Isomorphisms on Generic Mutually recursive polynomial types

Some Theory and an application

Marcelo Fiore
University of Cambridge

Joint work with Tom Leinster

Toulouse 28.X.05

Richness of the subject

- Mathematical logic
- Type Theory
- Programming-language Theory
- Rewriting Theory
- Number Theory
- Category Theory
- Combinatorics
- Computational algebra
- Algorithmics

The setting for this talk

Type Theory

$$\tau ::= X \mid \underbrace{\tau_1 + \tau_2 \mid \tau_1 \times \tau_2}_{\text{polynomial types}} \mid 0 \mid 1$$

$$\underbrace{\mu X.\tau}_{\text{generic recursive types}}$$

Lines of investigation

- Characterise the equational theory of type isomorphism
- Study the structure of isomorphisms

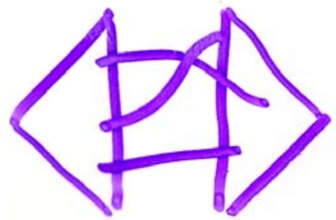
Isomorphisms between generic mutually recursive polynomial types

Example:

<u>datatype</u> $E = b$ <u>of</u> $E * E$

<u>fun</u> $p_0(x) =$ <u>case</u> x <u>of</u>
$b(x_1, x_2) \Rightarrow$ <u>case</u> x_1 <u>of</u>
$b(x_{11}, x_{12}) \Rightarrow$
$b(x_{11}, b(x_{12}, x_2))$

$$p_0(b(b(x_{11}, x_{12}), x_2)) = b(x_{11}, b(x_{12}, x_2))$$



Characterisation of type isomorphism

Thm: Two generic mutually recursive polynomial types are isomorphic iff they are provable equal in the equational theory of commutative semirings augmented with the equations $\mu x.\tau = \tau[\mu x.\tau / x]$.

Categorical version

Thm: The groupoid underlying

$$\mathcal{D}[x_1, \ldots, x_n] / (x_1 \cong p_1(x_1 \ldots x_n), \ldots, x_n \cong p_n(x_1 \ldots x_n))$$

is

$$\mathcal{R}[x_1 \ldots x_n] / (x_1 \cong p_1(x_1 \ldots x_n), \ldots, x_n \cong p_n(x_1 \ldots x_n))$$

with Burnside semiring

$$\mathbb{N}[x_1 \ldots x_n] / (x_1 = p_1(x_1 - x_n), \ldots, x_n = p_n(x_1 - x_n))$$

Deciding Type isomorphism

computational algebra

$$\mathbb{N}[x]/(x = p(x)) \hookrightarrow R \times \mathbb{Z}[x]/(x - p(x))$$

The analysis somewhat extends to the multivariate case

The structure of isomorphisms

Example:

datatype $E = b$ of $E * E$

fun $p_1(x) =$ **case** x **of**
$\quad b(x_1, x_2)$
$\quad\quad \Rightarrow$ **case** x_1 **of**
$\quad\quad\quad b(x_{11}, x_{12})$
$\quad\quad\quad\quad \Rightarrow$ **case** x_{11} **of**
$\quad\quad\quad\quad\quad b(x_{111}, x_{112})$
$\quad\quad\quad\quad\quad\quad \Rightarrow$
$\quad\quad b(\, b(\, x_{111}, b(x_{112}, x_{12})), x_2)$

$$p_1(b(b(b(x_{111}, x_{112}), x_{12}), x_2))$$
$$= b(b(x_{111}, b(x_{112}, x_{12})), x_2)$$

$p_1 =$ 

Thm: The automorphism group of the generic commutative idempotent is Richard Thompson's group V.

The groupoid underlying

$$G[X]/(X \cong X \times X)$$

is

$$SM[X]/(X \cong X \otimes X)$$

and equivalent to

$$1 + V$$

- V is a very interesting (symmetry) group!

geometric view

Thm: The automorphism group of the generic idempotent is Richard Thompson's group F.

The groupoid

$$\mathcal{M}[X]/(X \cong X \bullet X)$$

is equivalent to

$$1 + F$$

- F is a very interesting (symmetry) group!

geometric view

Two proofs

Syntactic

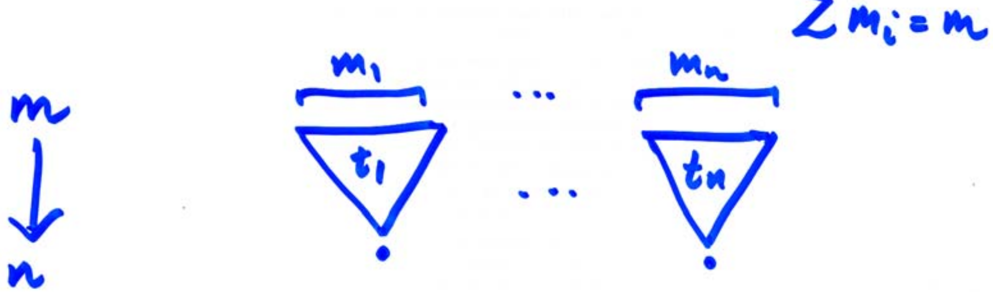
Type-theoretic analysis leads to isomorphisms being given by pattern-mappings of the form

$$b(b(x_{11}, x_{12}), x_2) \mapsto b(x_{11}, b(x_{12}, x_2))$$

Semantic

1. Free monoidal category on an operator $X \bullet X \to X$.
2. Free monoidal groupoid.

Free monoidal category on a binary operator

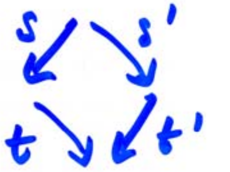


Free (monoidal) groupoid on a (monoidal) category

Morphisms

where

whenever



... This can be greatly simplified by some rewriting ...

... for nice categories, like the one we are concerned with, the rewriting system

$$\overset{u}{\swarrow}\overset{t}{\searrow}\overset{t'}{\swarrow}\overset{u'}{\searrow} \rightsquigarrow \overset{us}{\swarrow}\overset{u's'}{\searrow}$$

whenever $ts = t's'$

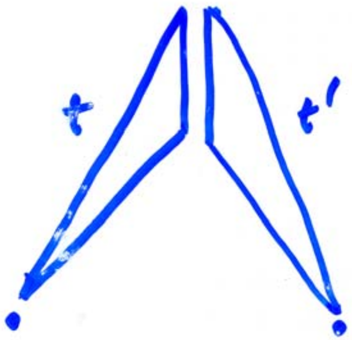
$$\overset{ts}{\swarrow}\overset{t's}{\searrow} \rightsquigarrow \overset{t}{\swarrow}\overset{t'}{\searrow} \qquad s \neq id$$

is locally confluent and terminating

So the automorphism group of the generic idempotent in

$$\mathcal{M}[x]/(x \cong x\cdot x) = \mathcal{F}g\left(\mathcal{M}[x]/(x\cdot x \to x)\right)$$

has morphisms given by tree-cotree pairs



in normal form. (Composition is given by taking linear m.g.u.s.)

An application to algorithms

Sean Cleary

Problem: give an [efficient] algorithm that given any two binary trees with the same number of leaves yields a [minimum] number of rotations transforming one tree into the other.

REFERENCES

- M. Fiore and T. Leinster. A simple description of Thompson's group F. <www.arxiv.org/abs/math.GR/0508617>
- M. Fiore and T. Leinster. Objects of categories as complex numbers. Advances in Mathematics, 190(2):264-277, 2005.
- M. Fiore. Isomorphisms of generic recursive polynomial types. In 31st Symp. on Principles of Programming Languages, POPL 2004, pages 77-88. ACM Press, 2004.
- S. Cleary. Restricted rotation distance between binary trees. Information Processing Letters, 84(6), 2002.
- J. Lucas. A direct algorithm for restricted rotation distance. Information Processing Letters, 90(3):129-134, 2004.