The Lifting Lemma

Ralf Hinze

Computing Laboratory, University of Oxford
Wolfson Building, Parks Road, Oxford, OX1 3QD, England
ralf.hinze@comlab.ox.ac.uk
http://www.comlab.ox.ac.uk/ralf.hinze/

June 2009

Mathematicians do it . . .

$$(f + g)(x) = f(x) + g(x)$$

. . . over and . . .

$$A + B = \{ a + b \mid a \in A, \ b \in B \}$$

. . . and over again.

$$(x_1, x_2, x_3) + (y_1, y_2, y_3) = (x_1 + y_1, x_2 + y_2, x_3 + y_3)$$

Haskell programmers do it . . .

$$\mathbf{data}\ \mathsf{Maybe}\ \alpha = \mathsf{Nothing} \mid \mathsf{Just}\ \alpha$$

$$\begin{aligned}
(+) &:: \mathsf{Maybe}\ \mathbb{N} \to \mathsf{Maybe}\ \mathbb{N} \to \mathsf{Maybe}\ \mathbb{N} \\
\mathsf{Nothing} + \mathsf{n} &= \mathsf{Nothing} \\
\mathsf{m} + \mathsf{Nothing} &= \mathsf{Nothing} \\
\mathsf{Just}\ \mathsf{a} + \mathsf{Just}\ \mathsf{b} &= \mathsf{Just}\ (\mathsf{a} + \mathsf{b})
\end{aligned}$$

... over and over again.

$$(+) \quad :: \mathrm{IO}\ \mathbb{N} \to \mathrm{IO}\ \mathbb{N} \to \mathrm{IO}\ \mathbb{N}$$
$$m + n = \mathbf{do}\ \{a \leftarrow m; b \leftarrow n; \mathrm{return}\ (a + b)\}$$

I do it: lifting

$$\mathbf{data}\ \mathsf{Stream}\ \alpha = \mathsf{Cons}\ \{\mathsf{head} :: \alpha, \mathsf{tail} :: \mathsf{Stream}\ \alpha\}$$

$$\begin{aligned} (+) \quad &:: \mathsf{Stream}\ \mathbb{N} \to \mathsf{Stream}\ \mathbb{N} \to \mathsf{Stream}\ \mathbb{N} \\ \mathsf{s} + \mathsf{t} &= \mathsf{Cons}\ (\mathsf{head}\ \mathsf{s} + \mathsf{head}\ \mathsf{t})\ (\mathsf{tail}\ \mathsf{s} + \mathsf{tail}\ \mathsf{t}) \end{aligned}$$

Since the arithmetic operations are defined point-wise, the familiar arithmetic laws also hold for streams.

More general, given a point-level identity, does the lifted version hold as well?

$$(x + y) + z = x + (y + z)$$

$$x + y = y + x$$

$$x * 0 = 0$$

Idioms

Idioms aka applicative functors

$$
\begin{array}{l}
\textbf{class}\ \mathrm{Idiom}\ \iota\ \textbf{where} \\
\quad \mathrm{pure} :: \alpha \to \iota\ \alpha \\
\quad (\diamond) \quad :: \iota\ (\alpha \to \beta) \to (\iota\ \alpha \to \iota\ \beta)
\end{array}
$$

$$
\begin{array}{l}
\textbf{instance}\ \mathrm{Idiom}\ (\tau\rightarrow)\ \textbf{where} \\
\quad \mathrm{pure}\ a = \lambda x \rightarrow a \\
\quad f \diamond g \quad = \lambda x \rightarrow (f\ x)\ (g\ x)
\end{array}
$$

$$\begin{aligned}
&\textbf{instance}\ \mathsf{Idiom}\ (\tau\rightarrow)\ \textbf{where} \\
&\quad \mathsf{pure}\ a = \lambda x \rightarrow a \\
&\quad f \diamond g \quad = \lambda x \rightarrow (f\ x)\ (g\ x)
\end{aligned}$$

So, pure is the $\mathbb{K}$ combinator and $\diamond$ is the $\mathbb{S}$ combinator.

```
instance Idiom Stream where
  pure a = s where s = a ≺ s
  s ⋄ t    = Cons ((head s) (head t)) (tail s ⋄ tail t)
```

Lifting, generically

$$
\begin{aligned}
&(+) \quad :: (\mathrm{Idiom}\ \iota) \Rightarrow \iota\ \mathbb{N} \to \iota\ \mathbb{N} \to \iota\ \mathbb{N} \\
&u + v = \mathrm{pure}\ (+) \diamond u \diamond v
\end{aligned}
$$

$$
\begin{aligned}
&(\star) \quad :: (\mathrm{Idiom}\ \iota) \Rightarrow \iota\ \alpha \to \iota\ \beta \to \iota\ (\alpha, \beta) \\
&u \star v = \mathrm{pure}\ (,) \diamond u \diamond v
\end{aligned}
$$

Idiom laws

$$\begin{aligned}
\text{pure id} \diamond u &= u && \text{(identity)} \\
\text{pure } (\cdot) \diamond u \diamond v \diamond w &= u \diamond (v \diamond w) && \text{(composition)} \\
\text{pure } f \diamond \text{pure } x &= \text{pure } (f\ x) && \text{(homomorphism)} \\
u \diamond \text{pure } x &= \text{pure } (\diamond x) \diamond u && \text{(interchange)}
\end{aligned}$$

$$
\begin{aligned}
& \mathrm{pure}\ f \diamond (u \star v) \\
= \quad & \{\ \text{definition of } \star\ \} \\
& \mathrm{pure}\ f \diamond (\mathrm{pure}\ (,) \diamond u \diamond v) \\
= \quad & \{\ \text{idiom composition}\ \} \\
& \mathrm{pure}\ (\cdot) \diamond \mathrm{pure}\ f \diamond (\mathrm{pure}\ (,) \diamond u) \diamond v \\
= \quad & \{\ \text{idiom homomorphism}\ \} \\
& \mathrm{pure}\ (f\ \cdot) \diamond (\mathrm{pure}\ (,) \diamond u) \diamond v \\
= \quad & \{\ \text{idiom composition}\ \} \\
& \mathrm{pure}\ (\cdot) \diamond \mathrm{pure}\ (f\ \cdot) \diamond \mathrm{pure}\ (,) \diamond u \diamond v \\
= \quad & \{\ \text{idiom homomorphism}\ \} \\
& \mathrm{pure}\ ((f\ \cdot) \cdot (,)) \diamond u \diamond v \\
= \quad & \{\ ((f\ \cdot) \cdot (,))\ x\ y = (f \cdot (,)\ x)\ y = f\ ((,)\ x\ y) = \mathrm{curry}\ f\ x\ y\ \} \\
& \mathrm{pure}\ (\mathrm{curry}\ f) \diamond u \diamond v
\end{aligned}
$$

The Idiomatic Calculus

Syntax: variables

$$
\begin{aligned}
&\mathbf{data}\ \mathrm{Ix} :: * \to * \to *\ \ \mathbf{where}\\
&\quad \mathrm{Zero} :: \mathrm{Ix}\ (\rho, \alpha)\ \alpha\\
&\quad \mathrm{Succ} :: \mathrm{Ix}\ \rho\ \beta \to \mathrm{Ix}\ (\rho, \alpha)\ \beta
\end{aligned}
$$

Syntax: terms

$$
\begin{aligned}
&\textbf{data}\ \mathrm{Term} :: * \to * \to *\ \ \textbf{where}\\
&\quad \mathrm{Con} :: \alpha \to \mathrm{Term}\ \rho\ \alpha\\
&\quad \mathrm{Var}\ \ :: \mathrm{Ix}\ \rho\ \alpha \to \mathrm{Term}\ \rho\ \alpha\\
&\quad \mathrm{App} :: \mathrm{Term}\ \rho\ (\alpha \to \beta) \to \mathrm{Term}\ \rho\ \alpha \to \mathrm{Term}\ \rho\ \beta
\end{aligned}
$$

Semantics: variables

$$\begin{aligned}
&\mathbf{data}\ \mathrm{Env} :: (* \to *) \to (* \to *)\ \mathbf{where} \\
&\quad \mathrm{Empty} :: \mathrm{Env}\ \iota\ () \\
&\quad \mathrm{Push}\ \ :: \mathrm{Env}\ \iota\ \rho \to \iota\ \alpha \to \mathrm{Env}\ \iota\ (\rho, \alpha)
\end{aligned}$$

We write Empty as $\langle\rangle$ and Push η u as $\langle \eta, u \rangle$.

$$\begin{aligned}
&\mathrm{acc} && :: \mathrm{Ix}\ \rho\ \alpha \to \mathrm{Env}\ \iota\ \rho \to \iota\ \alpha \\
&\mathrm{acc}\ \mathrm{Zero} && \langle \eta, v \rangle = v \\
&\mathrm{acc}\ (\mathrm{Succ}\ n) && \langle \eta, v \rangle = \mathrm{acc}\ n\ \eta
\end{aligned}$$

Semantics: terms

$$
\begin{aligned}
\mathcal{I}[\![\,]\!] \quad &:: (\mathrm{Idiom}\ \iota) \Rightarrow \mathrm{Term}\ \rho\ \alpha \to \mathrm{Env}\ \iota\ \rho \to \iota\ \alpha \\
\mathcal{I}[\![\mathrm{Con}\ v]\!]\eta &= \mathrm{pure}\ v \\
\mathcal{I}[\![\mathrm{Var}\ n]\!]\eta &= \mathrm{acc}\ n\ \eta \\
\mathcal{I}[\![\mathrm{App}\ e_1\ e_2]\!]\eta &= \mathcal{I}[\![e_1]\!]\eta \diamond \mathcal{I}[\![e_2]\!]\eta
\end{aligned}
$$

What about abstraction?

Syntax

$$
\begin{aligned}
&\textbf{data}\ \mathrm{Term} :: * \to * \to * \ \textbf{where} \\
&\quad \mathrm{Con} :: \alpha \to \mathrm{Term}\ \rho\ \alpha \\
&\quad \mathrm{Var}\ \ :: \mathrm{Ix}\ \rho\ \alpha \to \mathrm{Term}\ \rho\ \alpha \\
&\quad \mathrm{App} :: \mathrm{Term}\ \rho\ (\alpha \to \beta) \to \mathrm{Term}\ \rho\ \alpha \to \mathrm{Term}\ \rho\ \beta \\
&\quad \mathrm{Abs}\ :: \mathrm{Term}\ (\rho, \alpha)\ \beta \to \mathrm{Term}\ \rho\ (\alpha \to \beta)
\end{aligned}
$$

An idiom is very similar to an *applicative structure*.

Semantics

$$
\begin{aligned}
\mathcal{I}[\![\,]\!] &:: (\mathrm{Idiom}\ \iota) \Rightarrow \mathrm{Term}\ \rho\ \alpha \to \mathrm{Env}\ \iota\ \rho \to \iota\ \alpha \\
\mathcal{I}[\![\mathrm{Con}\ v]\!]\eta &= \mathrm{pure}\ v \\
\mathcal{I}[\![\mathrm{Var}\ n]\!]\eta &= \mathrm{acc}\ n\ \eta \\
\mathcal{I}[\![\mathrm{App}\ e_1\ e_2]\!]\eta &= \mathcal{I}[\![e_1]\!]\eta \diamond \mathcal{I}[\![e_2]\!]\eta \\
\mathcal{I}[\![\mathrm{Abs}\ e]\!]\eta &= \text{the unique function } f \text{ such that} \\
&\quad \forall v\ .\ f \diamond v = \mathcal{I}[\![e]\!]\langle \eta, v \rangle
\end{aligned}
$$

Uniqueness?

Extensionality:

$$(\forall u \,.\, f \diamond u = g \diamond u) \Longrightarrow f = g$$

Existence?

Combinatory model condition:

$$\begin{aligned} \mathrm{pure}\ \mathbb{K} \diamond u \diamond v &= u \\ \mathrm{pure}\ \mathbb{S} \diamond u \diamond v \diamond w &= (u \diamond w) \diamond (v \diamond w) \end{aligned}$$

Ensures that ι has enough points.

Idiomatic interpretation specialised to the identity idiom:

$$
\begin{aligned}
[\![\,]\!] \quad & :: \ \mathrm{Term}\ \rho\ \alpha \to \mathrm{Env}\ \mathrm{Id}\ \rho \to \alpha \\
[\![\mathrm{Con}\ v]\!]\eta &= v \\
[\![\mathrm{Var}\ n]\!]\eta &= \mathrm{acc}\ n\ \eta \\
[\![\mathrm{App}\ e_1\ e_2]\!]\eta &= ([\![e_1]\!]\eta)\ ([\![e_2]\!]\eta) \\
[\![\mathrm{Abs}\ e]\!]\eta &= \lambda v \to [\![e]\!]\langle \eta, v\rangle
\end{aligned}
$$

. . . written in a pointfree style:

$$
\begin{aligned}
[\![]\!] &:: \mathrm{Term}\ \rho\ \alpha \rightarrow \mathrm{Env\ Id}\ \rho \rightarrow \alpha \\
[\![\mathrm{Con}\ v]\!] &= \mathbb{K}\ v \\
[\![\mathrm{Var}\ n]\!] &= \mathrm{acc}\ n \\
[\![\mathrm{App}\ e_1\ e_2]\!] &= \mathbb{S}\ [\![e_1]\!]\ [\![e_2]\!] \\
[\![\mathrm{Abs}\ e]\!] &= \mathrm{curry}\ [\![e]\!]
\end{aligned}
$$

The Lifting Lemma

The lifting lemma

$$\mathcal{I}[\![\mathrm{e}]\!] = \mathrm{pure}\ [\![\mathrm{e}]\!]$$

$$
\begin{aligned}
& \mathsf{pure}\ (+) \diamond (\mathsf{pure}\ (*) \diamond u \diamond w) \diamond (\mathsf{pure}\ (*) \diamond v \diamond w) \\
= \quad & \{ \text{definition of } \mathcal{I} \} \\
& \mathcal{I}[\![\mathsf{Abs}\ (\mathsf{Abs}\ (\mathsf{Abs}\ (2 * 0 + 1 * 0)))]\!] \diamond u \diamond v \diamond w \\
= \quad & \{ \text{lifting lemma} \} \\
& \mathsf{pure}\ [\![\mathsf{Abs}\ (\mathsf{Abs}\ (\mathsf{Abs}\ (2 * 0 + 1 * 0)))]\!] \diamond u \diamond v \diamond w \\
= \quad & \{ \text{definition of } [\![]\!] \} \\
& \mathsf{pure}\ (\lambda x\ y\ z \to x * z + y * z) \diamond u \diamond v \diamond w \\
= \quad & \{ \text{arithmetic} \} \\
& \mathsf{pure}\ (\lambda x\ y\ z \to (x + y) * z) \diamond u \diamond v \diamond w \\
= \quad & \{ \text{definition of } [\![]\!] \} \\
& \mathsf{pure}\ [\![\mathsf{Abs}\ (\mathsf{Abs}\ (\mathsf{Abs}\ ((2 + 1) * 0)))]\!] \diamond u \diamond v \diamond w \\
= \quad & \{ \text{lifting lemma} \} \\
& \mathcal{I}[\![\mathsf{Abs}\ (\mathsf{Abs}\ (\mathsf{Abs}\ ((2 + 1) * 0)))]\!] \diamond u \diamond v \diamond w \\
= \quad & \{ \text{definition of } \mathcal{I} \} \\
& \mathsf{pure}\ (*) \diamond (\mathsf{pure}\ (+)\ u\ v) \diamond w
\end{aligned}
$$

The lifting lemma, general form

$$\mathcal{I}[\![e]\!]\eta = \mathrm{pure}\ [\![e]\!] \diamond \mathrm{zip}\ \eta$$

$$\begin{aligned}
&\mathrm{zip} && :: (\mathrm{Idiom}\ \iota) \Rightarrow \mathrm{Env}\ \iota\ \rho \rightarrow \iota\ (\mathrm{Env\ Id}\ \rho)\\
&\mathrm{zip}\ \langle\rangle && = \mathrm{pure}\ \langle\rangle\\
&\mathrm{zip}\ \langle\eta, v\rangle && = \mathrm{pure}\ \langle,\rangle \diamond \eta \diamond v
\end{aligned}$$

Proof: Case $e = \mathsf{Con}\ v$:

$$
\begin{aligned}
& \mathsf{pure}\ [\![\mathsf{Con}\ v]\!] \diamond \mathsf{zip}\ \eta \\
= \quad & \{\ \text{definition of } [\![\,]\!]\ \} \\
& \mathsf{pure}\ (\mathbb{K}\ v) \diamond \mathsf{zip}\ \eta \\
= \quad & \{\ \text{idiom homomorphism}\ \} \\
& \mathsf{pure}\ \mathbb{K} \diamond \mathsf{pure}\ v \diamond \mathsf{zip}\ \eta \\
= \quad & \{\ \text{combinatory model condition I}\ \} \\
& \mathsf{pure}\ v \\
= \quad & \{\ \text{definition of } \mathcal{I}\ \} \\
& \mathcal{I}[\![\mathsf{Con}\ v]\!]\eta
\end{aligned}
$$

Proof: Case $e = \mathsf{Var}\ n$:

$$
\begin{aligned}
& \mathrm{pure}\ [\![\mathsf{Var}\ n]\!] \diamond \mathrm{zip}\ \eta \\
= \quad & \{\ \text{definition of } [\![\,]\!]\ \} \\
& \mathrm{pure}\ (\mathrm{acc}\ n) \diamond \mathrm{zip}\ \eta \\
= \quad & \{\ \text{Lemma: } \mathrm{pure}\ (\mathrm{acc}\ n) \diamond \mathrm{zip}\ \eta = \mathrm{acc}\ n\ \eta\ \} \\
& \mathrm{acc}\ n\ \eta \\
= \quad & \{\ \text{definition of } \mathcal{I}\ \} \\
& \mathcal{I}[\![\mathsf{Var}\ n]\!]\eta
\end{aligned}
$$

Proof: Case $e = \mathsf{App}\ e_1\ e_2$:

$$
\begin{aligned}
& \mathsf{pure}\ [\![\mathsf{App}\ e_1\ e_2]\!] \diamond \mathsf{zip}\ \eta \\
= \quad & \{\ \text{definition of } [\![\,]\!]\ \} \\
& \mathsf{pure}\ (\mathbb{S}\ [\![e_1]\!]\ [\![e_2]\!]) \diamond \mathsf{zip}\ \eta \\
= \quad & \{\ \text{idiom homomorphism}\ \} \\
& \mathsf{pure}\ \mathbb{S} \diamond \mathsf{pure}\ [\![e_1]\!] \diamond \mathsf{pure}\ [\![e_2]\!] \diamond \mathsf{zip}\ \eta \\
= \quad & \{\ \text{combinatory model condition II}\ \} \\
& (\mathsf{pure}\ [\![e_1]\!] \diamond \mathsf{zip}\ \eta) \diamond (\mathsf{pure}\ [\![e_2]\!] \diamond \mathsf{zip}\ \eta) \\
= \quad & \{\ \text{ex hypothesi}\ \} \\
& \mathcal{I}[\![e_1]\!]\eta \diamond \mathcal{I}[\![e_2]\!]\eta \\
= \quad & \{\ \text{definition of } \mathcal{I}\ \} \\
& \mathcal{I}[\![\mathsf{App}\ e_1\ e_2]\!]\eta
\end{aligned}
$$

Proof: Case $e = \mathsf{Abs}\ e$:

$$
\begin{aligned}
& \mathsf{pure}\ [\![\mathsf{Abs}\ e]\!] \diamond \mathsf{zip}\ \eta \\
= \quad & \{\ \text{definition of } [\![\,]\!]\ \} \\
& \mathsf{pure}\ (\mathsf{curry}\ [\![e]\!]) \diamond \mathsf{zip}\ \eta \\
= \quad & \{\ \text{proof obligation (see next slide)}\ \} \\
& f \\
= \quad & \{\ \text{definition of } \mathcal{I}\ \} \\
& \mathcal{I}[\![\mathsf{Abs}\ e]\!]\eta
\end{aligned}
$$

Proof obligation

$$
\begin{aligned}
& \mathrm{pure}\ (\mathrm{curry}\ [\![e]\!]) \diamond \mathrm{zip}\ \eta = f \\
\iff & \quad \{\ \text{extensionality}\ \} \\
& \mathrm{pure}\ (\mathrm{curry}\ [\![e]\!]) \diamond \mathrm{zip}\ \eta \diamond v = f \diamond v \\
\iff & \quad \{\ \text{definition of f}\ \} \\
& \mathrm{pure}\ (\mathrm{curry}\ [\![e]\!]) \diamond \mathrm{zip}\ \eta \diamond v = \mathcal{I}[\![e]\!]\langle \eta, v \rangle \\
\iff & \quad \{\ \text{curry-lemma (see above)}\ \} \\
& \mathrm{pure}\ [\![e]\!] \diamond (\mathrm{zip}\ \eta \star v) = \mathcal{I}[\![e]\!]\langle \eta, v \rangle \\
\iff & \quad \{\ \text{definition of zip}\ \} \\
& \mathrm{pure}\ [\![e]\!] \diamond (\mathrm{zip}\ \langle \eta, v \rangle) = \mathcal{I}[\![e]\!]\langle \eta, v \rangle \\
\iff & \quad \{\ \text{ex hypothesi}\ \} \\
& \mathrm{True}
\end{aligned}
$$

What about . . .

- $\tau\rightarrow$: $\surd$;
- Set: ↯, but λI-calculus;
- Vector: $\surd$;
- Maybe: ↯, but λI-calculus;
- IO: ↯, but NF Lemma;
- Stream: $\surd$.