from } (\!|\sigma|\!) \text{ to } (\!|\tau|\!)\text{”} \end{array}$$

Proposal

Let's interpret terms of function type as functions!

Type interpretations:

- For every base type ι: a set $\mathcal{A}_\iota$, ordering $>_\iota$ and quasi-ordering $\geq_\iota$
- Define:

$$
\begin{array}{rcl}
(\!|\iota|\!) & = & \mathcal{A}_\iota \\
(\!|\sigma \Rightarrow \tau|\!) & = & \text{“monotonic functions from } (\!|\sigma|\!) \text{ to } (\!|\tau|\!)\text{”} \\
F >_{\sigma\Rightarrow\tau} G & \text{if} & F(a) >_\tau G(a) \text{ for all } a \in (\!|\sigma|\!) \\
F \geq_{\sigma\Rightarrow\tau} G & \text{if} & F(a) \geq_\tau G(a) \text{ for all } a \in (\!|\sigma|\!)
\end{array}
$$

Higher-order monotonic algebras: definition

Given: a set $\mathcal{A}$ with a well-founded ordering $>$ (for example: $\mathbb{N}$)

Choose: a function $[\mathtt{f}]$ from $\mathcal{A}^k$ to $\mathcal{A}$ for every $\mathtt{f}$ of arity k

Define: for a given α mapping variables to $\mathcal{A}$:

- $[\![x]\!] = \alpha(x)$
- $[\![\mathtt{f}(s_1, \dots, s_k)]\!] = [\mathtt{f}]([\![s_1]\!], \dots, [\![s_k]\!])$

Prove: $[\![\ell]\!] > [\![r]\!]$ for all rules $\ell \to r$, all α

Then: $[\![s]\!] > [\![t]\!]$ whenver $s \to_{\mathcal{R}} t$.

Therefore: $[\![s]\!] \geq \mathtt{derivationheight}(s)$.

Higher-order monotonic algebras: definition

Given: a set $\mathcal{A}$ with a well-founded ordering $>$ (for example: $\mathbb{N}$)

Choose: a function $[\mathtt{f}]$ in $(\!|\sigma|\!)$ for every $\mathtt{f}$ of type σ

Define: for a given α mapping variables to $\mathcal{A}$:

- $[\![x]\!] = \alpha(x)$
- $[\![\mathtt{f}]\!] = [\mathtt{f}]$
- $[\![s \cdot t]\!] = [\![s]\!]([\![t]\!])$

Prove: $[\![\ell]\!] > [\![r]\!]$ for all rules $\ell \to r$, all α

Then: $[\![s]\!] > [\![t]\!]$ whenver $s \to_{\mathcal{R}} t$.

Therefore: $[\![s]\!] \geq \mathtt{derivationheight}(s)$.

Example:

$$
\begin{array}{rcl}
[] & :: & \text{list} \\
\texttt{cons} & :: & \text{nat} \Rightarrow \text{list} \Rightarrow \text{list} \\
\texttt{map} & :: & (\text{nat} \Rightarrow \text{nat}) \Rightarrow \text{list} \Rightarrow \text{list}
\end{array}
$$

$$
\begin{array}{rcl}
\texttt{map}(F, []) & \to & [] \\
\texttt{map}(F, \texttt{cons}(x, l)) & \to & \texttt{cons}(F \cdot x, \texttt{map}(F, l))
\end{array}
$$

Example:

$$\begin{array}{rcl} [] & :: & \text{list} \\ \texttt{cons} & :: & \text{nat} \Rightarrow \text{list} \Rightarrow \text{list} \\ \texttt{map} & :: & (\text{nat} \Rightarrow \text{nat}) \Rightarrow \text{list} \Rightarrow \text{list} \end{array}$$

$$\begin{array}{rcl} \texttt{map}(F, []) & \rightarrow & [] \\ \texttt{map}(F, \texttt{cons}(x, l)) & \rightarrow & \texttt{cons}(F \cdot x, \texttt{map}(F, l)) \end{array}$$

Choose: $\mathcal{A}_\iota = \mathbb{N}$ for all ι

Example:

$$\begin{array}{rcl} [] & :: & \text{list} \\ \texttt{cons} & :: & \text{nat} \Rightarrow \text{list} \Rightarrow \text{list} \\ \texttt{map} & :: & (\text{nat} \Rightarrow \text{nat}) \Rightarrow \text{list} \Rightarrow \text{list} \end{array}$$

$$\begin{array}{rcl} \texttt{map}(F, []) & \to & [] \\ \texttt{map}(F, \texttt{cons}(x, l)) & \to & \texttt{cons}(F \cdot x, \texttt{map}(F, l)) \end{array}$$

Choose: $\mathcal{A}_\iota = \mathbb{N}$ for all ι

$$\begin{array}{rcl} [[]] & = & 0 \\ [\texttt{cons}](x, y) & = & x + y + 1 \\ [\texttt{map}](F, x) & = & (x + 1) * F(x) \end{array}$$

Example:

$$
\begin{array}{rcl}
[] & :: & \text{list} \\
\texttt{cons} & :: & \text{nat} \Rightarrow \text{list} \Rightarrow \text{list} \\
\texttt{map} & :: & (\text{nat} \Rightarrow \text{nat}) \Rightarrow \text{list} \Rightarrow \text{list}
\end{array}
$$

$$
\begin{array}{rcl}
\texttt{map}(F, []) & \rightarrow & [] \\
\texttt{map}(F, \texttt{cons}(x, l)) & \rightarrow & \texttt{cons}(F \cdot x, \texttt{map}(F, l))
\end{array}
$$

Choose: $\mathcal{A}_\iota = \mathbb{N}$ for all ι

$$
\begin{array}{rcl}
[[]] & = & 0 \\
[\texttt{cons}](x, y) & = & x + y + 1 \\
[\texttt{map}](F, x) & = & (x + 1) * F(x)
\end{array}
$$

Monotonicity:

Example:

$$\begin{array}{rcl} [] & :: & \text{list} \\ \texttt{cons} & :: & \text{nat} \Rightarrow \text{list} \Rightarrow \text{list} \\ \texttt{map} & :: & (\text{nat} \Rightarrow \text{nat}) \Rightarrow \text{list} \Rightarrow \text{list} \end{array}$$

$$\begin{array}{rcl} \texttt{map}(F, []) & \to & [] \\ \texttt{map}(F, \texttt{cons}(x, l)) & \to & \texttt{cons}(F \cdot x, \texttt{map}(F, l)) \end{array}$$

Choose: $\mathcal{A}_\iota = \mathbb{N}$ for all ι

$$\begin{array}{rcl} [[]] & = & 0 \\ [\texttt{cons}](x, y) & = & x + y + 1 \\ [\texttt{map}](F, x) & = & (x + 1) * F(x) \end{array}$$

Monotonicity: holds.

Example

$$\begin{array}{rcl} [\texttt{[]}] & = & 0 \\ [\texttt{cons}](x,y) & = & x+y+1 \\ [\texttt{map}](F,x) & = & (x+1)*F(x)+1 \end{array}$$

Goal 1:

$$[\![\texttt{map}(F,\texttt{[]})]\!] > [\![\texttt{[]}]\!]$$

Example

$$[[]] = 0$$
$$[\texttt{cons}](x, y) = x + y + 1$$
$$[\texttt{map}](F, x) = (x + 1) * F(x) + 1$$

Goal 1:

$$(0 + 1) * F(0) + 1 > 0$$

Example

$$\begin{aligned} [\texttt{[]}] &= 0 \\ [\texttt{cons}](x, y) &= x + y + 1 \\ [\texttt{map}](F, x) &= (x + 1) * F(x) + 1 \end{aligned}$$

Goal 2:

$$[\![\texttt{map}(F, \texttt{cons}(x, l))]\!] > [\![\texttt{cons}(F \cdot x, \texttt{map}(F, l))]\!]$$

Example

$$
\begin{aligned}
[[]] &= 0 \\
[\mathtt{cons}](x, y) &= x + y + 1 \\
[\mathtt{map}](F, x) &= (x + 1) * F(x) + 1
\end{aligned}
$$

Goal 2:

$$
((x + l + 1) + 1) * F(x + l + 1) + 1 \quad > \\
F(x) + ((l + 1) * F(l) + 1) + 1
$$

Example

$$
\begin{aligned}
[[]] &= 0 \\
[\mathtt{cons}](x, y) &= x + y + 1 \\
[\mathtt{map}](F, x) &= (x + 1) * F(x) + 1
\end{aligned}
$$

Goal 2:

$$x * F(x + l + 1) + l * F(x + l + 1) + F(x + l + 1) + F(x + l + 1) + 1 \quad >$$
$$F(x) + l * F(l) + F(l) + 1$$

Example

$$
\begin{array}{rcl}
[[]] & = & 0 \\
[\mathtt{cons}](x, y) & = & x + y + 1 \\
[\mathtt{map}](F, x) & = & (x + 1) * F(x) + 1
\end{array}
$$

Goal 2:

$$
\begin{array}{lllll}
x * F(x + l + 1) & + l * F(x + l + 1) & + F(x + l + 1) & + F(x + l + 1) & +1 \\
> & + l * F(l) & + F(x) & + F(l) & +1
\end{array}
$$

Exercise

Given:

$$\begin{array}{rcl} [] & :: & \text{list} \\ \texttt{cons} & :: & \text{nat} \Rightarrow \text{list} \Rightarrow \text{list} \\ \texttt{filter} & :: & (\text{nat} \Rightarrow \text{bool}) \Rightarrow \text{list} \Rightarrow \text{list} \\ \texttt{helper} & :: & \text{bool} \Rightarrow \text{nat} \Rightarrow \text{list} \Rightarrow \text{list} \end{array}$$

$$\begin{array}{rcl} \texttt{filter}(F, []) & \to & [] \\ \texttt{filter}(F, \texttt{cons}(x, l)) & \to & \texttt{helper}(F \cdot x, x, \texttt{filter}(F, l)) \\ \texttt{helper}(\texttt{true}, x, l) & \to & \texttt{cons}(x, l) \\ \texttt{helper}(\texttt{false}, x, l) & \to & l \end{array}$$

Task: show that the following interpretation suffices:

$$\begin{array}{rcl} [[]] & = & 0 \\ [\texttt{cons}](x, y) & = & x + y + 1 \\ [\texttt{helper}](b, x, y) & = & b + x + y + 1 \\ [\texttt{filter}](F, x) & = & (x + 1) * (F(x) + 1) \end{array}$$

$$\begin{array}{rcl} [\texttt{true}] & = & 1 \\ [\texttt{false}] & = & 0 \end{array}$$

Bonus exercise

Given:

$$
\begin{array}{rcl}
[] & :: & \text{list} \\
\texttt{cons} & :: & \text{nat} \Rightarrow \text{list} \Rightarrow \text{list} \\
\texttt{zip} & :: & (\text{nat} \Rightarrow \text{nat}) \Rightarrow \text{list} \Rightarrow \text{list}
\end{array}
$$

$$
\begin{array}{rcl}
\texttt{zip}(F, [], l) & = & l \\
\texttt{zip}(F, l, []) & = & l \\
\texttt{zip}(F, \texttt{cons}(x, l), \texttt{cons}(y, q)) & = & \texttt{cons}(F \cdot x \cdot y, \texttt{zip}(F, l, q))
\end{array}
$$

Task: find an interpretation that orients these rules!

Abstraction

Discussion: what should be the interpretation of $\lambda x.s$?

Abstraction

Discussion: what should be the interpretation of $\lambda x.s$?

Naive choice: $x \mapsto [\![x]\!]$

Abstraction

Discussion: what should be the interpretation of $\lambda x.s$?

Naive choice: $x \mapsto [\![x]\!]$

Problem: the naive interpretation for $\lambda x.\mathtt{0}$ is not monotonic!

Abstraction

Discussion: what should be the interpretation of $\lambda x.s$?

Naive choice: $x \mapsto [\![x]\!]$

Problem: the naive interpretation for $\lambda x.\mathtt{0}$ is not monotonic!

Solution: for each σ, τ, a function $\mathtt{makesm}_{\sigma,\tau}$:

Abstraction

Discussion: what should be the interpretation of $\lambda x.s$?

Naive choice: $x \mapsto [\![x]\!]$

Problem: the naive interpretation for $\lambda x.\mathtt{0}$ is not monotonic!

Solution: for each σ, τ, a function $\mathtt{makesm}_{\sigma,\tau}$:

- Input: a monotonic or constant function from $(\!|\sigma|\!)$ to $(\!|\tau|\!)$
- Output: a monotonic function from $(\!|\sigma|\!)$ to $(\!|\tau|\!)$

Abstraction

Discussion: what should be the interpretation of $\lambda x.s$?

Naive choice: $x \mapsto [\![x]\!]$

Problem: the naive interpretation for $\lambda x.\mathtt{0}$ is not monotonic!

Solution: for each σ, τ, a function $\mathtt{makesm}_{\sigma,\tau}$:

- Input: a monotonic or constant function from $(\!|\sigma|\!)$ to $(\!|\tau|\!)$
- Output: a monotonic function from $(\!|\sigma|\!)$ to $(\!|\tau|\!)$
- $\mathtt{makesm}_{\sigma,\tau}$ should itself be monotonic!

Abstraction

Discussion: what should be the interpretation of $\lambda x.s$?

Naive choice: $x \mapsto [\![x]\!]$

Problem: the naive interpretation for $\lambda x.\mathtt{0}$ is not monotonic!

Solution: for each σ, τ, a function $\mathtt{makesm}_{\sigma,\tau}$:

- Input: a monotonic or constant function from $(\!|\sigma|\!)$ to $(\!|\tau|\!)$
- Output: a monotonic function from $(\!|\sigma|\!)$ to $(\!|\tau|\!)$
- $\mathtt{makesm}_{\sigma,\tau}$ should itself be monotonic!
- we need to have $[\![(\lambda x.s) \cdot t]\!] > s[x := t]$

Abstraction

Discussion: what should be the interpretation of $\lambda x.s$?

Naive choice: $x \mapsto [\![x]\!]$

Problem: the naive interpretation for $\lambda x.\mathtt{0}$ is not monotonic!

Solution: for each σ, τ, a function $\mathtt{makesm}_{\sigma,\tau}$:

- Input: a monotonic or constant function from $(\!|\sigma|\!)$ to $(\!|\tau|\!)$
- Output: a monotonic function from $(\!|\sigma|\!)$ to $(\!|\tau|\!)$
- $\mathtt{makesm}_{\sigma,\tau}$ should itself be monotonic!
- we need to have $[\![(\lambda x.s) \cdot t]\!] > s[x := t]$

Example: (for $\sigma, \tau = \mathsf{nat}$ and $\mathcal{A}_{\mathsf{nat}} = \mathbb{N}$):

- if F is constant, then $\mathtt{makesm}_{\sigma,\tau}(F) = x \mapsto F(x) + x + 1$
- otherwise $\mathtt{makesm}_{\sigma,\tau}(F) = x \mapsto F(x) + 1$

An observation

Consider:

- $[\![\texttt{add}(\texttt{s}^n(\texttt{0}),\ \texttt{s}^m(\texttt{0}))]\!] = 1 + m + 2 * n$

An observation

Consider:

- $[\![\mathtt{add}(\mathtt{s}^n(\mathtt{0}),\ \mathtt{s}^m(\mathtt{0}))]\!] = 1 + m + 2 * n$
- actual cost of reduction: $n + 1$

An observation

Consider:

- $[\![\mathtt{add}(\mathtt{s}^n(\mathtt{0}),\ \mathtt{s}^m(\mathtt{0}))]\!] = 1 + m + 2 * n$
- actual cost of reduction: $n + 1$
- size of normal form: $n + m$

An observation

Consider:

- $[\![\texttt{add}(\texttt{s}^n(\texttt{0}),\ \texttt{s}^m(\texttt{0}))]\!] = 1 + m + 2 * n$
- actual cost of reduction: $n + 1$
- size of normal form: $n + m$
- $\Rightarrow$ interpretation = cost + size?

An observation

Consider:

- $[\![\mathtt{add}(\mathtt{s}^n(\mathtt{0}),\ \mathtt{s}^m(\mathtt{0}))]\!] = 1 + m + 2 * n$
- actual cost of reduction: $n + 1$
- size of normal form: $n + m$
- $\Rightarrow$ interpretation = cost + size?

Idea: separate cost and size already in the interpretation!

An observation

Consider:

- $[\![\mathtt{add}(\mathtt{s}^n(\mathtt{0}),\ \mathtt{s}^m(\mathtt{0}))]\!] = 1 + m + 2 * n$
- actual cost of reduction: $n + 1$
- size of normal form: $n + m$
- $\Rightarrow$ interpretation = cost + size?

Idea: separate cost and size already in the interpretation!

Mechanism: map to $\mathbb{N}^2$ instead of $\mathbb{N}$.

We let $\langle x, y\rangle > \langle x', y'\rangle$ if $x > x'$ and $y \geq y'$.

Separating cost and size

$$\begin{array}{rcl} \mathtt{add}(\mathtt{0}, y) & \to & y \\ \mathtt{add}(\mathtt{s}(x), y) & \to & \mathtt{s}(\mathtt{add}(x, y)) \end{array}$$

Let:

		cost		**size**	
$[\![\mathtt{0}]\!]$	= ⟨	0	,	0	⟩
$[\![\mathtt{s}(x)]\!]$	= ⟨	x_{cost}	,	$x_{size} + 1$	⟩
$[\![\mathtt{add}(x, y)]\!]$	= ⟨	$x_{cost} + y_{cost} + x_{size}$	,	$x_{size} + y_{size}$	⟩

Separating cost and size

$$\begin{array}{rcl} \mathtt{add}(\mathtt{0}, y) & \to & y \\ \mathtt{add}(\mathtt{s}(x), y) & \to & \mathtt{s}(\mathtt{add}(x, y)) \end{array}$$

Let:

		cost		**size**	
$[\![\mathtt{0}]\!]$	= $\langle$	0	,	0	$\rangle$
$[\![\mathtt{s}(x)]\!]$	= $\langle$	x_{cost}	,	$x_{size} + 1$	$\rangle$
$[\![\mathtt{add}(x, y)]\!]$	= $\langle$	$x_{cost} + y_{cost} + x_{size}$	,	$x_{size} + y_{size}$	$\rangle$

Then:

$$\begin{array}{rcccl} [\![\mathtt{add}(\mathtt{0}, y)]\!] & = & \langle 1 + y_1, y_2 \rangle & & \\ & > & \langle y_1, y_2 \rangle & = & [\![y]\!] \\ [\![\mathtt{add}(\mathtt{s}(x), y)]\!] & = & \langle 2 + x_1 + y_1 + x_2, 1 + x_2 + y_2 \rangle & & \\ & > & \langle 1 + x_1 + y_1 + x_2, 1 + x_2 + y_2 \rangle & = & [\![\mathtt{s}(\mathtt{add}(x, y))]\!] \end{array}$$

Separating cost and size

$$\begin{array}{rcl} \mathtt{add}(\mathtt{0}, y) & \to & y \\ \mathtt{add}(\mathtt{s}(x), y) & \to & \mathtt{s}(\mathtt{add}(x, y)) \end{array}$$

Let:

		cost		**size**	
$[\![\mathtt{0}]\!]$	= $\langle$	0	,	0	$\rangle$
$[\![\mathtt{s}(x)]\!]$	= $\langle$	x_{cost}	,	$x_{size} + 1$	$\rangle$
$[\![\mathtt{add}(x, y)]\!]$	= $\langle$	$x_{cost} + y_{cost} + x_{size}$	,	$x_{size} + y_{size}$	$\rangle$

Then:

$$\begin{array}{rcll} [\![\mathtt{add}(\mathtt{0}, y)]\!] & = & \langle 1 + y_1, y_2 \rangle & \\ & > & \langle y_1, y_2 \rangle & = [\![y]\!] \\ [\![\mathtt{add}(\mathtt{s}(x), y)]\!] & = & \langle 2 + x_1 + y_1 + x_2, 1 + x_2 + y_2 \rangle & \\ & > & \langle 1 + x_1 + y_1 + x_2, 1 + x_2 + y_2 \rangle & = [\![\mathtt{s}(\mathtt{add}(x, y))]\!] \end{array}$$

Hence: $[\![\mathtt{add}(\mathtt{s}^n(\mathtt{0}), \ \mathtt{s}^m(\mathtt{0}))]\!] = \langle 1 + n, n + m \rangle$: precise!

When interpretations to $\mathbb{N}$ are Not Great

$$\mathtt{a}(\mathtt{b}(x)) \quad \to \quad \mathtt{b}(\mathtt{a}(x))$$

When interpretations to $\mathbb{N}$ are Not Great

$$\mathtt{a}(\mathtt{b}(x)) \quad \to \quad \mathtt{b}(\mathtt{a}(x))$$

Let:

- $[\![\mathtt{a}(x)]\!] = 2 * x$
- $[\![\mathtt{b}(x)]\!] = x + 1$
- $[\![\epsilon]\!] = 0$

When interpretations to $\mathbb{N}$ are Not Great

$$\mathtt{a}(\mathtt{b}(x)) \;\rightarrow\; \mathtt{b}(\mathtt{a}(x))$$

Let:

- $[\![\mathtt{a}(x)]\!] = 2 * x$
- $[\![\mathtt{b}(x)]\!] = x + 1$
- $[\![\epsilon]\!] = 0$

Then:

$$[\![\mathtt{a}(\mathtt{b}(x))]\!] \;=\; 2 + 2 * x > 1 + 2 * x \;=\; [\![\mathtt{b}(\mathtt{a}(x))]\!]$$

When interpretations to $\mathbb{N}$ are Not Great

$$\mathtt{a}(\mathtt{b}(x)) \quad \to \quad \mathtt{b}(\mathtt{a}(x))$$

Let:

- $[\![\mathtt{a}(x)]\!] = 2 * x$
- $[\![\mathtt{b}(x)]\!] = x + 1$
- $[\![\epsilon]\!] = 0$

Then:

$$[\![\mathtt{a}(\mathtt{b}(x))]\!] \quad = \quad 2 + 2 * x > 1 + 2 * x \quad = \quad [\![\mathtt{b}(\mathtt{a}(x))]\!]$$

Hence: $[\![\mathtt{a}^n(\mathtt{b}^m(\epsilon))]\!] = 2^n * m$: exponential!

Separating cost and size

$$\mathtt{a}(\mathtt{b}(x)) \;\to\; \mathtt{b}(\mathtt{a}(x))$$

Let:

		cost		**size**	
$[\![\mathtt{a}(x)]\!]$	$= \langle$	$x_{cost} + x_{size}$	,	x_{size}	$\rangle$
$[\![\mathtt{b}(x)]\!]$	$= \langle$	x_{cost}	,	$x_{size} + 1$	$\rangle$
$[\![\epsilon]\!]$	$= \langle$	0	,	0	$\rangle$

Separating cost and size

$$\mathtt{a}(\mathtt{b}(x)) \quad\to\quad \mathtt{b}(\mathtt{a}(x))$$

Let:

		cost		**size**	
$[\![\mathtt{a}(x)]\!]$	$= \langle$	$x_{cost} + x_{size}$	,	x_{size}	$\rangle$
$[\![\mathtt{b}(x)]\!]$	$= \langle$	x_{cost}	,	$x_{size} + 1$	$\rangle$
$[\![\epsilon]\!]$	$= \langle$	0	,	0	$\rangle$

Then:

$$[\![\mathtt{a}(\mathtt{b}(x))]\!] \;=\; \langle x_1 + x_2 + 1, x_2 + 1\rangle > \langle x_1 + x_2, x_2 + 1\rangle \;=\; [\![\mathtt{b}(\mathtt{a}(x))]\!]$$

Separating cost and size

$$a(b(x)) \rightarrow b(a(x))$$

Let:

		cost		size	
$[\![a(x)]\!]$	$= \langle$	$x_{cost} + x_{size}$	,	x_{size}	$\rangle$
$[\![b(x)]\!]$	$= \langle$	x_{cost}	,	$x_{size} + 1$	$\rangle$
$[\![\epsilon]\!]$	$= \langle$	0	,	0	$\rangle$

Then:

$$[\![a(b(x))]\!] = \langle x_1 + x_2 + 1, x_2 + 1 \rangle > \langle x_1 + x_2, x_2 + 1 \rangle = [\![b(a(x))]\!]$$

Hence: $[\![a^n(b^m(\epsilon))]\!] = (n * m, m)$: precise!

Tuple interpretations

Definition:

Tuple interpretations

Definition: monotonic algebras with $\mathcal{A}_\iota = \mathbb{N}^{K[\iota]}$ for all ι

Tuple interpretations

Definition: monotonic algebras with $\mathcal{A}_\iota = \mathbb{N}^{K[\iota]}$ for all ι

$\Longrightarrow$ both for first- and higher-order!

Tuple interpretations

Definition: monotonic algebras with $\mathcal{A}_\iota = \mathbb{N}^{K[\iota]}$ for all ι

$\Longrightarrow$ both for first- and higher-order!

Example sort interpretations:

Tuple interpretations

Definition: monotonic algebras with $\mathcal{A}_\iota = \mathbb{N}^{K[\iota]}$ for all ι

$\Longrightarrow$ both for first- and higher-order!

Example sort interpretations:

- $\{\mathbf{N}\} = \mathbb{N}^2$ (cost, size of normal form)

Tuple interpretations

Definition: monotonic algebras with $\mathcal{A}_\iota = \mathbb{N}^{K[\iota]}$ for all ι

$\Longrightarrow$ both for first- and higher-order!

Example sort interpretations:

- $\{\textbf{N}\} = \mathbb{N}^2$ (cost, size of normal form)
- $\{\text{list}\} = \mathbb{N}^3$ (cost, list length, size of greatest element)

Tuple interpretations

Definition: monotonic algebras with $\mathcal{A}_\iota = \mathbb{N}^{K[\iota]}$ for all ι

$\Longrightarrow$ both for first- and higher-order!

Example sort interpretations:

- $\{\mathbf{N}\} = \mathbb{N}^2$ (cost, size of normal form)
- $\{\text{list}\} = \mathbb{N}^3$ (cost, list length, size of greatest element)
- $\{\text{bool}\} = \mathbb{N}^1$ (cost)

Example: interpreting list functions

$$
\begin{aligned}
\mathtt{append}([], l) &\rightarrow l \\
\mathtt{append}(\mathtt{cons}(x, l), q) &\rightarrow \mathtt{cons}(x, \mathtt{append}(l, q)) \\
\mathtt{sum}([]) &\rightarrow \mathtt{0} \\
\mathtt{sum}(\mathtt{cons}(x, l)) &\rightarrow \mathtt{add}(x, \mathtt{sum}(l))
\end{aligned}
$$

Example: interpreting list functions

$$\begin{aligned}
\mathtt{append}([], l) &\rightarrow l \\
\mathtt{append}(\mathtt{cons}(x, l), q) &\rightarrow \mathtt{cons}(x, \mathtt{append}(l, q)) \\
\mathtt{sum}([]) &\rightarrow \mathtt{0} \\
\mathtt{sum}(\mathtt{cons}(x, l)) &\rightarrow \mathtt{add}(x, \mathtt{sum}(l))
\end{aligned}$$

Interpretations:

- $\{\text{list}\} = \mathbb{N}^3$ (cost, list length, maximum element)

Example: interpreting list functions

$$\begin{array}{rcl} \mathtt{append}([], l) & \rightarrow & l \\ \mathtt{append}(\mathtt{cons}(x, l), q) & \rightarrow & \mathtt{cons}(x, \mathtt{append}(l, q)) \\ \mathtt{sum}([]) & \rightarrow & \mathtt{0} \\ \mathtt{sum}(\mathtt{cons}(x, l)) & \rightarrow & \mathtt{add}(x, \mathtt{sum}(l)) \end{array}$$

Interpretations:

- $\{\text{list}\} = \mathbb{N}^3$ (cost, list length, maximum element)
- $[\![[]]\!] =$

Example: interpreting list functions

$$\begin{array}{rcl} \mathtt{append}([], l) & \to & l \\ \mathtt{append}(\mathtt{cons}(x, l), q) & \to & \mathtt{cons}(x, \mathtt{append}(l, q)) \\ \mathtt{sum}([]) & \to & \mathtt{0} \\ \mathtt{sum}(\mathtt{cons}(x, l)) & \to & \mathtt{add}(x, \mathtt{sum}(l)) \end{array}$$

Interpretations:

- $\{$list$\} = \mathbb{N}^3$ (cost, list length, maximum element)
- $[\![[]]\!] = \langle 0, 0, 0 \rangle$

Example: interpreting list functions

$$\begin{array}{rcl} \mathtt{append}([], l) & \to & l \\ \mathtt{append}(\mathtt{cons}(x, l), q) & \to & \mathtt{cons}(x, \mathtt{append}(l, q)) \\ \mathtt{sum}([]) & \to & \mathtt{0} \\ \mathtt{sum}(\mathtt{cons}(x, l)) & \to & \mathtt{add}(x, \mathtt{sum}(l)) \end{array}$$

Interpretations:

- $\{\text{list}\} = \mathbb{N}^3$ (cost, list length, maximum element)
- $[\![[]]\!] = \langle 0, 0, 0 \rangle$
- $[\![\mathtt{cons}(x, l)]\!] =$

Example: interpreting list functions

$$\begin{array}{rcl} \mathtt{append}([], l) & \to & l \\ \mathtt{append}(\mathtt{cons}(x, l), q) & \to & \mathtt{cons}(x, \mathtt{append}(l, q)) \\ \mathtt{sum}([]) & \to & \mathtt{0} \\ \mathtt{sum}(\mathtt{cons}(x, l)) & \to & \mathtt{add}(x, \mathtt{sum}(l)) \end{array}$$

Interpretations:

- $\{\text{list}\} = \mathbb{N}^3$ (cost, list length, maximum element)
- $[\![[]]\!] = \langle 0, 0, 0 \rangle$
- $[\![\mathtt{cons}(x, l)]\!] = \langle x_{cost} + l_{cost}, l_{len} + 1, \text{max}(x_{size}, l_{max}) \rangle$

Example: interpreting list functions

$$\begin{array}{rcl} \mathtt{append}([], l) & \to & l \\ \mathtt{append}(\mathtt{cons}(x, l), q) & \to & \mathtt{cons}(x, \mathtt{append}(l, q)) \\ \mathtt{sum}([]) & \to & \mathtt{0} \\ \mathtt{sum}(\mathtt{cons}(x, l)) & \to & \mathtt{add}(x, \mathtt{sum}(l)) \end{array}$$

Interpretations:

- $\{\text{list}\} = \mathbb{N}^3$ (cost, list length, maximum element)
- $[\![[]]\!] = \langle 0, 0, 0 \rangle$
- $[\![\mathtt{cons}(x, l)]\!] = \langle x_{cost} + l_{cost}, l_{len} + 1, \max(x_{size}, l_{max}) \rangle$
- $[\![\mathtt{append}(l, q)]\!] = \langle \text{cost}, \text{length}, \text{maximum} \rangle$, where:
  - maximum =
  - length =
  - cost =

Example: interpreting list functions

$$\begin{array}{rcl} \mathtt{append}([], l) & \to & l \\ \mathtt{append}(\mathtt{cons}(x, l), q) & \to & \mathtt{cons}(x, \mathtt{append}(l, q)) \\ \mathtt{sum}([]) & \to & \mathtt{0} \\ \mathtt{sum}(\mathtt{cons}(x, l)) & \to & \mathtt{add}(x, \mathtt{sum}(l)) \end{array}$$

Interpretations:

- $\{\text{list}\} = \mathbb{N}^3$ (cost, list length, maximum element)
- $[\![[]]\!] = \langle 0, 0, 0 \rangle$
- $[\![\mathtt{cons}(x, l)]\!] = \langle x_{cost} + l_{cost}, l_{len} + 1, \max(x_{size}, l_{max}) \rangle$
- $[\![\mathtt{append}(l, q)]\!] = \langle \text{cost}, \text{length}, \text{maximum} \rangle$, where:
  - maximum = $\max(l_{max}, q_{max})$
  - length =
  - cost =

Example: interpreting list functions

$$\begin{aligned} \mathtt{append}([], l) &\to l \\ \mathtt{append}(\mathtt{cons}(x, l), q) &\to \mathtt{cons}(x, \mathtt{append}(l, q)) \\ \mathtt{sum}([]) &\to 0 \\ \mathtt{sum}(\mathtt{cons}(x, l)) &\to \mathtt{add}(x, \mathtt{sum}(l)) \end{aligned}$$

Interpretations:

- $\{\text{list}\} = \mathbb{N}^3$ (cost, list length, maximum element)
- $[\![[]]\!] = \langle 0, 0, 0 \rangle$
- $[\![\mathtt{cons}(x, l)]\!] = \langle x_{cost} + l_{cost}, l_{len} + 1, \max(x_{size}, l_{max}) \rangle$
- $[\![\mathtt{append}(l, q)]\!] = \langle \text{cost}, \text{length}, \text{maximum} \rangle$, where:
  - maximum = $\max(l_{max}, q_{max})$
  - length = $l_{len} + q_{len}$
  - cost =

Example: interpreting list functions

$$\begin{aligned}
\mathtt{append}([], l) &\to l \\
\mathtt{append}(\mathtt{cons}(x, l), q) &\to \mathtt{cons}(x, \mathtt{append}(l, q)) \\
\mathtt{sum}([]) &\to \mathtt{0} \\
\mathtt{sum}(\mathtt{cons}(x, l)) &\to \mathtt{add}(x, \mathtt{sum}(l))
\end{aligned}$$

Interpretations:

- $\{\text{list}\} = \mathbb{N}^3$ (cost, list length, maximum element)
- $[\![[]]\!] = \langle 0, 0, 0 \rangle$
- $[\![\mathtt{cons}(x, l)]\!] = \langle x_{cost} + l_{cost}, l_{len} + 1, \text{max}(x_{size}, l_{max}) \rangle$
- $[\![\mathtt{append}(l, q)]\!] = \langle \text{cost}, \text{length}, \text{maximum} \rangle$, where:
  - maximum = $\text{max}(l_{max}, q_{max})$
  - length = $l_{len} + q_{len}$
  - cost = $l_{cost} + q_{cost} + l_{len} + 1$

Example: interpreting list functions

$$
\begin{aligned}
\mathtt{append}([], l) &\to l \\
\mathtt{append}(\mathtt{cons}(x, l), q) &\to \mathtt{cons}(x, \mathtt{append}(l, q)) \\
\mathtt{sum}([]) &\to \mathtt{0} \\
\mathtt{sum}(\mathtt{cons}(x, l)) &\to \mathtt{add}(x, \mathtt{sum}(l))
\end{aligned}
$$

Interpretations:

- $\{\text{list}\} = \mathbb{N}^3$ (cost, list length, maximum element)
- $[\![[]]\!] = \langle 0, 0, 0 \rangle$
- $[\![\mathtt{cons}(x, l)]\!] = \langle x_{cost} + l_{cost}, l_{len} + 1, \max(x_{size}, l_{max}) \rangle$
- $[\![\mathtt{append}(l, q)]\!] = \langle \text{cost}, \text{length}, \text{maximum} \rangle$, where:
  - maximum = $\max(l_{max}, q_{max})$
  - length = $l_{len} + q_{len}$
  - cost = $l_{cost} + q_{cost} + l_{len} + 1$
- $[\![\mathtt{sum}(l)]\!] = \langle \text{cost}, \text{size} \rangle$, where:
  - size =
  - cost =

Example: interpreting list functions

$$\begin{array}{rcl} \mathtt{append}([], l) & \rightarrow & l \\ \mathtt{append}(\mathtt{cons}(x, l), q) & \rightarrow & \mathtt{cons}(x, \mathtt{append}(l, q)) \\ \mathtt{sum}([]) & \rightarrow & \mathtt{0} \\ \mathtt{sum}(\mathtt{cons}(x, l)) & \rightarrow & \mathtt{add}(x, \mathtt{sum}(l)) \end{array}$$

Interpretations:

- $\{\text{list}\} = \mathbb{N}^3$ (cost, list length, maximum element)
- $[\![[]]\!] = \langle 0, 0, 0 \rangle$
- $[\![\mathtt{cons}(x, l)]\!] = \langle x_{cost} + l_{cost}, l_{len} + 1, \max(x_{size}, l_{max}) \rangle$
- $[\![\mathtt{append}(l, q)]\!] = \langle \text{cost}, \text{length}, \text{maximum} \rangle$, where:
  - maximum = $\max(l_{max}, q_{max})$
  - length = $l_{len} + q_{len}$
  - cost = $l_{cost} + q_{cost} + l_{len} + 1$
- $[\![\mathtt{sum}(l)]\!] = \langle \text{cost}, \text{size} \rangle$, where:
  - size = $l_{len} * l_{max}$
  - cost =

Example: interpreting list functions

$$\begin{array}{rcl} \mathtt{append}([], l) & \to & l \\ \mathtt{append}(\mathtt{cons}(x, l), q) & \to & \mathtt{cons}(x, \mathtt{append}(l, q)) \\ \mathtt{sum}([]) & \to & \mathtt{0} \\ \mathtt{sum}(\mathtt{cons}(x, l)) & \to & \mathtt{add}(x, \mathtt{sum}(l)) \end{array}$$

Interpretations:

- $\{\text{list}\} = \mathbb{N}^3$ (cost, list length, maximum element)
- $[\![[]]\!] = \langle 0, 0, 0 \rangle$
- $[\![\mathtt{cons}(x, l)]\!] = \langle x_{cost} + l_{cost}, l_{len} + 1, \max(x_{size}, l_{max}) \rangle$
- $[\![\mathtt{append}(l, q)]\!] = \langle \text{cost}, \text{length}, \text{maximum} \rangle$, where:
  - maximum = $\max(l_{max}, q_{max})$
  - length = $l_{len} + q_{len}$
  - cost = $l_{cost} + q_{cost} + l_{len} + 1$
- $[\![\mathtt{sum}(l)]\!] = \langle \text{cost}, \text{size} \rangle$, where:
  - size = $l_{len} * l_{max}$
  - cost = $l_{cost} + 2 * l_{len} + l_{len} * l_{max} + 1$

Higher-order tuple interpretations: an example

$$
\begin{array}{rcl}
[] & :: & \mathsf{list} \\
\mathtt{cons} & :: & \mathbf{N} \Rightarrow \mathsf{list} \Rightarrow \mathsf{list} \\
\mathtt{map} & :: & (\mathbf{N} \Rightarrow \mathbf{N}) \Rightarrow \mathsf{list} \Rightarrow \mathsf{list}
\end{array}
$$

$$
\begin{array}{rcl}
\mathtt{map}(F, []) & \to & [] \\
\mathtt{map}(F, \mathtt{cons}(x, l)) & \to & \mathtt{cons}(F \cdot x, \mathtt{map}(F, l))
\end{array}
$$

Higher-order tuple interpretations: an example

$$\begin{array}{rcl} [] & :: & \text{list} \\ \mathtt{cons} & :: & \mathbf{N} \Rightarrow \text{list} \Rightarrow \text{list} \\ \mathtt{map} & :: & (\mathbf{N} \Rightarrow \mathbf{N}) \Rightarrow \text{list} \Rightarrow \text{list} \end{array}$$

$$\begin{array}{rcl} \mathtt{map}(F, []) & \rightarrow & [] \\ \mathtt{map}(F, \mathtt{cons}(x, l)) & \rightarrow & \mathtt{cons}(F \cdot x, \mathtt{map}(F, l)) \end{array}$$

Let:

- $[\![[]]\!] = \langle 0, 0, 0 \rangle$
- $[\![\mathtt{cons}(x, l)]\!] = \langle x_{cost} + l_{cost}, l_{len} + 1, \max(x_{size}, l_{max}) \rangle$
- $[\![\mathtt{map}(F, l)]\!] = \langle \text{cost}, \text{length}, \text{maximum} \rangle$, where:
  - length:
  - maximum:
  - cost:

Higher-order tuple interpretations: an example

$$\begin{array}{rcl} [] & :: & \mathsf{list} \\ \mathtt{cons} & :: & \mathbf{N} \Rightarrow \mathsf{list} \Rightarrow \mathsf{list} \\ \mathtt{map} & :: & (\mathbf{N} \Rightarrow \mathbf{N}) \Rightarrow \mathsf{list} \Rightarrow \mathsf{list} \end{array}$$

$$\begin{array}{rcl} \mathtt{map}(F, []) & \to & [] \\ \mathtt{map}(F, \mathtt{cons}(x, l)) & \to & \mathtt{cons}(F \cdot x, \mathtt{map}(F, l)) \end{array}$$

Let:

- $[\![[]]\!] = \langle 0, 0, 0 \rangle$
- $[\![\mathtt{cons}(x, l)]\!] = \langle x_{cost} + l_{cost}, l_{len} + 1, \max(x_{size}, l_{max}) \rangle$
- $[\![\mathtt{map}(F, l)]\!] = \langle \text{cost}, \text{length}, \text{maximum} \rangle$, where:
  - length: l_{len}
  - maximum:
  - cost:

Higher-order tuple interpretations: an example

$$\begin{array}{rcl} [] & :: & \text{list} \\ \texttt{cons} & :: & \mathbf{N} \Rightarrow \text{list} \Rightarrow \text{list} \\ \texttt{map} & :: & (\mathbf{N} \Rightarrow \mathbf{N}) \Rightarrow \text{list} \Rightarrow \text{list} \end{array}$$

$$\begin{array}{rcl} \texttt{map}(F, []) & \to & [] \\ \texttt{map}(F, \texttt{cons}(x, l)) & \to & \texttt{cons}(F \cdot x, \texttt{map}(F, l)) \end{array}$$

Let:

- $[\![[]]\!] = \langle 0, 0, 0 \rangle$
- $[\![\texttt{cons}(x, l)]\!] = \langle x_{cost} + l_{cost}, l_{len} + 1, \max(x_{size}, l_{max}) \rangle$
- $[\![\texttt{map}(F, l)]\!] = \langle \text{cost}, \text{length}, \text{maximum} \rangle$, where:
  - length: l_{len}
  - maximum: $F(\langle l_{cost}, l_{max} \rangle)_s$
  - cost:

Higher-order tuple interpretations: an example

$$\begin{array}{rcl} [] & :: & \text{list} \\ \texttt{cons} & :: & \mathbf{N} \Rightarrow \text{list} \Rightarrow \text{list} \\ \texttt{map} & :: & (\mathbf{N} \Rightarrow \mathbf{N}) \Rightarrow \text{list} \Rightarrow \text{list} \end{array}$$

$$\begin{array}{rcl} \texttt{map}(F, []) & \to & [] \\ \texttt{map}(F, \texttt{cons}(x, l)) & \to & \texttt{cons}(F \cdot x, \texttt{map}(F, l)) \end{array}$$

Let:

- $[\![[]]\!] = \langle 0, 0, 0 \rangle$
- $[\![\texttt{cons}(x, l)]\!] = \langle x_{cost} + l_{cost}, l_{len} + 1, \max(x_{size}, l_{max}) \rangle$
- $[\![\texttt{map}(F, l)]\!] = \langle \text{cost}, \text{length}, \text{maximum} \rangle$, where:
  - length: l_{len}
  - maximum: $F(\langle l_{cost}, l_{max} \rangle)_s$
  - cost: $(l_{len} + 1) * (F(\langle l_{cost}, l_{max} \rangle)_{cost} + 1)$

Higher-order tuple interpretations: an example

Goal: $[\![\mathtt{map}(F, \mathtt{cons}(x, l))]\!] > [\![\mathtt{cons}(F \cdot x, \mathtt{map}(F, l))]\!]$

Higher-order tuple interpretations: an example

Goal: $[\![\mathtt{map}(F, \mathtt{cons}(x, l))]\!] > [\![\mathtt{cons}(F \cdot x, \mathtt{map}(F, l))]\!]$

$$
\begin{array}{ll}
\langle & (q+2) * (F(\ \langle x_1 + l, \mathsf{max}(x_2, l_3)\rangle\ )_1 + 1), \\
& q + 1 \\
& F(\ \langle x_1 + l,\ \mathsf{max}(x_2, l_3)\rangle\ )_2 \\
\rangle & > \\
\langle & F(x)_1 + (q+1) * (F(\langle l, l_3\rangle)_1 + 1), \\
& q + 1 \\
& \mathsf{max}(F(x)_2, F(\langle l, l_3\rangle)_2) \\
\rangle &
\end{array}
$$

Higher-order tuple interpretations: an example

Goal: $[\![\mathtt{map}(F, \mathtt{cons}(x, l))]\!] > [\![\mathtt{cons}(F \cdot x, \mathtt{map}(F, l))]\!]$

$$(q + 2) * (F(\ \langle x_1 + l, \max(x_2, l_3)\rangle\ )_1 + 1) \quad > \\ F(x)_1 + (q + 1) * (F(\langle l, l_3\rangle)_1 + 1)$$

$$q + 1 \quad \geq \\ q + 1$$

$$F(\ \langle x_1 + l,\ \max(x_2, l_3)\rangle\ )_2 \quad \geq \\ \max(F(x)_2, F(\langle l, l_3\rangle)_2)$$

Higher-order tuple interpretations: an example

Goal: $[\![\mathtt{map}(F, \mathtt{cons}(x, l))]\!] > [\![\mathtt{cons}(F \cdot x, \mathtt{map}(F, l))]\!]$

$$(q + 2) * (F(\ \langle x_1 + l, \max(x_2, l_3)\rangle\ )_1 + 1) \quad > \\ F(x)_1 + (q + 1) * (F(\langle l, l_3\rangle)_1 + 1)$$

$$q + 1 \quad \geq \\ q + 1$$

$$F(\ \langle x_1 + l,\ \max(x_2, l_3)\rangle\ )_2 \quad \geq \\ \max(F(x)_2, F(\langle l, l_3\rangle)_2)$$

Higher-order tuple interpretations: an example

Goal: $[\![\mathtt{map}(F, \mathtt{cons}(x, l))]\!] > [\![\mathtt{cons}(F \cdot x, \mathtt{map}(F, l))]\!]$

$$(q+2) * (F(\langle x_1 + l, \max(x_2, l_3)\rangle)_1 + 1) \quad > \quad F(x)_1 + (q+1) * (F(\langle l, l_3\rangle)_1 + 1)$$

$$F(\langle x_1 + l, \max(x_2, l_3)\rangle)_2 \quad \geq \quad \max(F(x)_2, F(\langle l, l_3\rangle)_2)$$

Higher-order tuple interpretations: an example

Goal: $[\![\mathtt{map}(F, \mathtt{cons}(x, l))]\!] > [\![\mathtt{cons}(F \cdot x, \mathtt{map}(F, l))]\!]$

$$(q+2) * (F(\ \langle x_1 + l, \mathsf{max}(x_2, l_3)\rangle\ )_1 + 1) \quad > \\ F(x)_1 + (q+1) * (F(\langle l, l_3\rangle)_1 + 1)$$

$$F(\ \langle x_1 + l,\ \mathsf{max}(x_2, l_3)\rangle\ )_2 \quad \geq \\ F(x)_2$$

$$F(\ \langle x_1 + l,\ \mathsf{max}(x_2, l_3)\rangle\ )_2 \quad \geq \\ F(\langle l, l_3\rangle)_2$$

Higher-order tuple interpretations: an example

Goal: $[\![\mathtt{map}(F, \mathtt{cons}(x, l))]\!] > [\![\mathtt{cons}(F \cdot x, \mathtt{map}(F, l))]\!]$

$$(q + 2) * (F(\ \langle x_1 + l, \max(x_2, l_3)\rangle\ )_1 + 1) \quad > \\ F(x)_1 + (q + 1) * (F(\langle l, l_3\rangle)_1 + 1)$$

$$F(\ \langle x_1 + l,\ \max(x_2, l_3)\rangle\ )_2 \quad \geq \\ F(\ \langle x_1 \quad , \qquad x_2 \quad \rangle\ )_2$$

$$F(\ \langle x_1 + l,\ \max(x_2, l_3)\rangle\ )_2 \quad \geq \\ F(\langle l, l_3\rangle)_2$$

Higher-order tuple interpretations: an example

Goal: $[\![\mathtt{map}(F, \mathtt{cons}(x, l))]\!] > [\![\mathtt{cons}(F \cdot x, \mathtt{map}(F, l))]\!]$

$$(q + 2) * (F(\ \langle x_1 + l, \max(x_2, l_3)\rangle\ )_1 + 1) \quad > \\ F(x)_1 + (q + 1) * (F(\langle l, l_3\rangle)_1 + 1)$$

$$F(\ \langle x_1 + l,\ \max(x_2, l_3)\rangle\ )_2 \quad \geq \\ F(\ \langle x_1 \quad , \quad x_2 \quad \rangle\ )_2$$

$$F(\ \langle x_1 + l,\ \max(x_2, l_3)\rangle\ )_2 \quad \geq \\ F(\ \langle \quad l, \quad l_3\ \rangle\ )_2$$

Higher-order tuple interpretations: an example

Goal: $[\![\mathtt{map}(F, \mathtt{cons}(x, l))]\!] > [\![\mathtt{cons}(F \cdot x, \mathtt{map}(F, l))]\!]$

$$(q+2) * (F(\ \langle x_1 + l, \max(x_2, l_3)\rangle\ )_1 + 1) \quad > \\ F(x)_1 + (q+1) * (F(\langle l, l_3\rangle)_1 + 1)$$

Higher-order tuple interpretations: an example

Goal: $[\![\mathtt{map}(F, \mathtt{cons}(x, l))]\!] > [\![\mathtt{cons}(F \cdot x, \mathtt{map}(F, l))]\!]$

$$
\begin{aligned}
& F(\ \langle x_1 + l, \mathsf{max}(x_2, l_3)\rangle\ )_1 + 1 + \\
& (q+1) * (F(\ \langle x_1 + l, \mathsf{max}(x_2, l_3)\rangle\ )_1 + 1) \quad > \\
& F(x)_1 + \\
& (q+1) * (F(\langle l, l_3\rangle)_1 + 1)
\end{aligned}
$$

Higher-order tuple interpretations: an example

Goal: $[\![\mathtt{map}(F, \mathtt{cons}(x, l))]\!] > [\![\mathtt{cons}(F \cdot x, \mathtt{map}(F, l))]\!]$

$$
\begin{aligned}
& F(\ \langle x_1 + l, \max(x_2, l_3)\rangle\ )_1 + 1 + \\
& (q+1) * (F(\ \langle x_1 + l, \max(x_2, l_3)\rangle\ )_1 + 1) \quad > \\
& F(x)_1 + \\
& (q+1) * (F(\langle l, l_3\rangle)_1 + 1)
\end{aligned}
$$

Higher-order tuple interpretations: an example

Goal: $[\![\texttt{map}(F, \texttt{cons}(x, l))]\!] > [\![\texttt{cons}(F \cdot x, \texttt{map}(F, l))]\!]$

$$\begin{array}{l} F(\ \langle x_1 + l, \max(x_2, l_3)\rangle\ )_1\ + \\ (q + 1) * (F(\ \langle x_1 + l, \max(x_2, l_3)\rangle\ )_1 + 1) \quad \geq \\ F(x)_1\ + \\ (q + 1) * (F(\langle l, l_3\rangle)_1 + 1) \end{array}$$

Higher-order tuple interpretations: an example

Goal: $[\![\mathtt{map}(F, \mathtt{cons}(x, l))]\!] > [\![\mathtt{cons}(F \cdot x, \mathtt{map}(F, l))]\!]$

$$F(\ \langle x_1 + l, \max(x_2, l_3)\rangle\ )_1 \geq F(x)_1$$

$$(q+1) * (F(\ \langle x_1 + l, \max(x_2, l_3)\rangle\ )_1 + 1) \geq (q+1) * (F(\langle l, l_3\rangle)_1 + 1)$$

Higher-order tuple interpretations: an example

Goal: $[\![\texttt{map}(F, \texttt{cons}(x, l))]\!] > [\![\texttt{cons}(F \cdot x, \texttt{map}(F, l))]\!]$

$$F(\ \langle x_1 + l, \max(x_2, l_3)\rangle\ )_1 \geq$$
$$F(\ \langle x_1 \quad , \quad x_2 \quad \rangle\ )_1$$

$$(q+1) * (F(\ \langle x_1 + l, \max(x_2, l_3)\rangle\ )_1 + 1) \geq$$
$$(q+1) * (F(\ \langle \quad l, \quad l_3\ \rangle\ )_1 + 1)$$

Exercise

1. Find an interpretation, with $(\!|\text{nat}|\!) = \mathbb{N}^2$, for the following system:

$$\begin{array}{rcl} \mathtt{minus}(x, \mathtt{0}) & \to & x \\ \mathtt{minus}(\mathtt{s}(x), \mathtt{s}(y)) & \to & \mathtt{minus}(x, y) \\ \mathtt{quot}(\mathtt{0}, \mathtt{s}(y)) & \to & \mathtt{0} \\ \mathtt{quot}(\mathtt{s}(x), \mathtt{s}(y)) & \to & \mathtt{s}(\mathtt{quot}(\mathtt{minus}(x, y), \mathtt{s}(y))) \end{array}$$

   Warning: do not take $x_{size} - y_{size}$ for the size of $\mathtt{minus}(x, y)$!

2. Find an interpretation for the following HTRS, where $\mathtt{zip} :: (\text{nat} \Rightarrow \text{nat}) \Rightarrow \text{list} \Rightarrow \text{list}$.

$$\begin{array}{rcl} \mathtt{zip}(F, [], l) & = & l \\ \mathtt{zip}(F, l, []) & = & l \\ \mathtt{zip}(F, \mathtt{cons}(x, l), \mathtt{cons}(y, q)) & = & \mathtt{cons}(F \cdot x \cdot y, \mathtt{zip}(F, l, q)) \end{array}$$

A more challenging higher-order tuple interpretation

$$
\begin{aligned}
\mathtt{fold}(F, x, []) &\rightarrow [] \\
\mathtt{fold}(F, x, \mathtt{cons}(y, l)) &\rightarrow \mathtt{fold}(F, (F \cdot x \cdot y), l)
\end{aligned}
$$

A more challenging higher-order tuple interpretation

$$\begin{array}{rcl} \texttt{fold}(F, x, []) & \to & [] \\ \texttt{fold}(F, x, \texttt{cons}(y, l)) & \to & \texttt{fold}(F, (F \cdot x \cdot y), l) \end{array}$$

Interpretation:

$$[\![\texttt{fold}(F, x, l)]\!] = \langle \text{cost}, \ \text{size} \rangle$$

Where:

- cost = $1 + l_{cost} + F(\langle 0, 0 \rangle)_{cost} + Helper[F, \langle l_{cost}, l_{max} \rangle]^{l_{len}}(x)_{cost}$
- size = $Helper[F, \langle l_{cost}, l_{max} \rangle]^{l_{len}}(x)_{size}$
- And $Helper[F, y] = x \mapsto \langle F(x, y)_{cost}, \max(x_{size}, F(x, y)_{size}) \rangle$.

A more challenging higher-order tuple interpretation.

$$
\begin{array}{rcl}
\mathtt{add}(0, y) & \to & y \\
\mathtt{add}(\mathtt{s}(x), y) & \to & \mathtt{add}(x, \mathtt{s}(y)) \\
\mathtt{fold}(F, x, []) & \to & [] \\
\mathtt{fold}(F, x, \mathtt{cons}(y, l)) & \to & \mathtt{fold}(F, (F \cdot x \cdot y), l) \\
\mathtt{sum}(l) & \to & \mathtt{fold}(\lambda x.\lambda y.\mathtt{add}(x, y), 0, l)
\end{array}
$$

Method: Plug $[\![\lambda x.\lambda y.\mathtt{add}(x, y)]\!]$ into the interpretation for $\mathtt{fold}$.

A more challenging higher-order tuple interpretation.

$$\begin{array}{rcl}
\mathtt{add}(\mathtt{0}, y) & \to & y \\
\mathtt{add}(\mathtt{s}(x), y) & \to & \mathtt{add}(x, \mathtt{s}(y)) \\
\mathtt{fold}(F, x, []) & \to & [] \\
\mathtt{fold}(F, x, \mathtt{cons}(y, l)) & \to & \mathtt{fold}(F, (F \cdot x \cdot y), l) \\
\mathtt{sum}(l) & \to & \mathtt{fold}(\lambda x.\lambda y.\mathtt{add}(x, y), \mathtt{0}, l)
\end{array}$$

Method: Plug $[\![\lambda x.\lambda y.\mathtt{add}(x, y)]\!]$ into the interpretation for $\mathtt{fold}$.

Interpreting λ: use $\mathtt{makesm}_{\iota,\sigma_1\Rightarrow\ldots\Rightarrow\sigma_m\Rightarrow\kappa} =$

$$\begin{cases}
(F, x, y_1, \ldots, y_m) \mapsto (F(x,\vec{y})_1 + 1 + x_1, F(x,\vec{y})_2, \ldots, F(x,\vec{y})_{K[\kappa]}) & \text{if } F \text{ is constant} \\
(F, x, y_1, \ldots, y_m) \mapsto (F(x,\vec{y})_1 + 1, F(x,\vec{y})_2, \ldots, F(x,\vec{y})_{K[\kappa]}) & \text{if } F \text{ is monotonic}
\end{cases}$$

Derivational and runtime complexity (first-order)

Derivational and runtime complexity (first-order)

Derivational complexity:

$n \mapsto$ “maximum derivation height for a term of size n”

Derivational and runtime complexity (first-order)

Derivational complexity:

$n \mapsto$ "maximum derivation height for a term of size n"

Downside: can easily get large; e.g.:

$\mathtt{mul}(\mathtt{mul}(\mathtt{mul}(\mathtt{mul}(\mathtt{s}(\mathtt{s}(\mathtt{0})), \mathtt{s}(\mathtt{s}(\mathtt{0}))), \mathtt{s}(\mathtt{s}(\mathtt{0}))), \mathtt{s}(\mathtt{s}(\mathtt{0}))), \mathtt{s}(\mathtt{s}(\mathtt{0})))$

Derivational and runtime complexity (first-order)

Derivational complexity:

$n \mapsto$ "maximum derivation height for a term of size n"

Downside: can easily get large; e.g.:

$\mathtt{mul}(\mathtt{mul}(\mathtt{mul}(\mathtt{mul}(\mathtt{s}(\mathtt{s}(\mathtt{0})), \mathtt{s}(\mathtt{s}(\mathtt{0}))), \mathtt{s}(\mathtt{s}(\mathtt{0}))), \mathtt{s}(\mathtt{s}(\mathtt{0}))), \mathtt{s}(\mathtt{s}(\mathtt{0})))$

Runtime complexity:

$n \mapsto$ "maximum derivation height for a basic term of size n"

Derivational and runtime complexity (first-order)

Derivational complexity:

$n \mapsto$ "maximum derivation height for a term of size n"

Downside: can easily get large; e.g.:

`mul(mul(mul(mul(s(s(0)), s(s(0))), s(s(0))), s(s(0))), s(s(0)))`

Runtime complexity:

$n \mapsto$ "maximum derivation height for a basic term of size n"

Basic term: `function(data, ..., data)`

Derivational and runtime complexity (first-order)

Derivational complexity:

$n \mapsto$ "maximum derivation height for a term of size n"

Downside: can easily get large; e.g.:

$\texttt{mul}(\texttt{mul}(\texttt{mul}(\texttt{mul}(\texttt{s}(\texttt{s}(\texttt{0})), \texttt{s}(\texttt{s}(\texttt{0}))), \texttt{s}(\texttt{s}(\texttt{0}))), \texttt{s}(\texttt{s}(\texttt{0}))), \texttt{s}(\texttt{s}(\texttt{0})))$

Runtime complexity:

$n \mapsto$ "maximum derivation height for a basic term of size n"

Basic term: $\texttt{function}(\texttt{data}, \ldots, \texttt{data})$

Example: $\texttt{mul}(\texttt{s}(\texttt{s}(\texttt{s}(\texttt{s}(\texttt{s}(\texttt{0}))))), \texttt{s}(\texttt{s}(\texttt{s}(\texttt{s}(\texttt{s}(\texttt{s}(\texttt{s}(\texttt{0}))))))))$

Derivational and runtime complexity (first-order)

Derivational complexity:

$n \mapsto$ "maximum derivation height for a term of size n"

Downside: can easily get large; e.g.:

`mul(mul(mul(mul(s(s(0)), s(s(0))), s(s(0))), s(s(0))), s(s(0)))`

Runtime complexity:

$n \mapsto$ "maximum derivation height for a basic term of size n"

Basic term: `function(data, ..., data)`

Example: `mul(s(s(s(s(s(0))))), s(s(s(s(s(s(s(0))))))))`

Connection with computational complexity:

Derivational and runtime complexity (first-order)

Derivational complexity:

$n \mapsto$ "maximum derivation height for a term of size n"

Downside: can easily get large; e.g.:

$\mathtt{mul}(\mathtt{mul}(\mathtt{mul}(\mathtt{mul}(\mathtt{s}(\mathtt{s}(\mathtt{0})), \mathtt{s}(\mathtt{s}(\mathtt{0}))), \mathtt{s}(\mathtt{s}(\mathtt{0}))), \mathtt{s}(\mathtt{s}(\mathtt{0}))), \mathtt{s}(\mathtt{s}(\mathtt{0})))$

Runtime complexity:

$n \mapsto$ "maximum derivation height for a basic term of size n"

Basic term: $\mathtt{function}(\mathtt{data}, \dots, \mathtt{data})$

Example: $\mathtt{mul}(\mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{0}))))), \mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{0}))))))))$

Connection with computational complexity: depends

Termination (and complexity) competition

Complexity Analysis

Derivational_Complexity: TRS 41499

1. AProVE (**UP:742**, LOW:914, TIME:5d 14:51:28)
2. tct-trs_v3.2.0_2020-06-28 (UP:645, LOW:0, TIME:3d 23:25:46)

Derivational_Complexity: TRS Innermost 41500

1. AProVE (**UP:1530**, LOW:2070, TIME:8d 10:19:16)
2. tct-trs_v3.2.0_2020-06-28 (UP:636, LOW:0, TIME:6d 01:37:44)

Runtime_Complexity: TRS 41508

1. AProVE (**UP:665**, LOW:1782, TIME:1d 07:43:25)
2. tct-trs_v3.2.0_2020-06-28 (UP:380, LOW:1103, TIME:2d 00:28:55)

Runtime_Complexity: TRS Innermost 41507

1. AProVE (**UP:672**, LOW:1238, TIME:1d 03:51:23)
2. tct-trs_v3.2.0_2020-06-28 (UP:444, LOW:777, TIME:1d 08:04:34)

Runtime_Complexity: TRS Innermost Certified 41509

1. tct-trs_v3.2.0_2020-06-28 (**UP:419**, LOW:0, TIME:1d 01:02:42, Certification:00:00:39)
2. AProVE (UP:400, LOW:0, TIME:18:40:07, Certification:00:00:57)

Complexity of higher-order term rewriting

Open question: do derivational and runtime complexity even make sense for higher-order rewriting?

Complexity of higher-order term rewriting

Open question: do derivational and runtime complexity even make sense for higher-order rewriting?

$$\begin{array}{rcl} \texttt{fold}(F, x, []) & \rightarrow & [] \\ \texttt{fold}(F, x, \texttt{cons}(y, l)) & \rightarrow & \texttt{fold}(F, (F \cdot x \cdot y), l) \end{array}$$

Recall:

- What if: $F := \lambda x, y.\texttt{minimum}(x, y)$?
- What if: $F := \lambda x, y.\texttt{add}(x, y)$?
- What if: $F := \lambda x, y.\texttt{add}(x, x)$?

Higher-order derivational complexity?

Idea:

Higher-order derivational complexity?

Idea: naively extend the definition of derivational complexity

Higher-order derivational complexity?

Idea: naively extend the definition of derivational complexity

Result:

$$\begin{aligned} \mathtt{add}(x, \mathtt{0}) &\rightarrow x \\ \mathtt{add}(x, \mathtt{s}(y)) &\rightarrow \mathtt{s}(\mathtt{add}(x, y)) \end{aligned}$$

Higher-order derivational complexity?

Idea: naively extend the definition of derivational complexity

Result:

$$\begin{aligned} \mathtt{add}(x, \mathtt{0}) &\to x \\ \mathtt{add}(x, \mathtt{s}(y)) &\to \mathtt{s}(\mathtt{add}(x, y)) \end{aligned}$$

- $(\lambda x.\mathtt{add}(x, x)) \cdot (\mathtt{s}(\mathtt{s}(\mathtt{0})))$
- $(\lambda x.\mathtt{add}(x, x)) \cdot ((\lambda x.\mathtt{add}(x, x)) \cdot (\mathtt{s}(\mathtt{s}(\mathtt{0}))))$
- $(\lambda x.\mathtt{add}(x, x)) \cdot ((\lambda x.\mathtt{add}(x, x)) \cdot ((\lambda x.\mathtt{add}(x, x)) \cdot (\mathtt{s}(\mathtt{s}(\mathtt{0})))))$
- ...

Higher-order derivational complexity?

Idea: naively extend the definition of derivational complexity

Result:

$$\mathtt{add}(x, \mathtt{0}) \rightarrow x$$
$$\mathtt{add}(x, \mathtt{s}(y)) \rightarrow \mathtt{s}(\mathtt{add}(x, y))$$

- $(\lambda x.\mathtt{add}(x,x)) \cdot (\mathtt{s}(\mathtt{s}(\mathtt{0})))$
- $(\lambda x.\mathtt{add}(x,x)) \cdot ((\lambda x.\mathtt{add}(x,x)) \cdot (\mathtt{s}(\mathtt{s}(\mathtt{0}))))$
- $(\lambda x.\mathtt{add}(x,x)) \cdot ((\lambda x.\mathtt{add}(x,x)) \cdot ((\lambda x.\mathtt{add}(x,x)) \cdot (\mathtt{s}(\mathtt{s}(\mathtt{0})))))$
- ...

Conclusion: exponential complexity at a minimum

Runtime complexity: a simple extension

Runtime complexity:

$n \mapsto$ “maximum derivation height for a basic term of size n”

Basic term: $\texttt{function}(\texttt{data}, \ldots, \texttt{data})$

Runtime complexity: a simple extension

Runtime complexity:
$n \mapsto$ "maximum derivation height for a basic term of size n"
Basic term: `function`(`data`$, \dots,$ `data`)

Question: is it interesting to look at λ-functions over constructors?

Runtime complexity: a simple extension

Runtime complexity:

$n \mapsto$ "maximum derivation height for a basic term of size n"

Basic term: $\mathtt{function}(\mathtt{data}, \dots, \mathtt{data})$

Question: is it interesting to look at λ-functions over constructors?

- $\mathtt{map}(\lambda x.\mathtt{s}(x),$ some lst$)$?

Runtime complexity: a simple extension

Runtime complexity:
$n \mapsto$ "maximum derivation height for a basic term of size n"
Basic term: $\texttt{function}(\texttt{data}, \ldots, \texttt{data})$

Question: is it interesting to look at λ-functions over constructors?

- $\texttt{map}(\lambda x.\texttt{s}(x)$, some lst)?
- $\texttt{f}(\lambda x.\texttt{cons}(x, \texttt{cons}(x, [])))$, some data)?

Runtime complexity: a simple extension

Runtime complexity:
$n \mapsto$ "maximum derivation height for a basic term of size n"
Basic term: $\mathtt{function}(\mathtt{data}, \dots, \mathtt{data})$

Question: is it interesting to look at λ-functions over constructors?

- $\mathtt{map}(\lambda x.\mathtt{s}(x)$, some lst)?
- $\mathtt{f}(\lambda x.\mathtt{cons}(x, \mathtt{cons}(x, [])))$, some data)?

Choice: data must be a first-order term.

Higher-order runtime complexity example

$$
\begin{aligned}
\mathtt{add}(\mathtt{0}, y) &\to y \\
\mathtt{add}(\mathtt{s}(x), y) &\to \mathtt{add}(x, \mathtt{s}(y)) \\
\mathtt{fold}(F, x, []) &\to [] \\
\mathtt{fold}(F, x, \mathtt{cons}(y, l)) &\to \mathtt{fold}(F, (F \cdot x \cdot y), l) \\
\mathtt{sum}(l) &\to \mathtt{fold}(\lambda x.\lambda y.\mathtt{add}(x, y), \mathtt{0}, l)
\end{aligned}
$$

Higher-order runtime complexity example

$$
\begin{array}{rcl}
\mathtt{add}(0, y) & \to & y \\
\mathtt{add}(\mathtt{s}(x), y) & \to & \mathtt{add}(x, \mathtt{s}(y)) \\
\mathtt{fold}(F, x, []) & \to & [] \\
\mathtt{fold}(F, x, \mathtt{cons}(y, l)) & \to & \mathtt{fold}(F, (F \cdot x \cdot y), l) \\
\mathtt{sum}(l) & \to & \mathtt{fold}(\lambda x.\lambda y.\mathtt{add}(x, y), 0, l)
\end{array}
$$

Basic terms:

- $\mathtt{add}(\mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{s}(0))))), \mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{s}(0))))))))$
- $\mathtt{sum}(\mathtt{cons}(\mathtt{s}(\mathtt{s}(0)), \mathtt{cons}(0, \mathtt{cons}(\mathtt{s}(\mathtt{s}(\mathtt{s}(0))), []))))$

Higher-order runtime complexity example

$$\begin{array}{rcl}
\mathtt{add}(\mathtt{0}, y) & \to & y \\
\mathtt{add}(\mathtt{s}(x), y) & \to & \mathtt{add}(x, \mathtt{s}(y)) \\
\mathtt{fold}(F, x, []) & \to & [] \\
\mathtt{fold}(F, x, \mathtt{cons}(y, l)) & \to & \mathtt{fold}(F, (F \cdot x \cdot y), l) \\
\mathtt{sum}(l) & \to & \mathtt{fold}(\lambda x.\lambda y.\mathtt{add}(x, y), \mathtt{0}, l)
\end{array}$$

Basic terms:

- $\mathtt{add}(\mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{0}))))), \mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{0}))))))))$
- $\mathtt{sum}(\mathtt{cons}(\mathtt{s}(\mathtt{s}(\mathtt{0})), \mathtt{cons}(\mathtt{0}, \mathtt{cons}(\mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{0}))), []))))$

Runtime complexity: $n \mapsto \mathcal{O}(n^2)$

Higher-order runtime complexity example

$$
\begin{array}{rcl}
\mathtt{add}(\mathtt{0}, y) & \to & y \\
\mathtt{add}(\mathtt{s}(x), y) & \to & \mathtt{add}(x, \mathtt{s}(y)) \\
\mathtt{fold}(F, x, []) & \to & [] \\
\mathtt{fold}(F, x, \mathtt{cons}(y, l)) & \to & \mathtt{fold}(F, (F \cdot x \cdot y), l) \\
\mathtt{sum}(l) & \to & \mathtt{fold}(\lambda x.\lambda y.\mathtt{add}(x, y), \mathtt{0}, l)
\end{array}
$$

Basic terms:

- $\mathtt{add}(\mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{0}))))), \mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{0}))))))))$
- $\mathtt{sum}(\mathtt{cons}(\mathtt{s}(\mathtt{s}(\mathtt{0})), \mathtt{cons}(\mathtt{0}, \mathtt{cons}(\mathtt{s}(\mathtt{s}(\mathtt{s}(\mathtt{0}))), []))))$

Runtime complexity: $n \mapsto \mathcal{O}(n^2)$ (actually: length * max)

Exercises

1. Compute the runtime complexity of the following system.

$$
\begin{array}{rcl}
\mathtt{map}(F, []) & \to & [] \\
\mathtt{map}(F, \mathtt{cons}(x, l)) & \to & \mathtt{cons}(F \cdot x, \mathtt{map}(F, l)) \\
\mathtt{doublemap}(l) & \to & \mathtt{map}(\mathtt{double}, l) \\
\mathtt{double}(\mathtt{0}) & \to & \mathtt{0} \\
\mathtt{double}(\mathtt{s}(x)) & \to & \mathtt{s}(\mathtt{s}(\mathtt{double}(x)))
\end{array}
$$

2. Compute the runtime complexity of the following system.

$$
\begin{array}{rcl}
\mathtt{add}(x, \mathtt{0}) & \to & x \\
\mathtt{add}(x, \mathtt{s}(y)) & \to & \mathtt{s}(\mathtt{add}(x, y)) \\
\mathtt{zip}(F, [], l) & = & l \\
\mathtt{zip}(F, l, []) & = & l \\
\mathtt{zip}(F, \mathtt{cons}(x, l), \mathtt{cons}(y, q)) & = & \mathtt{cons}(F \cdot x \cdot y, \mathtt{zip}(F, l, q)) \\
\mathtt{zipadd}(l, q) & \to & \mathtt{zip}(\lambda x.\lambda y.\mathtt{add}(y, x), l, q)
\end{array}
$$

Higher-order complexity notion?

Idea:

- complexity of `map` is $\mathcal{O}(n * F(n))$?

Higher-order complexity notion?

Idea:

- complexity of `map` is $\mathcal{O}(n * F(n))$?
- complexity of `fold` is $\mathcal{O}(F^n(n))$?

Basic Feasible Functions

Idea:

Basic Feasible Functions

Idea:

- Oracle Turing Machines: these take n functions, k binary words

Basic Feasible Functions

Idea:

- Oracle Turing Machines: these take n functions, k binary words
- to compute function i:
  - copy input to tape i
  - go to special state
  - output is written on tape $n + i$

Basic Feasible Functions

Idea:

- Oracle Turing Machines: these take n functions, k binary words
- to compute function i:
  - copy input to tape i
  - go to special state
  - output is written on tape $n+i$
- $\Longrightarrow$ function cost is assumed zero, but function output size is important

Basic Feasible Functions

Idea:

- Oracle Turing Machines: these take n functions, k binary words
- to compute function i:
  - copy input to tape i
  - go to special state
  - output is written on tape $n+i$
- $\Longrightarrow$ function cost is assumed zero, but function output size is important
- Question: is the execution time limited by a higher-order polynomial over $F_1, \dots, F_n, w_1, \dots, w_k$?

Basic Feasible Functions

Idea:

- Oracle Turing Machines: these take n functions, k binary words
- to compute function i:
  - copy input to tape i
  - go to special state
  - output is written on tape $n + i$
- $\Longrightarrow$ function cost is assumed zero, but function output size is important
- Question: is the execution time limited by a higher-order polynomial over $F_1, \ldots, F_n, w_1, \ldots, w_k$?

Relevance: determined by polynomial tuple interpretation!