# Parametricity-Based Reasoning about Efficiency

D. Seidel* J. Voigtländer

University of Bonn

June 8th, 2011

*Supported by the DFG under grant VO 1512/1-1.

# Parametric Polymorphism in Haskell

A standard function:

$$\begin{aligned}
&\texttt{map} :: (\alpha \to \beta) \to [\alpha] \to [\beta] \\
&\texttt{map}\ g\ [\,] \qquad\quad = [\,] \\
&\texttt{map}\ g\ (a : as) = (g\ a) : (\texttt{map}\ g\ as)
\end{aligned}$$

Some invocations:

$$\begin{aligned}
&\texttt{map succ}\ [1,2,3] \qquad = [2,3,4] \\
&\texttt{map not}\ \ [\text{True}, \text{False}] = [\text{False}, \text{True}] \\
&\texttt{map even}\ [1,2,3] \qquad = [\text{False}, \text{True}, \text{False}] \\
&\texttt{map not}\ \ [1,2,3] \qquad \lightning \text{ rejected at compile-time}
\end{aligned}$$

# Parametric Polymorphism in Haskell

A standard function:

$$\texttt{map} :: (\alpha \to \beta) \to [\alpha] \to [\beta]$$

Some invocations:

$\texttt{map succ}\ [1,2,3] = [2,3,4]$

$\texttt{map not}\ [\text{True},\text{False}] = [\text{False},\text{True}]$

$\texttt{map even}\ [1,2,3] = [\text{False},\text{True},\text{False}]$

$\texttt{map not}\ [1,2,3]$ ↯ rejected at compile-time

# Another Example

$$
\begin{aligned}
&\texttt{reverse} :: [\alpha] \to [\alpha] \\
&\texttt{reverse}\ [\,] \qquad\quad = [\,] \\
&\texttt{reverse}\ (a : as) = (\texttt{reverse}\ as) +\!\!+ [a]
\end{aligned}
$$

# Another Example

$$
\begin{aligned}
&\texttt{reverse} :: [\alpha] \to [\alpha] \\
&\texttt{reverse}\ [\,] \qquad\ = [\,] \\
&\texttt{reverse}\ (a : as) = (\texttt{reverse}\ as) +\!\!+ [a]
\end{aligned}
$$

For every choice of $g$ and $l$:

$$\texttt{reverse}\ (\texttt{map}\ g\ l) \ = \ \texttt{map}\ g\ (\texttt{reverse}\ l)$$

Provable by induction.

# Another Example

$$\begin{aligned}
&\texttt{reverse} :: [\alpha] \to [\alpha] \\
&\texttt{reverse}\ [\,] \qquad\ = [\,] \\
&\texttt{reverse}\ (a : as) = (\texttt{reverse}\ as) +\!\!+ [a]
\end{aligned}$$

For every choice of $g$ and $l$:

$$\texttt{reverse}\ (\texttt{map}\ g\ l) \ = \ \texttt{map}\ g\ (\texttt{reverse}\ l)$$

Provable by induction.

Or as a "free theorem" [Wadler, FPCA'89].

# Another Example

$$\texttt{reverse} :: [\alpha] \to [\alpha]$$

For every choice of $g$ and $l$:

$$\texttt{reverse}\ (\texttt{map}\ g\ l) \;=\; \texttt{map}\ g\ (\texttt{reverse}\ l)$$

Provable by induction.

Or as a "free theorem" [Wadler, FPCA'89].

# Another Example

$$\texttt{reverse} :: [\alpha] \to [\alpha]$$

$$\texttt{tail} :: [\alpha] \to [\alpha]$$

For every choice of $g$ and $l$:

$$\texttt{reverse}\ (\texttt{map}\ g\ l) \;=\; \texttt{map}\ g\ (\texttt{reverse}\ l)$$

$$\texttt{tail}\ (\texttt{map}\ g\ l) \;=\; \texttt{map}\ g\ (\texttt{tail}\ l)$$

# Another Example

$$\texttt{reverse} :: [\alpha] \to [\alpha]$$

$$\texttt{tail} :: [\alpha] \to [\alpha]$$

$$\texttt{f} :: [\alpha] \to [\alpha]$$

For every choice of $g$ and $l$:

$$\texttt{reverse}\ (\texttt{map}\ g\ l) = \texttt{map}\ g\ (\texttt{reverse}\ l)$$

$$\texttt{tail}\ (\texttt{map}\ g\ l) = \texttt{map}\ g\ (\texttt{tail}\ l)$$

$$\texttt{f}\ (\texttt{map}\ g\ l) = \texttt{map}\ g\ (\texttt{f}\ l)$$

# Applications

- Short Cut Fusion [Gill et al., FPCA'93]

# Applications

- Short Cut Fusion [Gill et al., FPCA'93]
- The Dual of Short Cut Fusion [Svenningsson, ICFP'02]

# Applications

- Short Cut Fusion [Gill et al., FPCA'93]
- The Dual of Short Cut Fusion
  [Svenningsson, ICFP'02]
- Circular Short Cut Fusion
  [Fernandes et al., Haskell'07]
- ...

# Applications

- Short Cut Fusion [Gill et al., FPCA'93]
- The Dual of Short Cut Fusion
  [Svenningsson, ICFP'02]
- Circular Short Cut Fusion
  [Fernandes et al., Haskell'07]
- ...
- Testing polymorphic properties
  [Bernardy et al., ESOP'10]
- ...

# Free Theorems, Formally — In a Nutshell

Syntax for a typed $\lambda$-calculus:

$$\tau ::= \alpha \mid \mathsf{Nat} \mid \tau \to \tau \mid \ldots$$
$$t ::= x \mid n \mid t + t \mid \lambda x :: \tau . t \mid t\ t \mid \ldots$$

# Free Theorems, Formally — In a Nutshell

Syntax for a typed $\lambda$-calculus:

$$\tau ::= \alpha \mid \mathsf{Nat} \mid \tau \to \tau \mid \ldots$$
$$t ::= x \mid n \mid t + t \mid \lambda x :: \tau . t \mid t\ t \mid \ldots$$

Semantics:

$$\begin{aligned}
[\![\alpha]\!]_\theta &= \theta(\alpha) \\
[\![\mathsf{Nat}]\!]_\theta &= \mathbb{N} \\
[\![\tau_1 \to \tau_2]\!]_\theta &= [\![\tau_2]\!]_\theta^{[\![\tau_1]\!]_\theta} \\
\theta &\in \mathsf{Set}^{\mathsf{TVar}}
\end{aligned}$$

# Free Theorems, Formally — In a Nutshell

Syntax for a typed $\lambda$-calculus:

$$\tau ::= \alpha \mid \mathsf{Nat} \mid \tau \to \tau \mid \ldots$$
$$t ::= x \mid n \mid t + t \mid \lambda x :: \tau . t \mid t\ t \mid \ldots$$

Semantics:

$$\begin{array}{lll} [\![\alpha]\!]_\theta & = \theta(\alpha) & [\![x]\!]_\sigma = \sigma(x) \\ [\![\mathsf{Nat}]\!]_\theta & = \mathbb{N} & [\![n]\!]_\sigma = \mathbf{n} \\ [\![\tau_1 \to \tau_2]\!]_\theta & = [\![\tau_2]\!]_\theta^{[\![\tau_1]\!]_\theta} & [\![\lambda x :: \tau . t]\!]_\sigma = \lambda \mathbf{v} . [\![t]\!]_{\sigma[x \mapsto \mathbf{v}]} \\ \theta & \in \mathsf{Set}^{\mathsf{TVar}} & [\![t_1\ t_2]\!]_\sigma = [\![t_1]\!]_\sigma\ [\![t_2]\!]_\sigma \end{array}$$

# Free Theorems, Formally — In a Nutshell

Syntax for a typed $\lambda$-calculus:

$$\tau ::= \alpha \mid \mathsf{Nat} \mid \tau \to \tau \mid \ldots$$
$$t ::= x \mid n \mid t + t \mid \lambda x :: \tau . t \mid t\ t \mid \ldots$$

Semantics:

$$\begin{array}{llll} [\![\alpha]\!]_\theta & = \theta(\alpha) & [\![x]\!]_\sigma & = \sigma(x) \\ [\![\mathsf{Nat}]\!]_\theta & = \mathbb{N} & [\![n]\!]_\sigma & = \mathbf{n} \\ [\![\tau_1 \to \tau_2]\!]_\theta & = [\![\tau_2]\!]_\theta^{[\![\tau_1]\!]_\theta} & [\![\lambda x :: \tau . t]\!]_\sigma & = \lambda \mathbf{v} . [\![t]\!]_{\sigma[x \mapsto \mathbf{v}]} \\ \theta & \in \mathsf{Set}^{\mathsf{TVar}} & [\![t_1\ t_2]\!]_\sigma & = [\![t_1]\!]_\sigma\ [\![t_2]\!]_\sigma \end{array}$$

Logical relation:

$$\Delta_{\alpha,\rho} = \rho(\alpha) \qquad \Delta_{\mathsf{Nat},\rho} = id_{\mathbb{N}}$$
$$\Delta_{\tau_1 \to \tau_2,\rho} = \{(\mathbf{f}, \mathbf{g}) \mid \forall (\mathbf{x}, \mathbf{y}) \in \Delta_{\tau_1,\rho}.\ (\mathbf{f}\ \mathbf{x}, \mathbf{g}\ \mathbf{y}) \in \Delta_{\tau_2,\rho}\}$$

with $\rho \in \mathsf{Rel}^{\mathsf{TVar}}$

# Free Theorems, Formally — In a Nutshell

Syntax for a typed $\lambda$-calculus:

$$\tau ::= \alpha \mid \mathsf{Nat} \mid \tau \to \tau \mid \ldots$$
$$t ::= x \mid n \mid t + t \mid \lambda x :: \tau . t \mid t\ t \mid \ldots$$

Semantics:

$$\begin{array}{lll} [\![\alpha]\!]_\theta & = \theta(\alpha) & [\![x]\!]_\sigma = \sigma(x) \\ [\![\mathsf{Nat}]\!]_\theta & = \mathbb{N} & [\![n]\!]_\sigma = \mathbf{n} \\ [\![\tau_1 \to \tau_2]\!]_\theta & = [\![\tau_2]\!]_\theta^{[\![\tau_1]\!]_\theta} & [\![\lambda x :: \tau . t]\!]_\sigma = \lambda \mathbf{v} . [\![t]\!]_{\sigma[x \mapsto \mathbf{v}]} \\ \theta & \in \mathsf{Set}^{\mathsf{TVar}} & [\![t_1\ t_2]\!]_\sigma = [\![t_1]\!]_\sigma\ [\![t_2]\!]_\sigma \end{array}$$

Logical relation:

$$\begin{array}{ll} \Delta_{\alpha,\rho} & = \rho(\alpha) \qquad \Delta_{\mathsf{Nat},\rho} = id_{\mathbb{N}} \\ \Delta_{\tau_1 \to \tau_2,\rho} & = \{(\mathbf{f}, \mathbf{g}) \mid \forall (\mathbf{x}, \mathbf{y}) \in \Delta_{\tau_1,\rho}.\ (\mathbf{f}\ \mathbf{x}, \mathbf{g}\ \mathbf{y}) \in \Delta_{\tau_2,\rho}\} \end{array}$$

with $\rho \in \mathsf{Rel}^{\mathsf{TVar}}$

Theorem: for closed term $t$ of type $\tau$, $([\![t]\!]_\emptyset, [\![t]\!]_\emptyset) \in \Delta_{\tau,\rho}$.

# An Example Derivation, for `f` :: $\alpha \to \mathsf{Nat}$

$$\forall \rho, t \text{ closed with } t :: \tau.\ (\llbracket t \rrbracket_\emptyset, \llbracket t \rrbracket_\emptyset) \in \Delta_{\tau,\rho}$$

# An Example Derivation, for $\texttt{f} :: \alpha \to \mathsf{Nat}$

$$\forall \rho, t \text{ closed with } t :: \tau.\ (\llbracket t \rrbracket_\emptyset, \llbracket t \rrbracket_\emptyset) \in \Delta_{\tau,\rho}$$

$$\Rightarrow (t = \texttt{f} \text{ and } \tau = \alpha \to \mathsf{Nat})$$

$$\forall \rho.\ (\llbracket \texttt{f} \rrbracket_\emptyset, \llbracket \texttt{f} \rrbracket_\emptyset) \in \Delta_{\alpha \to \mathsf{Nat},\rho}$$

# An Example Derivation, for $\mathtt{f} :: \alpha \to \mathsf{Nat}$

$$\forall \rho, t \text{ closed with } t :: \tau.\ (\llbracket t \rrbracket_\emptyset, \llbracket t \rrbracket_\emptyset) \in \Delta_{\tau,\rho}$$

$$\Rightarrow (t = \mathtt{f} \text{ and } \tau = \alpha \to \mathsf{Nat})$$

$$\forall \rho.\ (\llbracket \mathtt{f} \rrbracket_\emptyset, \llbracket \mathtt{f} \rrbracket_\emptyset) \in \Delta_{\alpha \to \mathsf{Nat},\rho}$$

$$\Leftrightarrow (\Delta_{\tau_1 \to \tau_2,\rho} = \{(\mathbf{f}, \mathbf{g}) \mid \forall (\mathbf{x}, \mathbf{y}) \in \Delta_{\tau_1,\rho}.\ (\mathbf{f}\ \mathbf{x}, \mathbf{g}\ \mathbf{y}) \in \Delta_{\tau_2,\rho}\})$$

$$\forall \rho, (\mathbf{x}, \mathbf{y}) \in \Delta_{\alpha,\rho}.\ (\llbracket \mathtt{f} \rrbracket_\emptyset\ \mathbf{x}, \llbracket \mathtt{f} \rrbracket_\emptyset\ \mathbf{y}) \in \Delta_{\mathsf{Nat},\rho}$$

# An Example Derivation, for $\mathtt{f} :: \alpha \to \mathsf{Nat}$

$$\forall \rho, t \text{ closed with } t :: \tau.\ (\llbracket t \rrbracket_\emptyset, \llbracket t \rrbracket_\emptyset) \in \Delta_{\tau,\rho}$$

$$\Rightarrow (t = \mathtt{f} \text{ and } \tau = \alpha \to \mathsf{Nat})$$

$$\forall \rho.\ (\llbracket \mathtt{f} \rrbracket_\emptyset, \llbracket \mathtt{f} \rrbracket_\emptyset) \in \Delta_{\alpha \to \mathsf{Nat},\rho}$$

$$\Leftrightarrow (\Delta_{\tau_1 \to \tau_2,\rho} = \{(\mathbf{f}, \mathbf{g}) \mid \forall (\mathbf{x}, \mathbf{y}) \in \Delta_{\tau_1,\rho}.\ (\mathbf{f}\ \mathbf{x}, \mathbf{g}\ \mathbf{y}) \in \Delta_{\tau_2,\rho}\})$$

$$\forall \rho, (\mathbf{x}, \mathbf{y}) \in \Delta_{\alpha,\rho}.\ (\llbracket \mathtt{f} \rrbracket_\emptyset\ \mathbf{x}, \llbracket \mathtt{f} \rrbracket_\emptyset\ \mathbf{y}) \in \Delta_{\mathsf{Nat},\rho}$$

$$\Rightarrow (\Delta_{\alpha,\rho} = \rho(\alpha),\ \rho(\alpha) := \mathbf{g} \in \llbracket \tau_2 \rrbracket_\emptyset^{\llbracket \tau_1 \rrbracket_\emptyset},\ \Delta_{\mathsf{Nat},\rho} = id_{\mathbb{N}})$$

$$\forall \mathbf{g} \in \llbracket \tau_2 \rrbracket_\emptyset^{\llbracket \tau_1 \rrbracket_\emptyset}, \mathbf{x} \in \llbracket \tau_1 \rrbracket_\emptyset.\ \llbracket \mathtt{f} \rrbracket_\emptyset\ \mathbf{x} = \llbracket \mathtt{f} \rrbracket_\emptyset\ (\mathbf{g}\ \mathbf{x})$$

# An Example Derivation, for $\mathtt{f} :: \alpha \to \mathsf{Nat}$

$$\forall \rho, t \text{ closed with } t :: \tau.\ (\llbracket t \rrbracket_\emptyset, \llbracket t \rrbracket_\emptyset) \in \Delta_{\tau,\rho}$$

$\Rightarrow$ ($t = \mathtt{f}$ and $\tau = \alpha \to \mathsf{Nat}$)

$$\forall \rho.\ (\llbracket \mathtt{f} \rrbracket_\emptyset, \llbracket \mathtt{f} \rrbracket_\emptyset) \in \Delta_{\alpha \to \mathsf{Nat},\rho}$$

$\Leftrightarrow$ ($\Delta_{\tau_1 \to \tau_2,\rho} = \{(\mathbf{f}, \mathbf{g}) \mid \forall (\mathbf{x}, \mathbf{y}) \in \Delta_{\tau_1,\rho}.\ (\mathbf{f}\ \mathbf{x}, \mathbf{g}\ \mathbf{y}) \in \Delta_{\tau_2,\rho}\}$)

$$\forall \rho, (\mathbf{x}, \mathbf{y}) \in \Delta_{\alpha,\rho}.\ (\llbracket \mathtt{f} \rrbracket_\emptyset\ \mathbf{x}, \llbracket \mathtt{f} \rrbracket_\emptyset\ \mathbf{y}) \in \Delta_{\mathsf{Nat},\rho}$$

$\Rightarrow$ ($\Delta_{\alpha,\rho} = \rho(\alpha)$, $\rho(\alpha) := \mathbf{g} \in \llbracket \tau_2 \rrbracket_\emptyset^{\llbracket \tau_1 \rrbracket_\emptyset}$, $\Delta_{\mathsf{Nat},\rho} = id_{\mathbb{N}}$)

$$\forall \mathbf{g} \in \llbracket \tau_2 \rrbracket_\emptyset^{\llbracket \tau_1 \rrbracket_\emptyset}, \mathbf{x} \in \llbracket \tau_1 \rrbracket_\emptyset.\ \llbracket \mathtt{f} \rrbracket_\emptyset\ \mathbf{x} = \llbracket \mathtt{f} \rrbracket_\emptyset\ (\mathbf{g}\ \mathbf{x})$$

$\Rightarrow$ (term semantics)

$$\forall g :: \tau_1 \to \tau_2, x :: \tau_1.\ \llbracket \mathtt{f}\ x \rrbracket_\emptyset = \llbracket \mathtt{f}\ (g\ x) \rrbracket_\emptyset$$

# Automatic Generation of Free Theorems

At `http://www-ps.iai.uni-bonn.de/ft`:

Please enter a (polymorphic) type, e.g. "(a -> Bool) -> [a] -> [a]" or simply "filter":

```
f :: (a -> Bool) -> [a] -> [a]
```

Please choose a sublanguage of Haskell:

- (●) no bottoms (hence no general recursion and no selective strictness)
- ( ) general recursion but no selective strictness
- ( ) general recursion and selective strictness

Please choose a theorem style (without effect in the sublanguage with no bottoms):

- (●) equational
- ( ) inequational

[Generate] [✓] hide type instantiations (●) PNG ( ) Plain ( ) TeX ( ) PDF ?

# Automatic Generation of Free Theorems

**The Free Theorem for "f :: forall a . (a -> Bool) -> [a] -> [a]"**

$$\begin{array}{l}\forall\, t_1,t_2 \in \mathsf{TYPES},\, \mathsf{R} \in \mathsf{REL}(t_1,t_2).\\ \quad \forall\, p :: t_1 \to \textsc{Bool}.\\ \quad\quad \forall\, q :: t_2 \to \textsc{Bool}.\\ \quad\quad\quad (\forall\, (x, y) \in \mathsf{R}.\; p\; x = q\; y)\\ \quad\quad\quad \Rightarrow (\forall\, (z, v) \in \mathsf{LIFT}\{[]\}(\mathsf{R}).\; (f\; p\; z, f\; q\; v) \in \mathsf{LIFT}\{[]\}(\mathsf{R}))\end{array}$$

$$\begin{array}{l}\mathsf{LIFT}\{[]\}(\mathsf{R})\\ \quad = \{([], [])\}\\ \quad \cup \{(x : xs, y : ys) \mid ((x, y) \in \mathsf{R}) \wedge ((xs, ys) \in \mathsf{LIFT}\{[]\}(\mathsf{R}))\}\end{array}$$

**Reducing all permissable relation variables to functions**

$$\begin{array}{l}\forall\, t_1,t_2 \in \mathsf{TYPES},\, g :: t_1 \to t_2.\\ \quad \forall\, p :: t_1 \to \textsc{Bool}.\\ \quad\quad \forall\, q :: t_2 \to \textsc{Bool}.\\ \quad\quad\quad (\forall\, x :: t_1.\; p\; x = q\; (g\; x))\\ \quad\quad\quad \Rightarrow (\forall\, y :: [t_1].\; \mathsf{map}\; g\; (f\; p\; y) = f\; q\; (\mathsf{map}\; g\; y))\end{array}$$

# Applications

- Short Cut Fusion [Gill et al., FPCA'93]
- The Dual of Short Cut Fusion
  [Svenningsson, ICFP'02]
- Circular Short Cut Fusion
  [Fernandes et al., Haskell'07]
- ...
- Testing polymorphic properties
  [Bernardy et al., ESOP'10]
- ...

# What About Efficiency?

| `f` :: $\alpha \to$ Nat | `f` :: $\alpha \to \alpha \to \alpha$ | `f` :: $\alpha \to (\alpha, \alpha)$ |
|---|---|---|
| `f` $(g\ x)$ $=$ `f` $x$ | `f` $(g\ x)\ (g\ y)$ $=$ $g$ $(\texttt{f}\ x\ y)$ | `f` $(g\ x)$ $=$ **let** $y =$ `f` $x$ **in** $(g\ (\texttt{fst}\ y), g\ (\texttt{snd}\ y))$ |

# What About Efficiency?

| | `f` :: $\alpha \to$ Nat | `f` :: $\alpha \to \alpha \to \alpha$ | `f` :: $\alpha \to (\alpha, \alpha)$ |
|---|---|---|---|
| | `f` $(g\ x)$ $=$ `f` $x$ | `f` $(g\ x)\ (g\ y)$ $=$ $g\ (\texttt{f}\ x\ y)$ | `f` $(g\ x)$ $=$ **let** $y =$ `f` $x$ **in** $(g\ (\texttt{fst}\ y), g\ (\texttt{snd}\ y))$ |
| call-by-value | $>$ | $>$ | $<$ |

# What About Efficiency?

| | `f` :: $\alpha \to$ Nat | `f` :: $\alpha \to \alpha \to \alpha$ | `f` :: $\alpha \to (\alpha, \alpha)$ |
|---|---|---|---|
| | `f` $(g\ x)$ $=$ `f` $x$ | `f` $(g\ x)\ (g\ y)$ $=$ $g$ (`f` $x\ y$) | `f` $(g\ x)$ $=$ **let** $y =$ `f` $x$ **in** $(g$ (`fst` $y$), $g$ (`snd` $y$)) |
| call-by-value | $>$ | $>$ | $<$ |
| call-by-name | $=$ | $=$ | $<$ (?) |

# What About Efficiency?

| | $\texttt{f} :: \alpha \to \mathsf{Nat}$ | $\texttt{f} :: \alpha \to \alpha \to \alpha$ | $\texttt{f} :: \alpha \to (\alpha, \alpha)$ |
|---|---|---|---|
| | $\texttt{f}\ (g\ x)$ $=$ $\texttt{f}\ x$ | $\texttt{f}\ (g\ x)\ (g\ y)$ $=$ $g\ (\texttt{f}\ x\ y)$ | $\texttt{f}\ (g\ x)$ $=$ **let** $y = \texttt{f}\ x$ **in** $(g\ (\texttt{fst}\ y), g\ (\texttt{snd}\ y))$ |
| call-by-value | $>$ | $>$ | $<$ |
| call-by-name | $=$ | $=$ | $<$ (?) |
| call-by-need | $=$ | $=$ | $<$ |

# What About Efficiency?

| | $\texttt{f} :: \alpha \to \mathsf{Nat}$ | $\texttt{f} :: \alpha \to \alpha \to \alpha$ | $\texttt{f} :: \alpha \to (\alpha, \alpha)$ |
|---|---|---|---|
| | $\texttt{f}\ (g\ x)$ $=$ $\texttt{f}\ x$ | $\texttt{f}\ (g\ x)\ (g\ y)$ $=$ $g\ (\texttt{f}\ x\ y)$ | $\texttt{f}\ (g\ x)$ $=$ **let** $y = \texttt{f}\ x$ **in** $(g\ (\texttt{fst}\ y), g\ (\texttt{snd}\ y))$ |
| call-by-value | $>$ | $>$ | $<$ |
| call-by-name | $=$ | $=$ | $<$ (?) |
| call-by-need | $=$ | $=$ | $<$ |

# What About Efficiency?

Now, how about, say `f` :: $\alpha \to \alpha \to (\alpha, \alpha)$ ?

# What About Efficiency?

Now, how about, say `f` :: $\alpha \to \alpha \to (\alpha, \alpha)$ ?

Or `f` :: $[\alpha] \to [\alpha]$ ?

# What About Efficiency?

Now, how about, say `f` :: $\alpha \to \alpha \to (\alpha, \alpha)$ ?

Or `f` :: $[\alpha] \to [\alpha]$ ?

And how to actually establish such statements?

# But isn't it Somehow Trivial?

Recall `f` :: $\alpha \to$ Nat, with standard free theorem:

$$\texttt{f}\ (g\ x) = \texttt{f}\ x$$

for all choices of types $\tau_1$, $\tau_2$, function $g :: \tau_1 \to \tau_2$, and $x :: \tau_1$.

# But isn't it Somehow Trivial?

Recall `f` :: $\alpha \to$ Nat, with standard free theorem:

$$\texttt{f}\ (g\ x) = \texttt{f}\ x$$

for all choices of types $\tau_1$, $\tau_2$, function $g :: \tau_1 \to \tau_2$, and $x :: \tau_1$. Clearly, this means that `f` is a constant function, i.e., for all $x$ and $y$:

$$\texttt{f}\ y = \texttt{f}\ x$$

# But isn't it Somehow Trivial?

Recall `f` :: $\alpha \to$ Nat, with standard free theorem:

$$\texttt{f}\ (g\ x) = \texttt{f}\ x$$

for all choices of types $\tau_1$, $\tau_2$, function $g :: \tau_1 \to \tau_2$, and $x :: \tau_1$. Clearly, this means that `f` is a constant function, i.e., for all $x$ and $y$:

$$\texttt{f}\ y = \texttt{f}\ x$$

So, "obviously",

$$\texttt{f}\ (g\ x) \sqsupseteq \texttt{f}\ x$$

where $\sqsupseteq$ means "the same result, and equally fast or slower".

# But isn't it Somehow Trivial?

Recall $\texttt{f} :: \alpha \to \text{Nat}$, with standard free theorem:

$$\texttt{f}\ (g\ x) = \texttt{f}\ x$$

for all choices of types $\tau_1$, $\tau_2$, function $g :: \tau_1 \to \tau_2$, and $x :: \tau_1$. Clearly, this means that $\texttt{f}$ is a constant function, i.e., for all $x$ and $y$:

$$\texttt{f}\ y = \texttt{f}\ x$$

So, "obviously",

$$\texttt{f}\ (g\ x) \sqsupseteq \texttt{f}\ x$$

where $\sqsupseteq$ means "the same result, and equally fast or slower". What's wrong with this reasoning?

# But isn't it Somehow Trivial?

Consider:

$$\texttt{f}\ x = \textbf{if}\ x == 0 \ \textbf{then}\ 0\ \textbf{else}\ \texttt{f}\ (x-1) \qquad g\ x = 0$$

# But isn't it Somehow Trivial?

Consider:

$$\texttt{f}\ x = \textbf{if}\ x == 0\ \textbf{then}\ 0\ \textbf{else}\ \texttt{f}\ (x-1) \qquad g\ x = 0$$

Then certainly not, for $x > 0$:

$$\texttt{f}\ (g\ x) \sqsupseteq \texttt{f}\ x$$

# But isn't it Somehow Trivial?

Consider:

$$\texttt{f} :: \mathsf{Nat} \to \mathsf{Nat}$$
$$\texttt{f}\ x = \textbf{if}\ x == 0\ \textbf{then}\ 0\ \textbf{else}\ \texttt{f}\ (x - 1)$$

$$g :: \alpha \to \mathsf{Nat}$$
$$g\ x = 0$$

Then certainly not, for $x > 0$:

$$\texttt{f}\ (g\ x) \sqsupseteq \texttt{f}\ x$$

# But isn't it Somehow Trivial?

Consider:

$$\texttt{f} :: \mathsf{Nat} \to \mathsf{Nat}$$
$$\texttt{f}\ x = \textbf{if}\ x == 0\ \textbf{then}\ 0\ \textbf{else}\ \texttt{f}\ (x - 1)$$

$$g :: \alpha \to \mathsf{Nat}$$
$$g\ x = 0$$

Then certainly not, for $x > 0$:

$$\texttt{f}\ (g\ x) \sqsupseteq \texttt{f}\ x$$

Exploiting polymorphism is really essential,
and not just for the extensional statements!

# Bringing Costs into the Semantics

New semantics:

$$
\begin{aligned}
[\![x]\!]^{¢}_{\sigma} &= (\sigma(x), 0) \\
[\![n]\!]^{¢}_{\sigma} &= (\mathbf{n}, 0) \\
[\![\lambda x :: \tau.t]\!]^{¢}_{\sigma} &= (\lambda \mathbf{v}.1 \rhd [\![t]\!]^{¢}_{\sigma[x \mapsto \mathbf{v}]}, 0) \\
[\![t_1\ t_2]\!]^{¢}_{\sigma} &= [\![t_1]\!]^{¢}_{\sigma}\ ¢\ [\![t_2]\!]^{¢}_{\sigma}
\end{aligned}
$$

where: $c \rhd (\mathbf{v}, c') = (\mathbf{v}, c + c')$ and

$$
\mathbf{f}\ ¢\ \mathbf{x} = (c + c') \rhd (\mathbf{g}\ \mathbf{v}) \quad \text{if } \mathbf{f} = (\mathbf{g}, c), \ \mathbf{x} = (\mathbf{v}, c')
$$

# Bringing Costs into the Semantics

New semantics:

$$
\begin{aligned}
[\![x]\!]^{¢}_{\sigma} &= (\sigma(x), 0) \\
[\![n]\!]^{¢}_{\sigma} &= (\mathbf{n}, 0) \\
[\![\lambda x :: \tau . t]\!]^{¢}_{\sigma} &= (\lambda \mathbf{v}.1 \rhd [\![t]\!]^{¢}_{\sigma[x \mapsto \mathbf{v}]}, 0) \\
[\![t_1\ t_2]\!]^{¢}_{\sigma} &= [\![t_1]\!]^{¢}_{\sigma}\ ¢\ [\![t_2]\!]^{¢}_{\sigma}
\end{aligned}
$$

where: $c \rhd (\mathbf{v}, c') = (\mathbf{v}, c + c')$ and

$$
\mathbf{f}\ ¢\ \mathbf{x} = (c + c') \rhd (\mathbf{g}\ \mathbf{v}) \quad \text{if } \mathbf{f} = (\mathbf{g}, c),\ \mathbf{x} = (\mathbf{v}, c')
$$

Note: $t :: \alpha \to \mathsf{Nat}$ implies $[\![t]\!]^{¢}_{\emptyset} \in \mathcal{C}(\mathcal{C}(\mathbb{N})^{\theta(\alpha)})$,
for every $\theta \in \mathsf{Set}^{\mathsf{TVar}}$,
where $\mathcal{C}(S) = \{(\mathbf{v}, c) \mid \mathbf{v} \in S, c \in \mathbb{Z}\}$.

# How about the Logical Relation?

Tempting would be:

$$\Delta^{¢}_{\alpha,\rho} = \rho(\alpha) \qquad \Delta^{¢}_{\mathsf{Nat},\rho} = id_{\mathbb{N}\times\mathbb{Z}}$$

$$\Delta^{¢}_{\tau_1\to\tau_2,\rho} = \{(\mathbf{f},\mathbf{g}) \mid \forall(\mathbf{x},\mathbf{y}) \in \Delta^{¢}_{\tau_1,\rho}. \; (\mathbf{f}\ ¢\ \mathbf{x}, \mathbf{g}\ ¢\ \mathbf{y}) \in \Delta^{¢}_{\tau_2,\rho}\}$$

with $\rho(\alpha_1), \rho(\alpha_2), \ldots \subseteq \mathcal{C}(S_i) \times \mathcal{C}(T_i)$

# How about the Logical Relation?

Tempting would be:

$$\Delta^{\Cent}_{\alpha,\rho} = \rho(\alpha) \qquad \Delta^{\Cent}_{\mathsf{Nat},\rho} = id_{\mathbb{N}\times\mathbb{Z}}$$

$$\Delta^{\Cent}_{\tau_1\to\tau_2,\rho} = \{(\mathbf{f},\mathbf{g}) \mid \forall(\mathbf{x},\mathbf{y}) \in \Delta^{\Cent}_{\tau_1,\rho}. \; (\mathbf{f}\,\Cent\,\mathbf{x}, \mathbf{g}\,\Cent\,\mathbf{y}) \in \Delta^{\Cent}_{\tau_2,\rho}\}$$

with $\rho(\alpha_1), \rho(\alpha_2), \ldots \subseteq \mathcal{C}(S_i) \times \mathcal{C}(T_i)$

But NO, does not work!

# How about the Logical Relation?

Tempting would be:

$$\Delta^{\mathbin{¢}}_{\alpha,\rho} = \rho(\alpha) \qquad \Delta^{\mathbin{¢}}_{\mathsf{Nat},\rho} = id_{\mathbb{N}\times\mathbb{Z}}$$

$$\Delta^{\mathbin{¢}}_{\tau_1\to\tau_2,\rho} = \{(\mathbf{f},\mathbf{g}) \mid \forall(\mathbf{x},\mathbf{y}) \in \Delta^{\mathbin{¢}}_{\tau_1,\rho}. \; (\mathbf{f} \mathbin{¢} \mathbf{x}, \mathbf{g} \mathbin{¢} \mathbf{y}) \in \Delta^{\mathbin{¢}}_{\tau_2,\rho}\}$$

with $\rho(\alpha_1), \rho(\alpha_2), \ldots \subseteq \mathcal{C}(S_i) \times \mathcal{C}(T_i)$

But NO, does not work! It would allow us to conclude from:

$$\forall\rho, \mathtt{f} :: \alpha \to \mathsf{Nat}.\ (\llbracket \mathtt{f} \rrbracket^{\mathbin{¢}}_{\emptyset}, \llbracket \mathtt{f} \rrbracket^{\mathbin{¢}}_{\emptyset}) \in \Delta^{\mathbin{¢}}_{\alpha\to\mathsf{Nat},\rho}$$

that:

$$\forall g :: \tau_1 \to \tau_2, x :: \tau_1.\ \llbracket \mathtt{f}\ x \rrbracket^{\mathbin{¢}}_{\emptyset} = \llbracket \mathtt{f}\ (g\ x) \rrbracket^{\mathbin{¢}}_{\emptyset}$$

# How about the Logical Relation?

Much more disciplined:

$$\Delta^{\mathbin{¢}}_{\alpha,\rho} = \mathcal{C}(\rho(\alpha)) \qquad \Delta^{\mathbin{¢}}_{\mathsf{Nat},\rho} = id_{\mathbb{N}\times\mathbb{Z}}$$

$$\Delta^{\mathbin{¢}}_{\tau_1\to\tau_2,\rho} = \{(\mathbf{f},\mathbf{g}) \mid cost(\mathbf{f}) = cost(\mathbf{g}) \wedge \forall(\mathbf{x},\mathbf{y}) \in \Delta^{\mathbin{¢}}_{\tau_1,\rho}.\ (\mathbf{f} \mathbin{¢} \mathbf{x}, \mathbf{g} \mathbin{¢} \mathbf{y}) \in \Delta^{\mathbin{¢}}_{\tau_2,\rho}\}$$

with $\rho(\alpha_1), \rho(\alpha_2), \ldots \subseteq S_i \times T_i$,

where $\mathcal{C}(R) = \{((\mathbf{u},c),(\mathbf{v},c) \mid (\mathbf{u},\mathbf{v}) \in R, c \in \mathbb{Z}\}$

# How about the Logical Relation?

Much more disciplined:

$$\Delta^{\mathfrak{C}}_{\alpha,\rho} = \mathcal{C}(\rho(\alpha)) \qquad \Delta^{\mathfrak{C}}_{\mathsf{Nat},\rho} = id_{\mathbb{N}\times\mathbb{Z}}$$

$$\Delta^{\mathfrak{C}}_{\tau_1\to\tau_2,\rho} = \{(\mathbf{f},\mathbf{g}) \mid cost(\mathbf{f}) = cost(\mathbf{g}) \wedge \forall(\mathbf{x},\mathbf{y}) \in \Delta^{\mathfrak{C}}_{\tau_1,\rho}.\ (\mathbf{f}\ \mathfrak{C}\ \mathbf{x}, \mathbf{g}\ \mathfrak{C}\ \mathbf{y}) \in \Delta^{\mathfrak{C}}_{\tau_2,\rho}\}$$

with $\rho(\alpha_1), \rho(\alpha_2), \ldots \subseteq S_i \times T_i$,

where $\mathcal{C}(R) = \{((\mathbf{u},c),(\mathbf{v},c) \mid (\mathbf{u},\mathbf{v}) \in R, c \in \mathbb{Z}\}$

Now indeed ...

Theorem: for closed term $t$ of type $\tau$,

$$([\![t]\!]^{\mathfrak{C}}_{\emptyset}, [\![t]\!]^{\mathfrak{C}}_{\emptyset}) \in \Delta^{\mathfrak{C}}_{\tau,\rho}$$

# Now, What about our Example $\texttt{f} :: \alpha \to \mathsf{Nat}$ ?

$$\forall \rho.\ (\llbracket \texttt{f} \rrbracket^{\mathfrak{C}}_{\emptyset}, \llbracket \texttt{f} \rrbracket^{\mathfrak{C}}_{\emptyset}) \in \Delta^{\mathfrak{C}}_{\alpha \to \mathsf{Nat}, \rho}$$

# Now, What about our Example $\texttt{f} :: \alpha \to \mathsf{Nat}$ ?

$$\forall \rho.\ (\llbracket \texttt{f} \rrbracket^{\mathfrak{C}}_{\emptyset}, \llbracket \texttt{f} \rrbracket^{\mathfrak{C}}_{\emptyset}) \in \Delta^{\mathfrak{C}}_{\alpha \to \mathsf{Nat}, \rho}$$

$$\Rightarrow (\Delta^{\mathfrak{C}}_{\tau_1 \to \tau_2, \rho} = \{(\mathbf{f}, \mathbf{g}) \mid \mathit{cost}(\mathbf{f}) = \mathit{cost}(\mathbf{g}) \wedge \forall (\mathbf{x}, \mathbf{y}) \in \Delta^{\mathfrak{C}}_{\tau_1, \rho}.\ (\mathbf{f}\ \mathfrak{C}\ \mathbf{x}, \mathbf{g}\ \mathfrak{C}\ \mathbf{y}) \in \Delta^{\mathfrak{C}}_{\tau_2, \rho}\})$$

$$\forall \rho, (\mathbf{x}, \mathbf{y}) \in \Delta^{\mathfrak{C}}_{\alpha, \rho}.\ (\llbracket \texttt{f} \rrbracket^{\mathfrak{C}}_{\emptyset}\ \mathfrak{C}\ \mathbf{x}, \llbracket \texttt{f} \rrbracket^{\mathfrak{C}}_{\emptyset}\ \mathfrak{C}\ \mathbf{y}) \in \Delta^{\mathfrak{C}}_{\mathsf{Nat}, \rho}$$

# Now, What about our Example $\texttt{f} :: \alpha \to \mathsf{Nat}$ ?

$$\forall \rho.\ (\llbracket \texttt{f} \rrbracket^{\mathfrak{C}}_{\emptyset}, \llbracket \texttt{f} \rrbracket^{\mathfrak{C}}_{\emptyset}) \in \Delta^{\mathfrak{C}}_{\alpha \to \mathsf{Nat}, \rho}$$

$$\Rightarrow (\Delta^{\mathfrak{C}}_{\tau_1 \to \tau_2, \rho} = \{(\mathbf{f}, \mathbf{g}) \mid \mathit{cost}(\mathbf{f}) = \mathit{cost}(\mathbf{g}) \wedge \forall (\mathbf{x}, \mathbf{y}) \in \Delta^{\mathfrak{C}}_{\tau_1, \rho}.\ (\mathbf{f}\ \mathfrak{C}\ \mathbf{x}, \mathbf{g}\ \mathfrak{C}\ \mathbf{y}) \in \Delta^{\mathfrak{C}}_{\tau_2, \rho}\})$$

$$\forall \rho, (\mathbf{x}, \mathbf{y}) \in \Delta^{\mathfrak{C}}_{\alpha, \rho}.\ (\llbracket \texttt{f} \rrbracket^{\mathfrak{C}}_{\emptyset}\ \mathfrak{C}\ \mathbf{x}, \llbracket \texttt{f} \rrbracket^{\mathfrak{C}}_{\emptyset}\ \mathfrak{C}\ \mathbf{y}) \in \Delta^{\mathfrak{C}}_{\mathsf{Nat}, \rho}$$

$$\Rightarrow (\Delta^{\mathfrak{C}}_{\alpha, \rho} = \mathcal{C}(\rho(\alpha)),\ \rho(\alpha) := R \in \mathsf{Rel},\ \Delta^{\mathfrak{C}}_{\mathsf{Nat}, \rho} = \mathit{id}_{\mathbb{N} \times \mathbb{Z}})$$

$$\forall R \in \mathsf{Rel}, (\mathbf{x}, \mathbf{y}) \in \mathcal{C}(R).\ \llbracket \texttt{f} \rrbracket^{\mathfrak{C}}_{\emptyset}\ \mathfrak{C}\ \mathbf{x} = \llbracket \texttt{f} \rrbracket^{\mathfrak{C}}_{\emptyset}\ \mathfrak{C}\ \mathbf{y}$$

# Now, What about our Example $\mathtt{f} :: \alpha \to \mathsf{Nat}$ ?

$$\forall \rho.\ (\llbracket \mathtt{f} \rrbracket^{\mathfrak{C}}_{\emptyset}, \llbracket \mathtt{f} \rrbracket^{\mathfrak{C}}_{\emptyset}) \in \Delta^{\mathfrak{C}}_{\alpha \to \mathsf{Nat}, \rho}$$

$$\Rightarrow (\Delta^{\mathfrak{C}}_{\tau_1 \to \tau_2, \rho} = \{(\mathbf{f}, \mathbf{g}) \mid \mathit{cost}(\mathbf{f}) = \mathit{cost}(\mathbf{g}) \wedge \forall (\mathbf{x}, \mathbf{y}) \in \Delta^{\mathfrak{C}}_{\tau_1, \rho}.\ (\mathbf{f}\ \mathfrak{C}\ \mathbf{x}, \mathbf{g}\ \mathfrak{C}\ \mathbf{y}) \in \Delta^{\mathfrak{C}}_{\tau_2, \rho}\})$$

$$\forall \rho, (\mathbf{x}, \mathbf{y}) \in \Delta^{\mathfrak{C}}_{\alpha, \rho}.\ (\llbracket \mathtt{f} \rrbracket^{\mathfrak{C}}_{\emptyset}\ \mathfrak{C}\ \mathbf{x}, \llbracket \mathtt{f} \rrbracket^{\mathfrak{C}}_{\emptyset}\ \mathfrak{C}\ \mathbf{y}) \in \Delta^{\mathfrak{C}}_{\mathsf{Nat}, \rho}$$

$$\Rightarrow (\Delta^{\mathfrak{C}}_{\alpha, \rho} = \mathcal{C}(\rho(\alpha)),\ \rho(\alpha) := R \in \mathsf{Rel},\ \Delta^{\mathfrak{C}}_{\mathsf{Nat}, \rho} = \mathit{id}_{\mathbb{N} \times \mathbb{Z}})$$

$$\forall R \in \mathsf{Rel}, (\mathbf{x}, \mathbf{y}) \in \mathcal{C}(R).\ \llbracket \mathtt{f} \rrbracket^{\mathfrak{C}}_{\emptyset}\ \mathfrak{C}\ \mathbf{x} = \llbracket \mathtt{f} \rrbracket^{\mathfrak{C}}_{\emptyset}\ \mathfrak{C}\ \mathbf{y}$$

How to choose $R$ to bring $g :: \tau_1 \to \tau_2$ into play?

# Now, What about our Example $\mathtt{f} :: \alpha \to \mathsf{Nat}$ ?

$$\forall \rho.\ (\llbracket \mathtt{f} \rrbracket^{\mathfrak{C}}_{\emptyset}, \llbracket \mathtt{f} \rrbracket^{\mathfrak{C}}_{\emptyset}) \in \Delta^{\mathfrak{C}}_{\alpha \to \mathsf{Nat}, \rho}$$

$$\Rightarrow (\Delta^{\mathfrak{C}}_{\tau_1 \to \tau_2, \rho} = \{(\mathbf{f}, \mathbf{g}) \mid \mathit{cost}(\mathbf{f}) = \mathit{cost}(\mathbf{g}) \wedge \forall (\mathbf{x}, \mathbf{y}) \in \Delta^{\mathfrak{C}}_{\tau_1, \rho}.\ (\mathbf{f}\ \mathfrak{C}\ \mathbf{x}, \mathbf{g}\ \mathfrak{C}\ \mathbf{y}) \in \Delta^{\mathfrak{C}}_{\tau_2, \rho}\})$$

$$\forall \rho, (\mathbf{x}, \mathbf{y}) \in \Delta^{\mathfrak{C}}_{\alpha, \rho}.\ (\llbracket \mathtt{f} \rrbracket^{\mathfrak{C}}_{\emptyset}\ \mathfrak{C}\ \mathbf{x}, \llbracket \mathtt{f} \rrbracket^{\mathfrak{C}}_{\emptyset}\ \mathfrak{C}\ \mathbf{y}) \in \Delta^{\mathfrak{C}}_{\mathsf{Nat}, \rho}$$

$$\Rightarrow (\Delta^{\mathfrak{C}}_{\alpha, \rho} = \mathcal{C}(\rho(\alpha)),\ \rho(\alpha) := R \in \mathsf{Rel},\ \Delta^{\mathfrak{C}}_{\mathsf{Nat}, \rho} = \mathit{id}_{\mathbb{N} \times \mathbb{Z}})$$

$$\forall R \in \mathsf{Rel}, (\mathbf{x}, \mathbf{y}) \in \mathcal{C}(R).\ \llbracket \mathtt{f} \rrbracket^{\mathfrak{C}}_{\emptyset}\ \mathfrak{C}\ \mathbf{x} = \llbracket \mathtt{f} \rrbracket^{\mathfrak{C}}_{\emptyset}\ \mathfrak{C}\ \mathbf{y}$$

How to choose $R$ to bring $g :: \tau_1 \to \tau_2$ into play?

Problem: $\{(\mathbf{x}, \mathbf{y}) \mid \llbracket g \rrbracket^{\mathfrak{C}}_{\emptyset}\ \mathfrak{C}\ \mathbf{x} = \mathbf{y}\}$ not $\mathcal{C}(R)$ for any $R$

# Now, What about our Example `f` :: $\alpha \to \mathsf{Nat}$ ?

Problem: $\{(\mathbf{x}, \mathbf{y}) \mid [\![g]\!]^{\mathbb{¢}}_{\emptyset}\ \mathbb{¢}\ \mathbf{x} = \mathbf{y}\}$ not $\mathcal{C}(R)$ for any $R$

Solution: but

$$\{(c \rhd \mathit{appCost}(g, \mathbf{x}) \rhd \mathbf{x}, c \rhd \mathbf{y}) \mid [\![g]\!]^{\mathbb{¢}}_{\emptyset}\ \mathbb{¢}\ \mathbf{x} = \mathbf{y}, c \in \mathbb{Z}\} = \mathcal{C}(R^g)$$

for

$$R^g = \{(\mathit{val}(\mathbf{x}), \mathit{val}([\![g]\!]^{\mathbb{¢}}_{\emptyset}\ \mathbb{¢}\ \mathbf{x})) \mid \mathbf{x} \in [\![\tau_1]\!]^{\mathbb{¢}}_{\emptyset}\}$$

where $\mathit{appCost}(g, \mathbf{x}) = \mathit{cost}([\![g]\!]^{\mathbb{¢}}_{\emptyset}\ \mathbb{¢}\ \mathbf{x}) - \mathit{cost}(\mathbf{x})$

# Now, What about our Example $\texttt{f} :: \alpha \to \mathsf{Nat}$ ?

Problem: $\{(\mathbf{x}, \mathbf{y}) \mid [\![g]\!]^{¢}_{\emptyset}\ ¢\ \mathbf{x} = \mathbf{y}\}$ not $\mathcal{C}(R)$ for any $R$

Solution: but

$$\{(c \rhd \mathit{appCost}(g, \mathbf{x}) \rhd \mathbf{x}, c \rhd \mathbf{y}) \mid [\![g]\!]^{¢}_{\emptyset}\ ¢\ \mathbf{x} = \mathbf{y}, c \in \mathbb{Z}\} = \mathcal{C}(R^g)$$

for

$$R^g = \{(\mathit{val}(\mathbf{x}), \mathit{val}([\![g]\!]^{¢}_{\emptyset}\ ¢\ \mathbf{x})) \mid \mathbf{x} \in [\![\tau_1]\!]^{¢}_{\emptyset}\}$$

where $\mathit{appCost}(g, \mathbf{x}) = \mathit{cost}([\![g]\!]^{¢}_{\emptyset}\ ¢\ \mathbf{x}) - \mathit{cost}(\mathbf{x})$

Now:

$$\forall R \in \mathsf{Rel}, (\mathbf{x}, \mathbf{y}) \in \mathcal{C}(R).\ [\![\texttt{f}]\!]^{¢}_{\emptyset}\ ¢\ \mathbf{x} = [\![\texttt{f}]\!]^{¢}_{\emptyset}\ ¢\ \mathbf{y}$$

$$\Rightarrow\ \forall \mathbf{x} \in [\![\tau_1]\!]^{¢}_{\emptyset}.\ [\![\texttt{f}]\!]^{¢}_{\emptyset}\ ¢\ (\mathit{appCost}(g, \mathbf{x}) \rhd \mathbf{x}) = [\![\texttt{f}]\!]^{¢}_{\emptyset}\ ¢\ ([\![g]\!]^{¢}_{\emptyset}\ ¢\ \mathbf{x})$$

# Now, What about our Example $\texttt{f} :: \alpha \to \mathsf{Nat}$ ?

Now:

$$\forall R \in \mathsf{Rel}, (\mathbf{x}, \mathbf{y}) \in \mathcal{C}(R).\ [\![\texttt{f}]\!]^{¢}_{\emptyset}\ ¢\ \mathbf{x} = [\![\texttt{f}]\!]^{¢}_{\emptyset}\ ¢\ \mathbf{y}$$

$$\Rightarrow\ \forall \mathbf{x} \in [\![\tau_1]\!]^{¢}_{\emptyset}.\ [\![\texttt{f}]\!]^{¢}_{\emptyset}\ ¢\ (\mathit{appCost}(g, \mathbf{x}) \rhd \mathbf{x}) = [\![\texttt{f}]\!]^{¢}_{\emptyset}\ ¢\ ([\![g]\!]^{¢}_{\emptyset}\ ¢\ \mathbf{x})$$

# Now, What about our Example $\mathtt{f} :: \alpha \to \mathsf{Nat}$ ?

Now:

$$\forall R \in \mathsf{Rel}, (\mathbf{x}, \mathbf{y}) \in \mathcal{C}(R).\ [\![\mathtt{f}]\!]^{¢}_{\emptyset}\ ¢\ \mathbf{x} = [\![\mathtt{f}]\!]^{¢}_{\emptyset}\ ¢\ \mathbf{y}$$

$$\Rightarrow\ \forall \mathbf{x} \in [\![\tau_1]\!]^{¢}_{\emptyset}.\ [\![\mathtt{f}]\!]^{¢}_{\emptyset}\ ¢\ (\mathit{appCost}(g, \mathbf{x}) \rhd \mathbf{x}) = [\![\mathtt{f}]\!]^{¢}_{\emptyset}\ ¢\ ([\![g]\!]^{¢}_{\emptyset}\ ¢\ \mathbf{x})$$

$$\Rightarrow\ \forall x :: \tau_1.\ \mathit{appCost}(g, [\![x]\!]^{¢}_{\emptyset}) \rhd ([\![\mathtt{f}]\!]^{¢}_{\emptyset}\ ¢\ [\![x]\!]^{¢}_{\emptyset}) = [\![\mathtt{f}]\!]^{¢}_{\emptyset}\ ¢\ ([\![g]\!]^{¢}_{\emptyset}\ ¢\ [\![x]\!]^{¢}_{\emptyset})$$

# Now, What about our Example $\texttt{f} :: \alpha \to \mathsf{Nat}$ ?

Now:

$$\forall R \in \mathsf{Rel}, (\mathbf{x}, \mathbf{y}) \in \mathcal{C}(R).\ [\![\texttt{f}]\!]^{¢}_{\emptyset}\ ¢\ \mathbf{x} = [\![\texttt{f}]\!]^{¢}_{\emptyset}\ ¢\ \mathbf{y}$$

$$\Rightarrow\ \forall \mathbf{x} \in [\![\tau_1]\!]^{¢}_{\emptyset}.\ [\![\texttt{f}]\!]^{¢}_{\emptyset}\ ¢\ (\mathit{appCost}(g, \mathbf{x}) \rhd \mathbf{x}) = [\![\texttt{f}]\!]^{¢}_{\emptyset}\ ¢\ ([\![g]\!]^{¢}_{\emptyset}\ ¢\ \mathbf{x})$$

$$\Rightarrow\ \forall x :: \tau_1.\ \mathit{appCost}(g, [\![x]\!]^{¢}_{\emptyset}) \rhd ([\![\texttt{f}]\!]^{¢}_{\emptyset}\ ¢\ [\![x]\!]^{¢}_{\emptyset}) = [\![\texttt{f}]\!]^{¢}_{\emptyset}\ ¢\ ([\![g]\!]^{¢}_{\emptyset}\ ¢\ [\![x]\!]^{¢}_{\emptyset})$$

$$\Rightarrow\ \forall x :: \tau_1.\ \mathit{appCost}(g, [\![x]\!]^{¢}_{\emptyset}) \rhd [\![\texttt{f}\ x]\!]^{¢}_{\emptyset} = [\![\texttt{f}\ (g\ x)]\!]^{¢}_{\emptyset}$$

# Now, What about our Example $\texttt{f} :: \alpha \to \mathsf{Nat}$ ?

Now:

$$\forall R \in \mathsf{Rel}, (\mathbf{x}, \mathbf{y}) \in \mathcal{C}(R).\ [\![\texttt{f}]\!]^{¢}_{\emptyset}\ ¢\ \mathbf{x} = [\![\texttt{f}]\!]^{¢}_{\emptyset}\ ¢\ \mathbf{y}$$

$$\begin{aligned}
\Rightarrow\ & \forall \mathbf{x} \in [\![\tau_1]\!]^{¢}_{\emptyset}.\ [\![\texttt{f}]\!]^{¢}_{\emptyset}\ ¢\ (\mathit{appCost}(g, \mathbf{x}) \rhd \mathbf{x}) \\
& \qquad = [\![\texttt{f}]\!]^{¢}_{\emptyset}\ ¢\ ([\![g]\!]^{¢}_{\emptyset}\ ¢\ \mathbf{x}) \\
\Rightarrow\ & \forall x :: \tau_1.\ \mathit{appCost}(g, [\![x]\!]^{¢}_{\emptyset}) \rhd ([\![\texttt{f}]\!]^{¢}_{\emptyset}\ ¢\ [\![x]\!]^{¢}_{\emptyset}) \\
& \qquad = [\![\texttt{f}]\!]^{¢}_{\emptyset}\ ¢\ ([\![g]\!]^{¢}_{\emptyset}\ ¢\ [\![x]\!]^{¢}_{\emptyset}) \\
\Rightarrow\ & \forall x :: \tau_1.\ \mathit{appCost}(g, [\![x]\!]^{¢}_{\emptyset}) \rhd [\![\texttt{f}\ x]\!]^{¢}_{\emptyset} = [\![\texttt{f}\ (g\ x)]\!]^{¢}_{\emptyset}
\end{aligned}$$

Hence:

$$\texttt{f}\ x \sqsubset \texttt{f}\ (g\ x)$$

# Let's Try another Example, $\texttt{f} :: \alpha \to \alpha$

$$\forall \rho.\ (\llbracket \texttt{f} \rrbracket^{\mathfrak{C}}_{\emptyset}, \llbracket \texttt{f} \rrbracket^{\mathfrak{C}}_{\emptyset}) \in \Delta^{\mathfrak{C}}_{\alpha \to \alpha, \rho}$$

# Let's Try another Example, $\texttt{f} :: \alpha \to \alpha$

$$\forall \rho.\ (\llbracket \texttt{f} \rrbracket^{\mathfrak{C}}_{\emptyset}, \llbracket \texttt{f} \rrbracket^{\mathfrak{C}}_{\emptyset}) \in \Delta^{\mathfrak{C}}_{\alpha \to \alpha, \rho}$$

$$\Rightarrow \quad \forall \rho, (\mathbf{x}, \mathbf{y}) \in \Delta^{\mathfrak{C}}_{\alpha, \rho}.\ (\llbracket \texttt{f} \rrbracket^{\mathfrak{C}}_{\emptyset}\ \mathfrak{C}\ \mathbf{x}, \llbracket \texttt{f} \rrbracket^{\mathfrak{C}}_{\emptyset}\ \mathfrak{C}\ \mathbf{y}) \in \Delta^{\mathfrak{C}}_{\alpha, \rho}$$

# Let's Try another Example, $\texttt{f} :: \alpha \to \alpha$

$$
\begin{array}{ll}
 & \forall \rho.\ (\llbracket \texttt{f} \rrbracket^{\mathbb{¢}}_{\emptyset}, \llbracket \texttt{f} \rrbracket^{\mathbb{¢}}_{\emptyset}) \in \Delta^{\mathbb{¢}}_{\alpha \to \alpha, \rho} \\
\Rightarrow & \forall \rho, (\mathbf{x}, \mathbf{y}) \in \Delta^{\mathbb{¢}}_{\alpha, \rho}.\ (\llbracket \texttt{f} \rrbracket^{\mathbb{¢}}_{\emptyset}\ \mathbb{¢}\ \mathbf{x}, \llbracket \texttt{f} \rrbracket^{\mathbb{¢}}_{\emptyset}\ \mathbb{¢}\ \mathbf{y}) \in \Delta^{\mathbb{¢}}_{\alpha, \rho} \\
\Rightarrow & \forall g :: \tau_1 \to \tau_2, (\mathbf{x}, \mathbf{y}) \in \mathcal{C}(R^g). \\
 & (\llbracket \texttt{f} \rrbracket^{\mathbb{¢}}_{\emptyset}\ \mathbb{¢}\ \mathbf{x}, \llbracket \texttt{f} \rrbracket^{\mathbb{¢}}_{\emptyset}\ \mathbb{¢}\ \mathbf{y}) \in \mathcal{C}(R^g)
\end{array}
$$

# Let's Try another Example, $\texttt{f} :: \alpha \to \alpha$

$$
\begin{array}{ll}
 & \forall \rho.\ (\llbracket \texttt{f} \rrbracket^{¢}_{\emptyset}, \llbracket \texttt{f} \rrbracket^{¢}_{\emptyset}) \in \Delta^{¢}_{\alpha \to \alpha, \rho} \\
\Rightarrow & \forall \rho, (\mathbf{x}, \mathbf{y}) \in \Delta^{¢}_{\alpha,\rho}.\ (\llbracket \texttt{f} \rrbracket^{¢}_{\emptyset}\ ¢\ \mathbf{x}, \llbracket \texttt{f} \rrbracket^{¢}_{\emptyset}\ ¢\ \mathbf{y}) \in \Delta^{¢}_{\alpha,\rho} \\
\Rightarrow & \forall g :: \tau_1 \to \tau_2, (\mathbf{x}, \mathbf{y}) \in \mathcal{C}(R^g). \\
 & \quad (\llbracket \texttt{f} \rrbracket^{¢}_{\emptyset}\ ¢\ \mathbf{x}, \llbracket \texttt{f} \rrbracket^{¢}_{\emptyset}\ ¢\ \mathbf{y}) \in \mathcal{C}(R^g) \\
\Rightarrow & \forall g :: \tau_1 \to \tau_2, \mathbf{x} \in \llbracket \tau_1 \rrbracket^{¢}_{\emptyset}. \\
 & \quad (\llbracket \texttt{f} \rrbracket^{¢}_{\emptyset}\ ¢\ (\mathit{appCost}(g, \mathbf{x}) \rhd \mathbf{x}), \\
 & \quad\ \llbracket \texttt{f} \rrbracket^{¢}_{\emptyset}\ ¢\ (\llbracket g \rrbracket^{¢}_{\emptyset}\ ¢\ \mathbf{x})) \in \mathcal{C}(R^g)
\end{array}
$$

# Let's Try another Example, $\mathtt{f} :: \alpha \to \alpha$

$$
\begin{array}{ll}
 & \forall \rho.\ (\llbracket \mathtt{f} \rrbracket^{\mathfrak{c}}_{\emptyset}, \llbracket \mathtt{f} \rrbracket^{\mathfrak{c}}_{\emptyset}) \in \Delta^{\mathfrak{c}}_{\alpha \to \alpha, \rho} \\
\Rightarrow & \forall \rho, (\mathbf{x}, \mathbf{y}) \in \Delta^{\mathfrak{c}}_{\alpha, \rho}.\ (\llbracket \mathtt{f} \rrbracket^{\mathfrak{c}}_{\emptyset}\ \mathfrak{c}\ \mathbf{x}, \llbracket \mathtt{f} \rrbracket^{\mathfrak{c}}_{\emptyset}\ \mathfrak{c}\ \mathbf{y}) \in \Delta^{\mathfrak{c}}_{\alpha, \rho} \\
\Rightarrow & \forall g :: \tau_1 \to \tau_2, (\mathbf{x}, \mathbf{y}) \in \mathcal{C}(R^g). \\
 & \quad (\llbracket \mathtt{f} \rrbracket^{\mathfrak{c}}_{\emptyset}\ \mathfrak{c}\ \mathbf{x}, \llbracket \mathtt{f} \rrbracket^{\mathfrak{c}}_{\emptyset}\ \mathfrak{c}\ \mathbf{y}) \in \mathcal{C}(R^g) \\
\Rightarrow & \forall g :: \tau_1 \to \tau_2, \mathbf{x} \in \llbracket \tau_1 \rrbracket^{\mathfrak{c}}_{\emptyset}. \\
 & \quad (\llbracket \mathtt{f} \rrbracket^{\mathfrak{c}}_{\emptyset}\ \mathfrak{c}\ (\mathit{appCost}(g, \mathbf{x}) \rhd \mathbf{x}), \\
 & \quad\ \llbracket \mathtt{f} \rrbracket^{\mathfrak{c}}_{\emptyset}\ \mathfrak{c}\ (\llbracket g \rrbracket^{\mathfrak{c}}_{\emptyset}\ \mathfrak{c}\ \mathbf{x})) \in \mathcal{C}(R^g)
\end{array}
$$

Does this imply

$$\llbracket g \rrbracket^{\mathfrak{c}}_{\emptyset}\ \mathfrak{c}\ (\llbracket \mathtt{f} \rrbracket^{\mathfrak{c}}_{\emptyset}\ \mathfrak{c}\ \mathbf{x}) = \llbracket \mathtt{f} \rrbracket^{\mathfrak{c}}_{\emptyset}\ \mathfrak{c}\ (\llbracket g \rrbracket^{\mathfrak{c}}_{\emptyset}\ \mathfrak{c}\ \mathbf{x})\,?$$

# Let's Try another Example, $\texttt{f} :: \alpha \to \alpha$

$$\forall g :: \tau_1 \to \tau_2, \mathbf{x} \in [\![\tau_1]\!]^{\mathbb{¢}}_{\emptyset}.$$
$$([\![\texttt{f}]\!]^{\mathbb{¢}}_{\emptyset} \mathbb{¢} (\mathit{appCost}(g, \mathbf{x}) \rhd \mathbf{x}),$$
$$[\![\texttt{f}]\!]^{\mathbb{¢}}_{\emptyset} \mathbb{¢} ([\![g]\!]^{\mathbb{¢}}_{\emptyset} \mathbb{¢} \mathbf{x})) \in \mathcal{C}(R^g)$$

Does this imply

$$[\![g]\!]^{\mathbb{¢}}_{\emptyset} \mathbb{¢} ([\![\texttt{f}]\!]^{\mathbb{¢}}_{\emptyset} \mathbb{¢} \mathbf{x}) = [\![\texttt{f}]\!]^{\mathbb{¢}}_{\emptyset} \mathbb{¢} ([\![g]\!]^{\mathbb{¢}}_{\emptyset} \mathbb{¢} \mathbf{x})\,?$$

Let's see:

$$\mathcal{C}(R^g) = \{(c \rhd \mathit{appCost}(g, \mathbf{x}) \rhd \mathbf{x}, c \rhd \mathbf{y}) \mid [\![g]\!]^{\mathbb{¢}}_{\emptyset} \mathbb{¢} \mathbf{x} = \mathbf{y}, c \in \mathbb{Z}\}$$

# Let's Try another Example, $\texttt{f} :: \alpha \to \alpha$

$$\forall g :: \tau_1 \to \tau_2, \mathbf{x} \in [\![\tau_1]\!]^{\mathfrak{C}}_{\emptyset}.$$
$$([\![\texttt{f}]\!]^{\mathfrak{C}}_{\emptyset} \mathbin{\mathfrak{C}} (\mathit{appCost}(g, \mathbf{x}) \rhd \mathbf{x}),$$
$$[\![\texttt{f}]\!]^{\mathfrak{C}}_{\emptyset} \mathbin{\mathfrak{C}} ([\![g]\!]^{\mathfrak{C}}_{\emptyset} \mathbin{\mathfrak{C}} \mathbf{x})) \in \mathcal{C}(R^g)$$

Does this imply

$$[\![g]\!]^{\mathfrak{C}}_{\emptyset} \mathbin{\mathfrak{C}} ([\![\texttt{f}]\!]^{\mathfrak{C}}_{\emptyset} \mathbin{\mathfrak{C}} \mathbf{x}) = [\![\texttt{f}]\!]^{\mathfrak{C}}_{\emptyset} \mathbin{\mathfrak{C}} ([\![g]\!]^{\mathfrak{C}}_{\emptyset} \mathbin{\mathfrak{C}} \mathbf{x})\,?$$

Let's see:

$$\mathcal{C}(R^g) = \{(c \rhd \mathit{appCost}(g, \mathbf{x}') \rhd \mathbf{x}', c \rhd \mathbf{y}) \mid [\![g]\!]^{\mathfrak{C}}_{\emptyset} \mathbin{\mathfrak{C}} \mathbf{x}' = \mathbf{y}, c \in \mathbb{Z}\}$$

# Let's Try another Example, $\texttt{f} :: \alpha \to \alpha$

$$\forall g :: \tau_1 \to \tau_2, \mathbf{x} \in [\![\tau_1]\!]^{\mathfrak{c}}_{\emptyset}.$$
$$([\![\texttt{f}]\!]^{\mathfrak{c}}_{\emptyset} \mathbin{\mathfrak{c}} (\mathit{appCost}(g, \mathbf{x}) \rhd \mathbf{x}),$$
$$[\![\texttt{f}]\!]^{\mathfrak{c}}_{\emptyset} \mathbin{\mathfrak{c}} ([\![g]\!]^{\mathfrak{c}}_{\emptyset} \mathbin{\mathfrak{c}} \mathbf{x})) \in \mathcal{C}(R^g)$$

Let's see:

$$\mathcal{C}(R^g) = \{(c \rhd \mathit{appCost}(g, \mathbf{x}') \rhd \mathbf{x}', c \rhd \mathbf{y}) \mid [\![g]\!]^{\mathfrak{c}}_{\emptyset} \mathbin{\mathfrak{c}} \mathbf{x}' = \mathbf{y}, c \in \mathbb{Z}\}$$

Actually, the above only imply:

$$\forall \mathbf{x}. \exists \mathbf{x}'. \quad \mathit{appCost}(g, \mathbf{x}) \rhd ([\![g]\!]^{\mathfrak{c}}_{\emptyset} \mathbin{\mathfrak{c}} ([\![\texttt{f}]\!]^{\mathfrak{c}}_{\emptyset} \mathbin{\mathfrak{c}} \mathbf{x})) = \mathit{appCost}(g, \mathbf{x}') \rhd ([\![\texttt{f}]\!]^{\mathfrak{c}}_{\emptyset} \mathbin{\mathfrak{c}} ([\![g]\!]^{\mathfrak{c}}_{\emptyset} \mathbin{\mathfrak{c}} \mathbf{x}))$$

# Let's Try another Example, `f` :: $\alpha \rightarrow \alpha$

Define:

$$R_x^g = \{(val([\![x]\!]_\emptyset^{¢}), val([\![g]\!]_\emptyset^{¢} \; ¢ \; [\![x]\!]_\emptyset^{¢}))\}$$

# Let's Try another Example, `f` :: $\alpha \to \alpha$

Define:

$$R_x^g = \{(val([\![x]\!]_\emptyset^{¢}), val([\![g]\!]_\emptyset^{¢} \; ¢ \; [\![x]\!]_\emptyset^{¢}))\}$$

Then:

$$\mathcal{C}(R_x^g) = \{(c \rhd appCost(g, [\![x]\!]_\emptyset^{¢}) \rhd [\![x]\!]_\emptyset^{¢}, \\ c \rhd ([\![g]\!]_\emptyset^{¢} \; ¢ \; [\![x]\!]_\emptyset^{¢})) \mid c \in \mathbb{Z}\}$$

# Let's Try another Example, $\texttt{f} :: \alpha \to \alpha$

Define:

$$R_x^g = \{(val([\![x]\!]_\emptyset^{¢}), val([\![g]\!]_\emptyset^{¢} \; ¢ \; [\![x]\!]_\emptyset^{¢}))\}$$

Then:

$$\mathcal{C}(R_x^g) = \{(c \rhd appCost(g, [\![x]\!]_\emptyset^{¢}) \rhd [\![x]\!]_\emptyset^{¢}, \\ c \rhd ([\![g]\!]_\emptyset^{¢} \; ¢ \; [\![x]\!]_\emptyset^{¢})) \mid c \in \mathbb{Z}\}$$

Thus:

$$\ldots$$

$$\Rightarrow \quad \forall g :: \tau_1 \to \tau_2, x :: \tau_1, (\mathbf{x}, \mathbf{y}) \in \mathcal{C}(R_x^g). \\ ([\![\texttt{f}]\!]_\emptyset^{¢} \; ¢ \; \mathbf{x}, [\![\texttt{f}]\!]_\emptyset^{¢} \; ¢ \; \mathbf{y}) \in \mathcal{C}(R_x^g)$$

$$\Rightarrow \quad \forall g :: \tau_1 \to \tau_2, x :: \tau_1. \\ ([\![\texttt{f}]\!]_\emptyset^{¢} \; ¢ \; (appCost(g, [\![x]\!]_\emptyset^{¢}) \rhd [\![x]\!]_\emptyset^{¢}), \\ [\![\texttt{f}]\!]_\emptyset^{¢} \; ¢ \; ([\![g]\!]_\emptyset^{¢} \; ¢ \; [\![x]\!]_\emptyset^{¢})) \in \mathcal{C}(R_x^g)$$

# Let's Try another Example, $\texttt{f} :: \alpha \to \alpha$

Then:

$$\mathcal{C}(R_x^g) = \{(c \rhd appCost(g, [\![x]\!]_\emptyset^\Cent) \rhd [\![x]\!]_\emptyset^\Cent, \\ c \rhd ([\![g]\!]_\emptyset^\Cent \Cent [\![x]\!]_\emptyset^\Cent)) \mid c \in \mathbb{Z}\}$$

Thus:

$$\ldots$$

$$\Rightarrow \quad \forall g :: \tau_1 \to \tau_2, x :: \tau_1, (\mathbf{x}, \mathbf{y}) \in \mathcal{C}(R_x^g). \\ ([\![\texttt{f}]\!]_\emptyset^\Cent \Cent \mathbf{x}, [\![\texttt{f}]\!]_\emptyset^\Cent \Cent \mathbf{y}) \in \mathcal{C}(R_x^g)$$

$$\Rightarrow \quad \forall g :: \tau_1 \to \tau_2, x :: \tau_1. \\ ([\![\texttt{f}]\!]_\emptyset^\Cent \Cent (appCost(g, [\![x]\!]_\emptyset^\Cent) \rhd [\![x]\!]_\emptyset^\Cent), \\ [\![\texttt{f}]\!]_\emptyset^\Cent \Cent ([\![g]\!]_\emptyset^\Cent \Cent [\![x]\!]_\emptyset^\Cent)) \in \mathcal{C}(R_x^g)$$

$$\Rightarrow \quad \forall g :: \tau_1 \to \tau_2, x :: \tau_1. \\ [\![g]\!]_\emptyset^\Cent \Cent ([\![\texttt{f}]\!]_\emptyset^\Cent \Cent [\![x]\!]_\emptyset^\Cent) = [\![\texttt{f}]\!]_\emptyset^\Cent \Cent ([\![g]\!]_\emptyset^\Cent \Cent [\![x]\!]_\emptyset^\Cent)$$

# Other Examples

For `f` :: $\alpha \to \alpha \to \alpha$,

$$g\ (\texttt{f}\ x\ y) \sqsubset \texttt{f}\ (g\ x)\ (g\ y)$$

# Other Examples

For $\texttt{f} :: \alpha \to \alpha \to \alpha$,

$$g\ (\texttt{f}\ x\ y) \sqsubset \texttt{f}\ (g\ x)\ (g\ y)$$

For $\texttt{f} :: \alpha \to (\alpha, \alpha)$,

$$\texttt{mapPair}\ (g, g)\ (\texttt{f}\ x) \sqsupset \texttt{f}\ (g\ x)$$

# Other Examples

For $\texttt{f} :: \alpha \to \alpha \to \alpha$,

$$g\ (\texttt{f}\ x\ y) \sqsubset \texttt{f}\ (g\ x)\ (g\ y)$$

For $\texttt{f} :: \alpha \to (\alpha, \alpha)$,

$$\texttt{mapPair}\ (g, g)\ (\texttt{f}\ x) \sqsupset \texttt{f}\ (g\ x)$$

For $\texttt{f} :: [\alpha] \to \mathsf{Nat}$,

$$\texttt{f}\ l \sqsubset \texttt{f}\ (\texttt{map}\ g\ l)$$

# Other Examples

For $\texttt{f} :: \alpha \to \alpha \to \alpha$,

$$g\ (\texttt{f}\ x\ y) \sqsubset \texttt{f}\ (g\ x)\ (g\ y)$$

For $\texttt{f} :: \alpha \to (\alpha, \alpha)$,

$$\texttt{mapPair}\ (g, g)\ (\texttt{f}\ x) \sqsupset \texttt{f}\ (g\ x)$$

For $\texttt{f} :: [\alpha] \to \text{Nat}$,

$$\texttt{f}\ l \sqsubset \texttt{f}\ (\texttt{map}\ g\ l)$$

For $\texttt{f} :: [\alpha] \to [\alpha]$, get conditional statements about relative efficiency of $\texttt{map}\ g\ (\texttt{f}\ l)$ and $\texttt{f}\ (\texttt{map}\ g\ l)$.

# Conclusion

We did:

- develop of “cost-aware” logical relation and its parametricity theorem

# Conclusion

We did:

- develop of “cost-aware” logical relation and its parametricity theorem
- realize that more effort has to go into the actual deriving of concrete free theorems

# Conclusion

We did:

- develop of “cost-aware” logical relation and its parametricity theorem
- realize that more effort has to go into the actual deriving of concrete free theorems
- establish a number of building blocks/heuristics for that

# Conclusion

We did:

- develop of "cost-aware" logical relation and its parametricity theorem
- realize that more effort has to go into the actual deriving of concrete free theorems
- establish a number of building blocks/heuristics for that

Further plans:

- apply to automatic program transformations
- extend to full recursion, other language features
- mechanize derivation of cost-aware statements
- investigate call-by-name/call-by-need
- use more realistic cost measures?

# References I



J.-P. Bernardy, P. Jansson, and K. Claessen.
Testing polymorphic properties.
In *European Symposium on Programming, Proceedings*, volume 6012 of *LNCS*, pages 125–144. Springer-Verlag, 2010.


B. Bjerner and S. Holmström.
A compositional approach to time analysis of first order lazy functional programs.
In *Functional Programming Languages and Computer Architecture, Proceedings*, pages 157–165. ACM Press, 1989.


J.P. Fernandes, A. Pardo, and J. Saraiva.
A shortcut fusion rule for circular program calculation.
In *Haskell Workshop, Proceedings*, pages 95–106. ACM Press, 2007.

# References II



A. Gill, J. Launchbury, and S.L. Peyton Jones.
A short cut to deforestation.
In *Functional Programming Languages and Computer Architecture, Proceedings*, pages 223–232. ACM Press, 1993.


Y. Liu and G. Gómez.
Automatic accurate cost-bound analysis for high-level languages.
*IEEE Transactions on Computers*, 50(12):1295–1309, 2001.


J.C. Reynolds.
Types, abstraction and parametric polymorphism.
In *Information Processing, Proceedings*, pages 513–523. Elsevier, 1983.

# References III


M. Rosendahl.
Automatic complexity analysis.
In *Functional Programming Languages and Computer Architecture, Proceedings*, pages 144–156. ACM Press, 1989.

D. Sands.
A naïve time analysis and its theory of cost equivalence.
*Journal of Logic and Computation*, 5(4):495–541, 1995.

J. Svenningsson.
Shortcut fusion for accumulating parameters & zip-like functions.
In *International Conference on Functional Programming, Proceedings*, pages 124–132. ACM Press, 2002.

# References IV



P. Wadler.
Strictness analysis aids time analysis.
In *Principles of Programming Languages, Proceedings*, pages 119–132. ACM Press, 1988.


P. Wadler.
Theorems for free!
In *Functional Programming Languages and Computer Architecture, Proceedings*, pages 347–359. ACM Press, 1989.