Isomorphisms for context-free types

joint work with Wouter Swierstra

Thorsten Altenkirch

School of Computer Science and IT
University of Nottingham

January 18, 2007

What is an isomorphism?

Given types A, B, an isomorphism is given by 2 functions

$$\begin{aligned} \phi &\in A \to B \\ \psi &\in B \to A \end{aligned}$$

s.t.

$$\begin{aligned} \psi \circ \phi &= \mathrm{id}_A \\ \phi \circ \psi &= \mathrm{id}_b \end{aligned}$$

We say that A and B are **isomorphic** ($A \simeq B$), if there is an isomorphism between them.

Examples:

- $\mathbb{N} \simeq \mathbb{N} + \mathbb{N}$
- $\mathbb{N} \simeq \mathbb{N} \times \mathbb{N}$
- $\mathbb{N} \not\simeq \mathbb{N} \to \mathbb{N}$

Type variables

Many interesting isomorphisms involve type variables, e.g.

$$\mathrm{List}\,(1 + X) \simeq \mathrm{List}\,X \times \mathrm{List}\,(\mathrm{List}\,X)$$

Types with variables (no $\rightarrow$) give rise to functors F:

$$\frac{A \in \mathbf{Type}}{F\,A \in \mathbf{Type}} \qquad \frac{f \in A \rightarrow B}{F\,f \in F\,A \rightarrow F\,B}$$

such that

$$\begin{aligned} F\,\mathrm{id}_A &= \mathrm{id}_{F\,A} \\ F\,(f \circ g) &= F\,f \circ F\,g \end{aligned}$$

Isomorphisms with variables

A **natural isomorphism** between functors F, G is given by an assignment:

$$\frac{A \in \textbf{Type}}{\begin{array}{c}\Phi_A \in F\,A \to G\,A \\ \Psi_A \in G\,A \to F\,A\end{array}}$$

such that

- Φ_A, Ψ_A are an isomorphism between $F\,A$ and $G\,A$.
- The assignment is *natural*, for any function $f \in A \to B$ we have that

$$\begin{array}{rcl} G\,f \circ \Psi_A & = & \Psi_A \circ F\,f \\ F\,f \circ \Phi_A & = & \Phi_A \circ G\,f \end{array}$$

Exercise: Show that we only need one of the two equations.

We write $F \simeq G$ if there is a natural isomorphism between the functors F, G.

Why study isomorphisms?

- Curry-Howard correspondence:

Proofs	$\sim$	Programs
Propositions	$\sim$	Types
Logical Equivalence	$\sim$	Isomorphism

- We can replace any type in our program by an isomorphic type
  (change of representation).
- We can replace any type operator by a naturally isomorphic operator.
- When searching for data by type, we may only want to specify the type upto isomorphism.

$\mathrm{List}\,(1 + X) \simeq \mathrm{List}\,X \times \mathrm{List}\,(\mathrm{List}\,X)$

$$
\begin{array}{l}
\phi :: [\mathit{Maybe}\ a] \to ([a], [[a]]) \\
\phi\ [\,] \qquad\qquad\quad = ([\,], [\,]) \\
\phi\ (\mathit{ma} : \mathit{mas}) = \\
\quad \textbf{case}\ \mathit{ma}\ \textbf{of} \\
\quad\quad \mathit{Nothing} \to ([\,], \mathit{as} : \mathit{aas}) \\
\quad\quad \mathit{Just}\ a \quad\ \to (a : \mathit{as}, \mathit{aas}) \\
\quad \textbf{where}\ (\mathit{as}, \mathit{aas}) = \phi\ \mathit{mas}
\end{array}
$$

$$
\begin{array}{l}
\psi :: ([a], [[a]]) \to [\mathit{Maybe}\ a] \\
\psi\ ([\,], [\,]) \qquad\quad = [\,] \\
\psi\ ([\,], \mathit{as} : \mathit{aas}) = \mathit{Nothing} : \psi\ (\mathit{as}, \mathit{aas}) \\
\psi\ (b : \mathit{bs}, \mathit{aas})\ \ = \mathit{Just}\ b : \psi\ (\mathit{bs}, \mathit{aas})
\end{array}
$$

Naturality? All polymorphic functions definable in Haskell are natural.

How to prove non-isomorphisms?

$$\mathrm{List}\, X \not\simeq \mathrm{List}\, X \times \mathrm{List}\, X$$

What are context-free types?

Given a finite set of parameters P and a finite set of of recursive variables X we define the set of context-free types $\mathrm{CF}_X\, P$ inductively by the following rules:

$$\frac{p \in P}{p \in \mathrm{CF}_X\, P} \qquad \frac{x \in X}{x \in \mathrm{CF}_X\, P}$$

$$\frac{}{0, 1 \in \mathrm{CF}_X\, P} \qquad \frac{\sigma, \tau \in \mathrm{CF}_X\, P}{\begin{array}{c}\sigma + \tau \in \mathrm{CF}_X\, P \\ \sigma \times \tau \in \mathrm{CF}_X\, P\end{array}}$$

$$\frac{\sigma \in \mathrm{CF}_{x+X}\, P}{\mu x.\sigma \in \mathrm{CF}_X\, P}$$

We write $\mathrm{CF}\, P$ for $\mathrm{CF}_\emptyset\, P$.

Examples of context-free types

Natural numbers

$$\mathbb{N} = \mu X.1 + X \in \mathrm{CF}\,\emptyset$$

Lists

$$\mathrm{List}\,A = \mu X.1 + A \times X \in \mathrm{CF}\,\{A\}$$

Binary trees

$$\begin{aligned} \mathrm{BT}\,A\,B &= \\ \mu X.A + B \times X^2 &= \\ \mu X.A + B \times X \times X &\in \mathrm{CF}\,\{A, B\} \end{aligned}$$

Spine trees

$$\begin{aligned} \mathrm{ST}\,A\,B &= \\ \mu X.B \times \mathrm{List}\,(A \times X) &= \\ \mu X.B \times \mu Y.1 + (A \times X) \times Y &\in \mathrm{CF}\,\{A, B\} \end{aligned}$$

Exercise: Show that $\mathrm{BT} \simeq \mathrm{ST}$.

Grammars vs Types

Context-free types	Context-free grammars
parameters	terminal symbols
recursive variables	non-terminal symbols
$\sigma + \tau$	$v + w$
$\sigma \times \tau$	vw
isomorphism ($\simeq$)	language equivalence ($\sim_L$) ???

Isomorphism vs. language equivalence

Commutativity of $\times$

$$\sigma \times \tau \simeq \tau \times \sigma$$

but

$$V \times W \not\sim_L W \times V$$

Idempotence of $+$

$$V + V \sim_L V$$

but

$$\sigma + \sigma \not\simeq \sigma$$

Finite sets and multisets

$$
\begin{array}{lcl}
\mathcal{P}_{<\omega}\,A & = & \text{finite sets over } A \\
\mathbb{N}^{+} & = & \mathbb{N} + \{\omega\} \\
\mathcal{M}\,A & = & \text{finite multi-sets over } A \\
\mathcal{M}^{+}\,A & = & \text{finite multi-sets using } \mathbb{N}^{+} \text{ instead of } \mathbb{N}.
\end{array}
$$

Parsing languages and multisets

Given $\sigma \in \mathrm{CF}\, P$:

Languages

Parser

$$[\![\sigma]\!] \in \mathrm{List}\, P \to \mathrm{Bool}$$

Partial parser

$$[\![\sigma]\!]^{\mathrm{L}}_{\mathrm{partial}} \in \mathrm{List}\, P \to \mathcal{P}_{<\omega}\,(\mathrm{List}\, P)$$

Multisets

Parser

$$[\![\sigma]\!]^{\mathrm{M}} \in \mathcal{M}\, P \to \mathbb{N}^{+}$$

Partial parser

$$[\![\sigma]\!]^{\mathrm{M}}_{\mathrm{partial}} \in \mathcal{M}\, P \to \mathcal{M}^{+}\,(\mathcal{M}\, P)$$

Relating multisets and types

For simplicity let $P = A$ then $[\![\sigma]\!]^{\mathrm{M}} \in \mathbb{N} \to \mathbb{N}^{+}$.

We can recover the typetheoretic interpretation of σ as a functor

$$[\![\sigma]\!]^{\mathrm{F}} \in \textbf{Type} \to \textbf{Type}$$

by

$$[\![\sigma]\!]^{\mathrm{F}}\, X = \Sigma i \in \mathbb{N}.[\![\sigma]\!]^{\mathrm{M}} \times X^{i}$$

Theorem:

$$[\![\sigma]\!]^{\mathrm{F}} \simeq [\![\tau]\!]^{\mathrm{F}} \iff [\![\sigma]\!]^{\mathrm{M}} = [\![\tau]\!]^{\mathrm{M}}$$

Corollary: Isomorphism of context-free types is semidecidable.

Sketch of the proof

1. Define a notion of morphisms between the multi-set interpretation $\mathbb{N} \to \mathbb{N}^+$ giving rise to a category.
2. Show that two objects in this category are isomorphic iff they are equal.
   The category is skeletal.
3. Every morphism gives rise to a natural transformation between the associated functors.
4. Vice versa: every natural transformaton gives rise to a morphism.
   The interpretation is full and faithful.

Questions

- Is isomorphism between context-free types decidable?
- Is isomorphism between regular types (only List instead of μ) decidable?
- Have Kleene algebras with commutative $\times$ and non-idempotent $+$ been studied?